VULCAN Analysis Combined

Andrew Holding

1/22/2018

Contents

1 Introduction

To reduce the overall size of this project read count and network data is stored within Rdata files. These are avalible from https://github.com/andrewholding/VULCANSupplementary. The raw data is avalible from the gene expression omnibus if reprocessing is required. The following script will download all the required files and rebuild the network file. The Rmd files are designed to recreate the processed data from the raw data if the processed data is not found.

```
git clone https://github.com/andrewholding/VULCANSupplementary.git

#Rebuild network
cd ChIP-seq_ER/data/
cat brca-expmat.rda.part1 brca-expmat.rda.part2 > brca-expmat.rda
```

2 Estrogen Receptor-alpha qPCR Analysis

2.1 Delta-delta ct Calculation

```
#Set varibles to make time points easier to spot
time0 <- 1
time45 <- 2
time90 <- 3

#Input raw data
qpcr <- list()
qpcr[[time0]] <- read.table("qpcr-0.txt",
                            header = T,
                            sep = "\t",
                            as.is = T)
qpcr[[time45]] <-
    read.table("qpcr-45.txt",
               header = T,
```

```
                sep = "\t",
                as.is = T)
qpcr[[time90]] <-
    read.table("qpcr-90.txt",
                header = T,
                sep = "\t",
                as.is = T)

for (time_point in time0:time90)
{
    qpcr[[time_point]][qpcr[[time_point]] == "No Ct"] <- NA
}


#Prepare lists
ct_values <- list()
ct_values_means <- list()
d_ct_values_means <- list()
input_values <- list()
input_mean <- list()
d_input_mean <- list()
dd_ct_values_means <- list()


for (time_point in time0:time90)
{
    #Create object
    ct_values[[time_point]] <- matrix(NA, 0, 2)

    #Add reps 1:8 to object
    for (rep in 1:8) {
        y_coord <- rep * 12 - 11
        ct_values[[time_point]] <- rbind(ct_values[[time_point]],
                                         cbind(
                                             as.numeric(qpcr[[time_point]]$Ct..dR[y_coord:(y_coord + 4)]
                                             as.numeric(qpcr[[time_point]]$Ct..dR[(y_coord + 5):(y_coor
                                                                                                 9)]

                                         ))
    }



    #Average techical reps from same isogenic replicate
    ct_values_means[[time_point]] <- matrix(NA, 0, 2)
    for (rep in 1:8) {
        y_coord <- rep * 5 - 4
        ct_values_means[[time_point]] <-
            rbind(ct_values_means[[time_point]], colMeans(ct_values[[time_point]][y_coord:(y_coord +
                                                                                           4), 1:2]
    }

    #Import Input from time_point mins
```

```
    input_values[[time_point]] <- cbind(as.numeric(qpcr[[time_point]]$Ct..dR[12 *
                                                                              (0:4) + 11]),
                                        as.numeric(qpcr[[time_point]]$Ct..dR[12 *
                                                                              (0:4) + 12]))

    #Calucate Average of technical reps.
    input_mean[[time_point]] <-
        colMeans(input_values[[time_point]], na.rm = T)

    if (time_point == 2)
    {
        #Plate layout is reversed wrt control and target primers. See lab book.
        d_ct_values_means[[time_point]] <-
            ct_values_means[[time_point]][, 2] - ct_values_means[[time_point]][, 1]
        d_input_mean[[time_point]] <-
            input_mean[[time_point]][2] - input_mean[[time_point]][1]

        dd_ct_values_means[[time_point]] <-
            d_ct_values_means[[time_point]] - d_input_mean[[time_point]]
    } else {
        d_ct_values_means[[time_point]] <-
            ct_values_means[[time_point]][, 1] - ct_values_means[[time_point]][, 2]
        d_input_mean[[time_point]] <-
            input_mean[[time_point]][1] - input_mean[[time_point]][2]

        dd_ct_values_means[[time_point]] <-
            d_ct_values_means[[time_point]] - d_input_mean[[time_point]]
    }
}

df_ct <- data.frame(cbind(
    2 ^ dd_ct_values_means[[time0]],
    2 ^ dd_ct_values_means[[time45]],
    2 ^ dd_ct_values_means[[time90]]
))
colnames(df_ct) <- c("0 min", "45 mins", "90 mins")
```

2.2 Box plot

```
set.seed(1)
par(mar = c(5, 5, 3, 1))
boxplot(
    df_ct,
    main = "ER-alpha occupany of the TFF1 promoter",
    range = 1,
    ylab = expression(paste(
        "Enrichment over Input, 2" ^ "delta-delta-Ct"
    )),
    xlab = "Time point",
    pch = NA
)
```

```
stripchart(
    df_ct,
    add = T,
    vertical = T,
    pch = 20,
    method = "jitter",
    jitter = 0.05
)
```

3 Estrogen Receptor-alpha ChIP-Seq and VULCAN Analysis

3.1 Introduction

Vulcan (VirtUaL ChIP-Seq Analysis through Networks) is a pipeline that combines ChIP-Seq data and regulatory networks to obtain transcription factors that are likely affected by a specific stimulus. In order to do so, our package combines strategies from different BioConductor packages: *DESeq* for data normalization, *ChIPpeakAnno* and *DiffBind* for annotation and definition of ChIP-Seq genomic peaks, *csaw* to define optimal peak width and *viper* for applying a regulatory network over a differential binding signature. Usage of gene regulatory networks to analyze biological systems has witnessed an exponential increase in the last decade, due to the ease of obtaining genome-wide expression data (Margolin et al., 2005; Giorgi et al., 2013; Castro et al., 2015). Recently, the VIPER approach to interrogate these network models has been proposed to infer transcription factor activity using the expression of a collection of their putative targets, i.e. their regulon (Alvarez et al., 2016). In the VIPER algorithm, gene-level differential expression signatures are obtained for either individual samples (relative to the mean of the dataset) or between groups of samples, and regulons are tested for enrichment. Ideally, TF regulons can be tested on promoter-level differential binding signatures generated from ChIP-Seq experiments, in order to ascertain the global change in promoter occupancy for gene sets. In our study, we propose an extension of the VIPER algorithm to specifically analyze TF occupancy in ChIP-Seq experiments. Our VULCAN algorithm uses ChIP-Seq data obtained for a given TF to provide candidate coregulators of the response to a given stimulus. The analysis is based on identifying differentially bound genes and testing their enrichment in the regulon of potential co-regulatory factors.

3.2 Software setup

VULCAN is conveniently provided as an R package available from the Bioconductor repository.

```
#source("http://bioconductor.org/biocLite.R")
#biocLite("vulcan")
library(vulcan)
```

Other packages are required for the execution of the analysis and the visualization of the results. A *results* folder will be also created to store intermediate steps of the analysis.

```
library(gplots) # for heatmap.2
library(org.Hs.eg.db)
library(gridExtra) # for the pathway part
library(GeneNet) # to generate partial correlation networks
if(!file.exists("results")){dir.create("results")}
```

Finally, we will define here some convenience functions, such as those to convert from gene symbols to entrez ids, and viceversa.

```
list_eg2symbol<-as.list(org.Hs.egSYMBOL[mappedkeys(org.Hs.egSYMBOL)])
e2s<-function(ids){
```

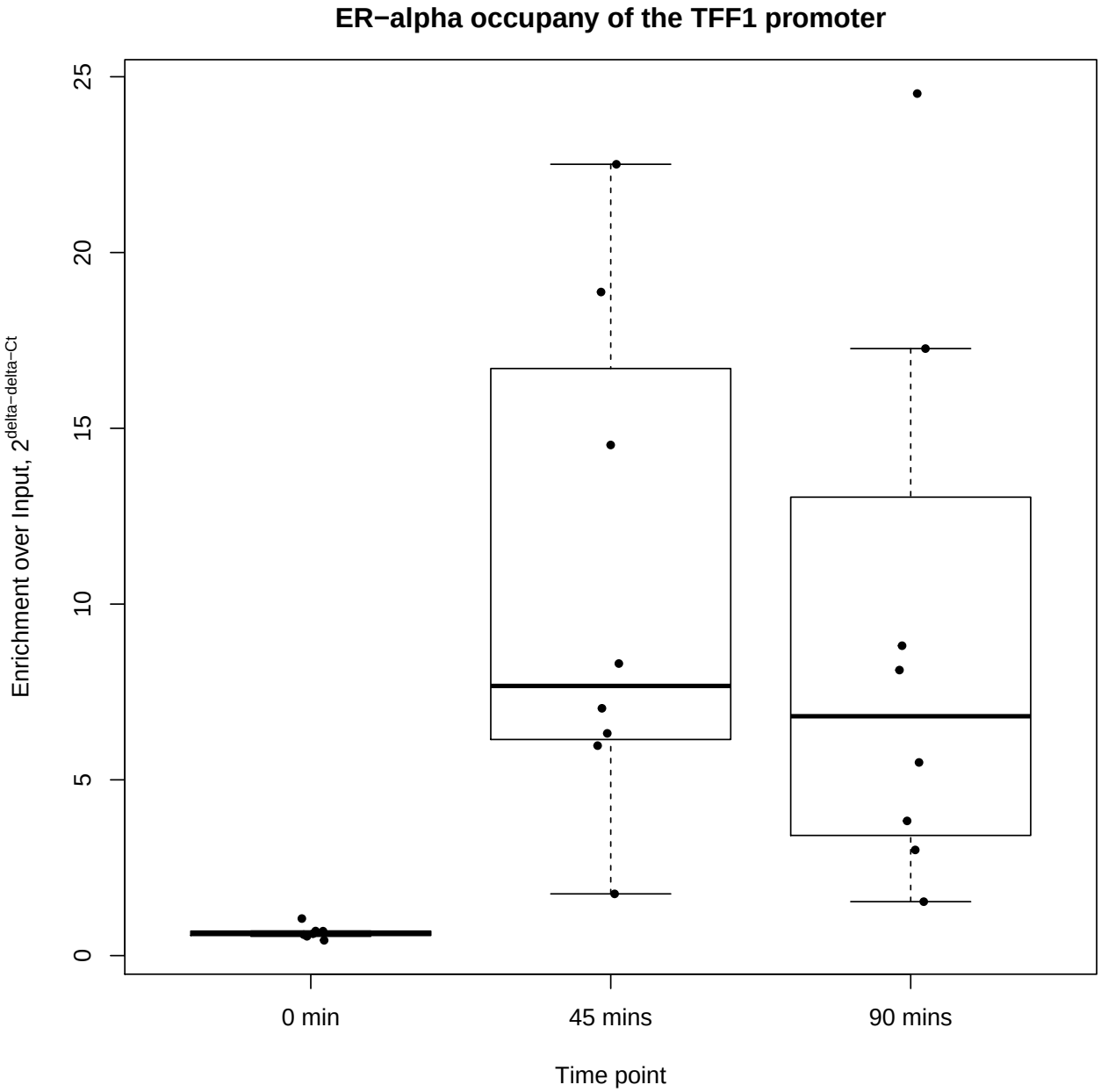


Figure 1: Box Plot showing ChIP enrichment over control loci for ER at 0, 45 and 90 minutes after stimuation with 100nM E2

```
  ids <- as.character(ids)
  outlist <- list_eg2symbol[ids]
  names(outlist) <- ids
  outlist[is.na(outlist)] <- paste("unknown.", ids[is.na(outlist)], sep = "")
  outlist <- gsub("unknown.unknown.", "", outlist)
  return(outlist)
}
list_symbol2eg <- as.character(org.Hs.egSYMBOL2EG[mappedkeys(org.Hs.egSYMBOL2EG)])
s2e<-function(ids){
  ids <- as.character(ids)
  outlist <- list_symbol2eg[ids]
  names(outlist) <- ids
  outlist[is.na(outlist)] <- paste("unknown.", ids[is.na(outlist)], sep = "")
  outlist <- gsub("unknown.unknown.", "", outlist)
  return(outlist)
}
```

3.3 Import ChIP-Seq data using Vulcan

ChIP-Seq data was generated using a cell line model of ER+ breast cancer, MCF7, at 0', 45' and 90' after stimulation with estradiol. The binding profile of ER at each timepoint was then compared between time points using differential binding analysis. From the temporal comparison ER binding we established four classes of binding pattern: early responders, repressed transcription factors, late responders and candidate cyclic genes.

The first part of our analysis highlights how to import ChIP-Seq data using the VULCAN adn DiffBind packages, provided alignment files (BAM format) and peak files (BED format) for each of the samples. In our dataset, we have 4 replicates for each time point.

3.3.1 Initial import

VULCAN requires an input sheet file in CSV format describing the samples and the location of the individual input files

```
sheetfile<-"chipseq/holding/sheet.csv"

fname<-"results/001_input.rda"
if(!file.exists(fname)){
  vobj<-vulcan.import(sheetfile)
  save(vobj,file=fname)
} else {
  load(fname)
}
```

3.3.2 Annotation

VULCAN identifies the location of each peak in each sample, and according to the selected method, assigns a score to each gene promoter. There are a few methods available.

- ***sum***: when multiple peaks are found, sum their contributions
- ***closest***: when multiple peaks are found, keep only the closest to the TSS as the representative one
- ***strongest***: when multiple peaks are found, keep the strongest as the representative one

- ***farthest***: when multiple peaks are found, keep only the closest to the TSS as the representative one
- ***topvar***: when multiple peaks are found, keep the most varying as the representative one
- ***lowvar***: when multiple peaks are found, keep the least varying as the representative one

```
vobj<-vulcan.annotate(vobj,lborder=-10000,rborder=10000,method="sum")
```

3.3.3 Normalization

At this step, genes are quantified according to the number of reads that could be associated to their promoters. The algorithms within VULCAN (*viper* and *DESeq2*) require however data to be normalized via Variance-Stabilizing Transformation (Anders and Huber, 2010).

```
vobj<-vulcan.normalize(vobj)
save(vobj,file="results/001_vobj.rda")
```

3.3.4 Dataset Sample Clustering

Here, we show how the samples cluster together using peak raw counts, VST-normalized peak raw counts and peak RPKMs. In the following heatmaps, there are 4288 rows, one per gene promoter, and 16 columns, one per sample. The sample name indicates the time point in minutes (T0, T45 or T90) and the number of replicate (_1, _2, _3, _4).

```
heatmap.2(vobj$rawcounts,scale="row",
          col=colorpanel(1000,"navy","white","red3"),tracecol="black",labRow="")
mtext("Raw Counts")
```

```
heatmap.2(vobj$normalized,
          col=colorpanel(1000,"navy","white","red3"),tracecol="black",labRow="")
mtext("VST-normalized")
```

```
heatmap.2(vobj$rpkms,scale="row",
          col=colorpanel(1000,"navy","white","red3"),tracecol="black",labRow="")
mtext("RPKMs")
```

Principal Component Analysis further confirms the clustering of replicates in two distinct groups: untreated (T0) and treated (T45/T90).

```
topvar<-apply(vobj$normalized[,],1,var)
topvar<-sort(topvar,dec=TRUE)[1:500]
submat<-vobj$normalized[names(topvar),]
pca<-prcomp(t(submat))
vars<-100*(pca$sdev^2)/sum(pca$sdev^2)
vars<-signif(vars,3)

p1<-1
p2<-2
plot(pca$x[,p1],pca$x[,p2],
     xlab=paste0("PC",p1," (Var.Explained: ",vars[p1],"%)"),
     ylab=paste0("PC",p2," (Var.Explained: ",vars[p2],"%)"),
     type="p",main="PC Analysis",pch=20,col="grey",
     xlim=c(min(pca$x[,p1])*1.1,max(pca$x[,p1])*1.1)
)
mtext("VST-Normalized data",cex=0.8)
textplot2(pca$x[,p1],pca$x[,p2],new=FALSE,
          words=gsub("_[0-9]","",rownames(pca$x),perl=TRUE)
```

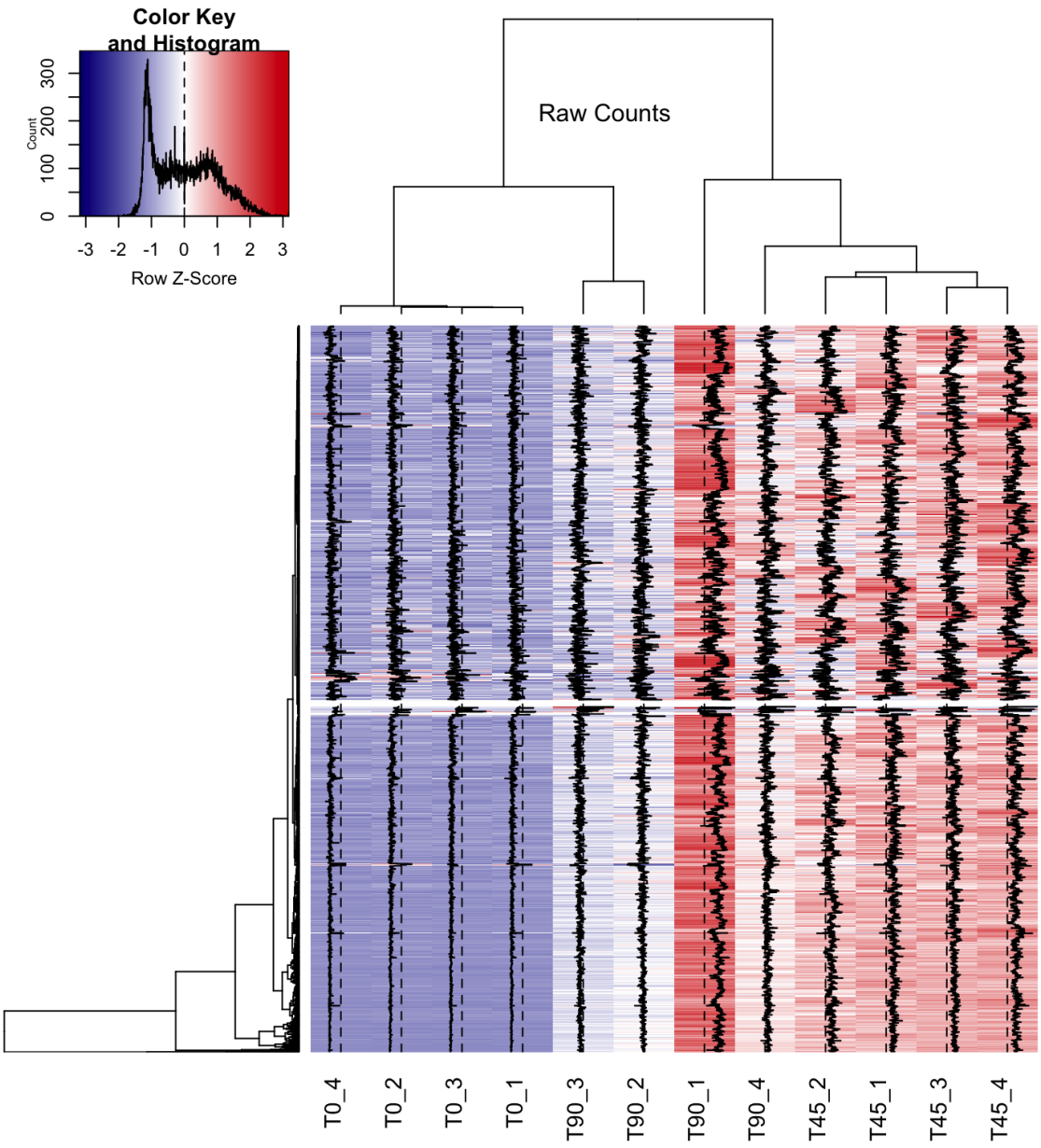


Figure 2: Dataset clustering with Raw Counts

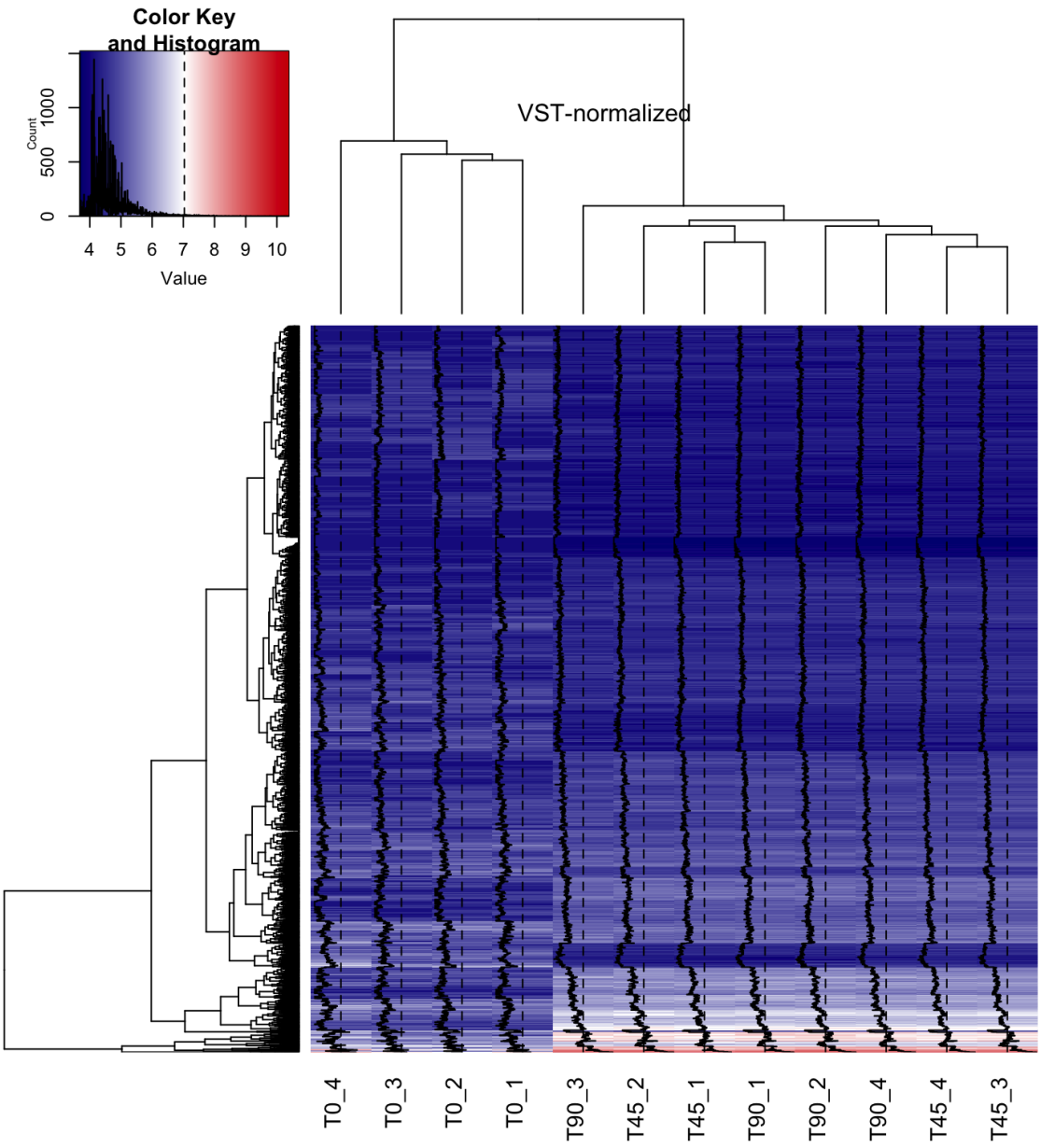


Figure 3: Dataset clustering with VST-normalized Counts

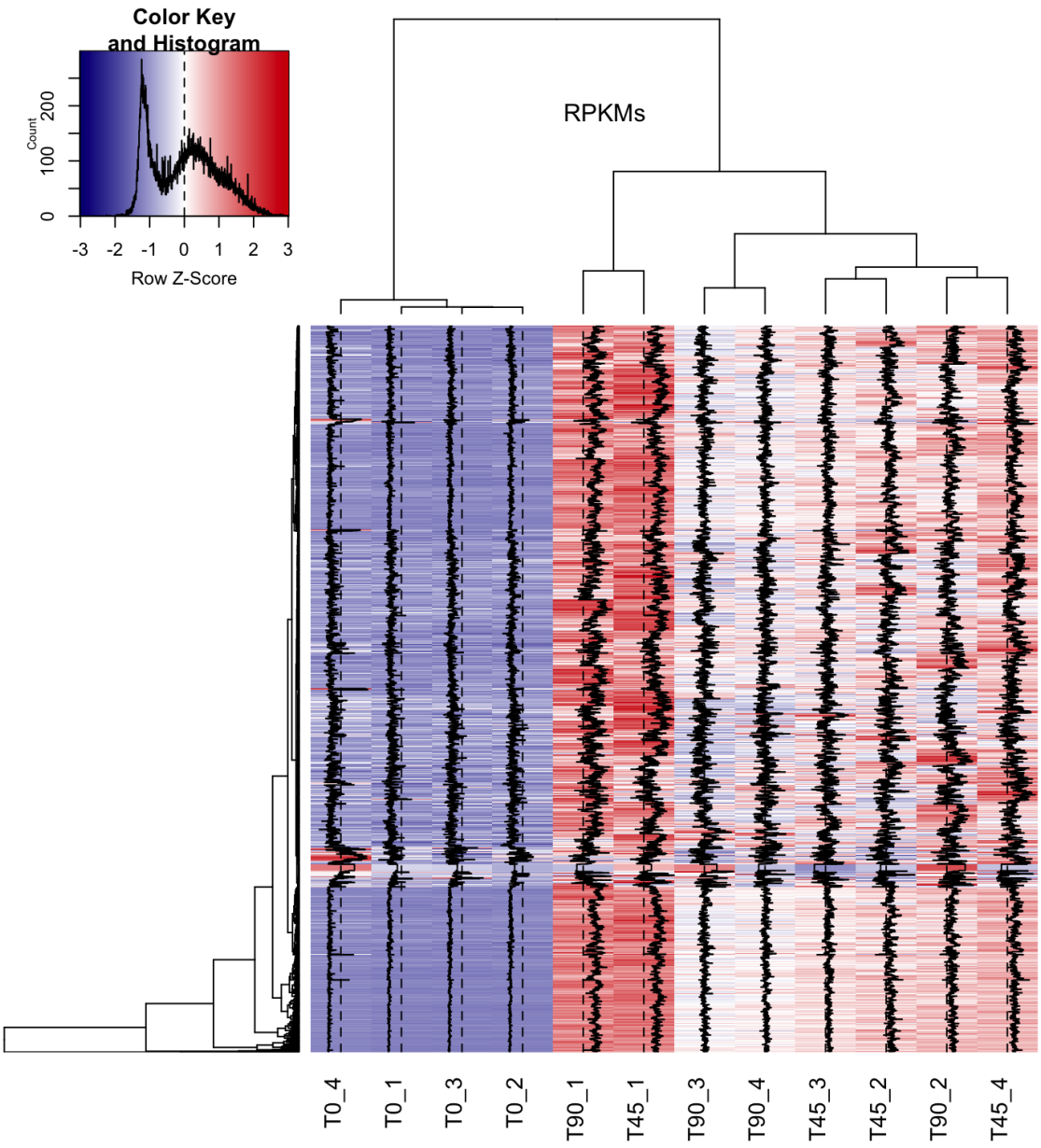


Figure 4: Dataset clustering with RPKMs

```
)
grid()
```

A clearer distinction of T45 and T90 is highlighted by PC5:

```
p1<-1
p2<-5
plot(pca$x[,p1],pca$x[,p2],
     xlab=paste0("PC",p1," (Var.Explained: ",vars[p1],"%)"),
     ylab=paste0("PC",p2," (Var.Explained: ",vars[p2],"%)"),
     type="p",main="PC Analysis",pch=20,col="grey",
     xlim=c(min(pca$x[,p1])*1.1,max(pca$x[,p1])*1.1)
)
mtext("VST-Normalized data",cex=0.8)
textplot2(pca$x[,p1],pca$x[,p2],new=FALSE,
          words=gsub("_[0-9]","",rownames(pca$x),perl=TRUE)
)
grid()
```

3.3.5 MA plots

Without changing the format of the input data, we can use the Bioconductor DiffBind package to visualize the amplitude of changes in ER binding between time points. One way to do this is an MA plot, which shows the differences between measurements taken in two groups (e.g. 45' vs 00'), by transforming the promoter peak intensity data onto M (log ratio) and A (mean average) scales.

We will process the data using the DiffBind package and then use the *dba.plotMA* function to visualize the contrasts (which we can extract using the *dba.contrast* function).

```
sheetfile<-"chipseq/holding/sheet.csv"
fname<-"results/001_diffbind.rda"
if(!file.exists(fname)){
  # Load a sample sheet
  chipseqSamples<-read.csv(sheetfile)
  dbaobj<-dba(sampleSheet=chipseqSamples)

  # Count reads 500bp either side of summits, a peak is considered a peak
  # if it is found in 3 samples out of loaded samples.
  dbaobj<-dba.count(dbaobj,minOverlap=3,summits=500)

  # Define contrasts
  dbaobj<-dba.contrast(dbaobj, categories=DBA_CONDITION)

  # Calculate differential binding using the DESeq2 engine within DiffBind
  dbaobj<-dba.analyze(dbaobj,method=DBA_DESEQ2)

  # Save the DiffBind object
  save(dbaobj,file=fname)
} else {
  load(fname)
}

# Visualize each contrast
dba.contrast(dbaobj)
```

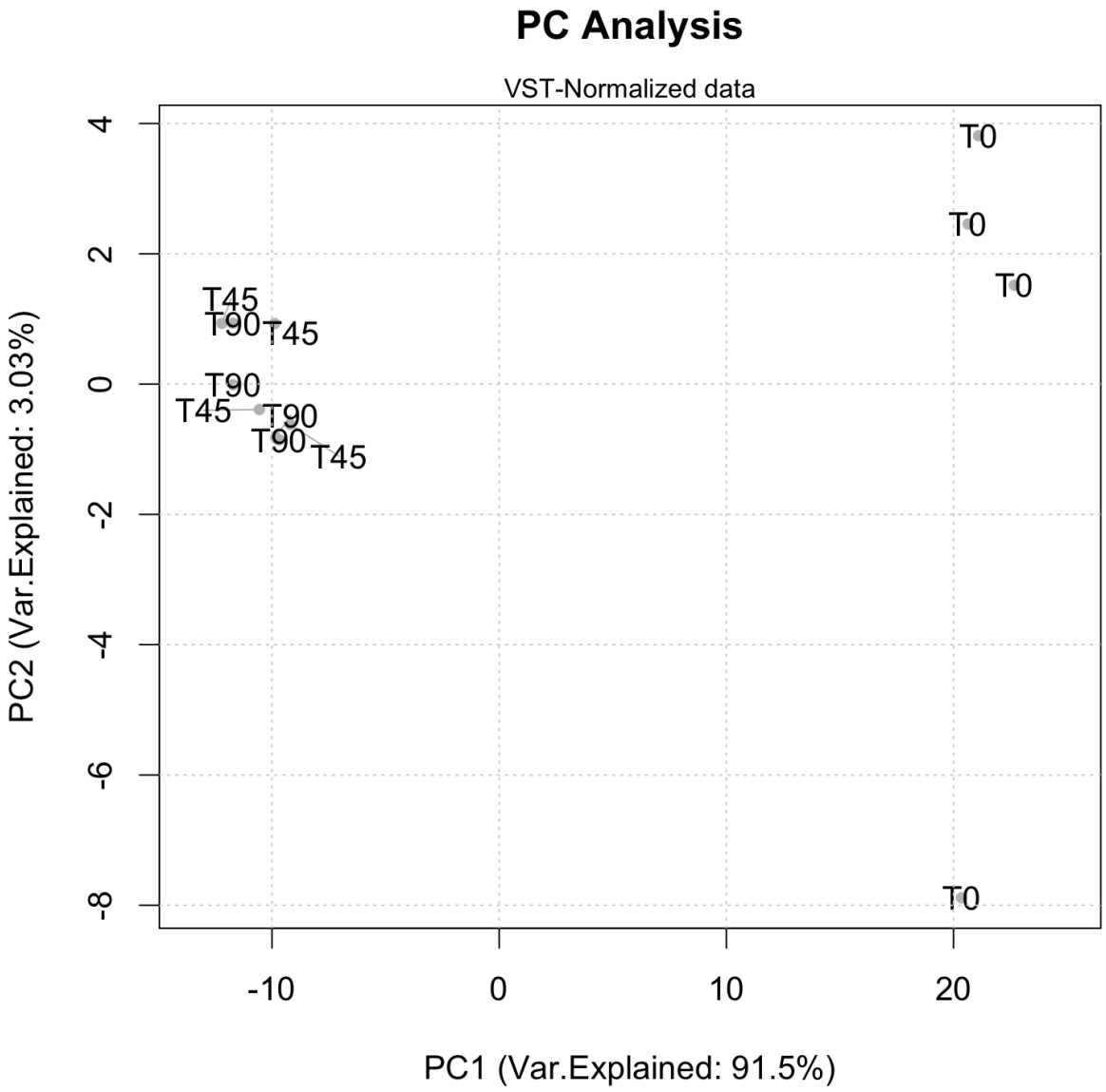


Figure 5: Principal Component Analysis of the dataset, highlighting components 1 and 2

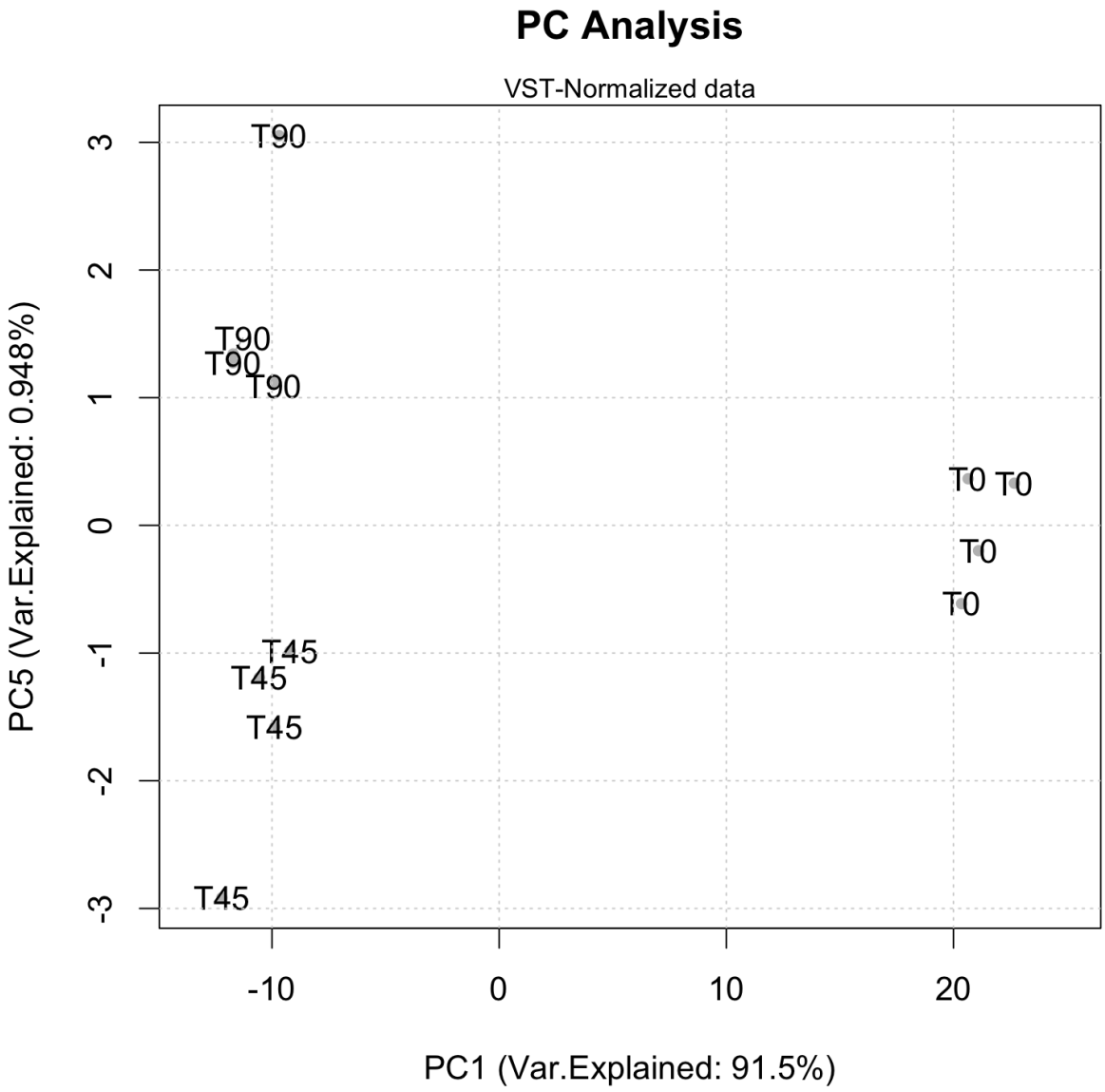


Figure 6: Principal Component Analysis of the dataset, highlighting components 1 and 5

```
## 12 Samples, 19351 sites in matrix:
##       ID Tissue Factor Condition Replicate Caller Intervals FRiP
## 1  T90_1   MCF7     ER       t90         1 counts     19351 0.15
## 2  T90_2   MCF7     ER       t90         2 counts     19351 0.12
## 3  T90_3   MCF7     ER       t90         3 counts     19351 0.09
## 4  T90_4   MCF7     ER       t90         4 counts     19351 0.09
## 5  T45_1   MCF7     ER       t45         1 counts     19351 0.16
## 6  T45_2   MCF7     ER       t45         2 counts     19351 0.11
## 7  T45_3   MCF7     ER       t45         3 counts     19351 0.11
## 8  T45_4   MCF7     ER       t45         4 counts     19351 0.13
## 9   T0_1   MCF7     ER        t0         1 counts     19351 0.01
## 10  T0_2   MCF7     ER        t0         2 counts     19351 0.01
## 11  T0_3   MCF7     ER        t0         3 counts     19351 0.01
## 12  T0_4   MCF7     ER        t0         4 counts     19351 0.02
##
## 3 Contrasts:
##   Group1 Members1 Group2 Members2
## 1    t90        4    t45        4
## 2    t90        4     t0        4
## 3    t45        4     t0        4
```

```
dba.plotMA(dbaobj,contrast=1,method=DBA_DESEQ2)
```

```
dba.plotMA(dbaobj,contrast=2,method=DBA_DESEQ2)
```

```
dba.plotMA(dbaobj,contrast=3,method=DBA_DESEQ2)
```

3.4 Apply a coregulatory network over a ChIP-Seq profile

Once the data has been loaded, VULCAN applies a regulatory network over differential binding signatures to define Transcription Factors whose networks are most affected by the treatment. In our analysis, we will be using three different networks generated via ARACNe (Margolin et al., 2006): two are breast cancer-specific and are derived from the TCGA and METABRIC data collection, respectively. A third network is used as a negative control, highlighting regulatory mechanisms derived from the Amyloid Leukemia dataset (TCGA). ### Loading the input networks and processed data First, we will load the output from the previous paragraphs (chipseq data annotated and normalized) and the three networks.

```
# Load imported vulcan object
load("results/001_vobj.rda")

# Vulcan Analysis (multiple networks)
## Networks
load("networks/laml-tf-regulon.rda")
laml_regulon<-regul
rm(regul)

load("networks/brca-tf-regulon.rda")
tcga_regulon<-regul
rm(regul)

load("networks/metabric-regulon-tfs.rda")
metabric_regulon<-regulon
rm(regulon)
```

Accessing the vulcan object /texit{vobj} can give us informations on the sample groups thereby contained.

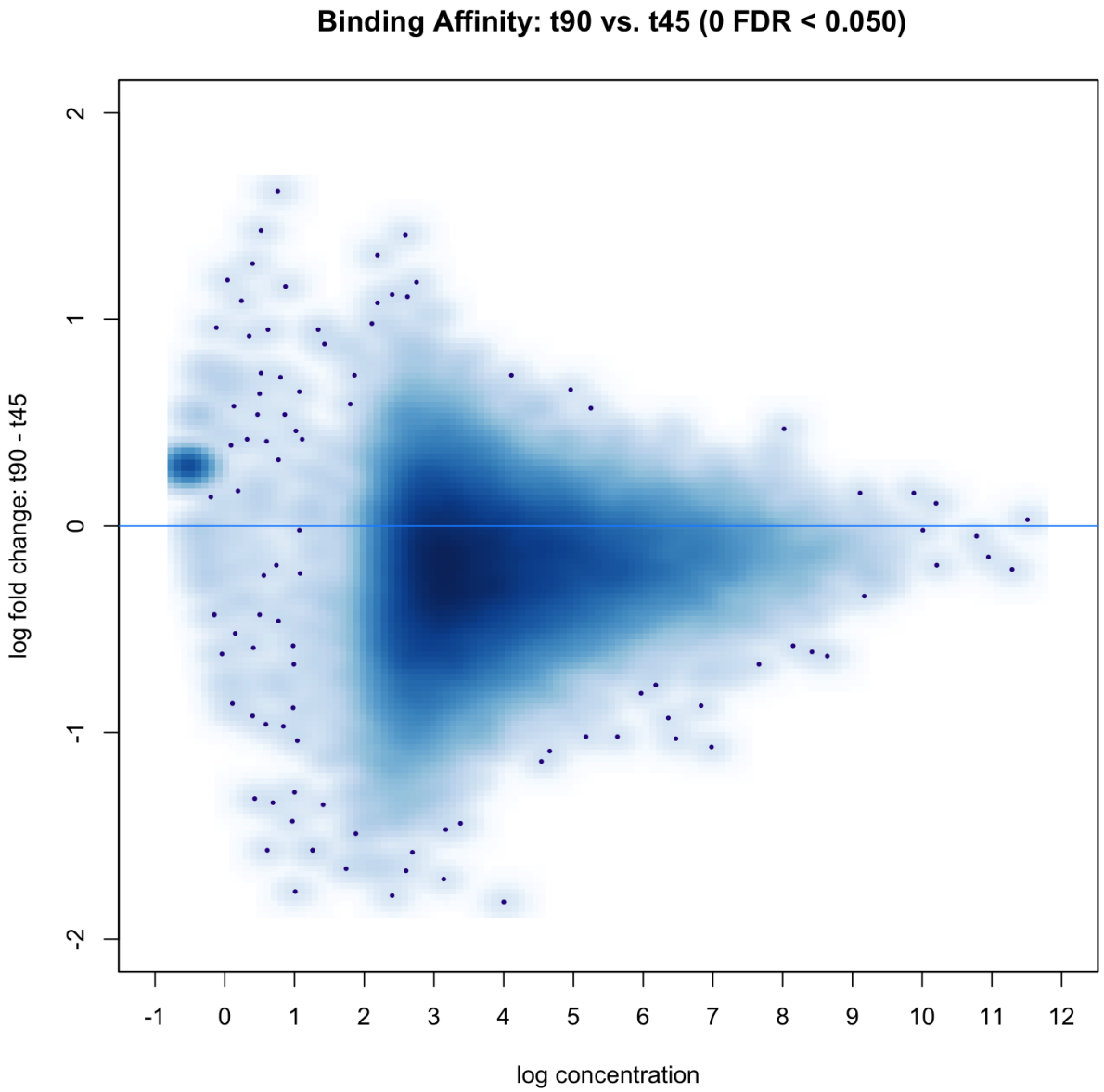


Figure 7: MA plot for Contrast 90mins vs 45mins, highlighting each individual peak. Significant peaks are highlighted in red

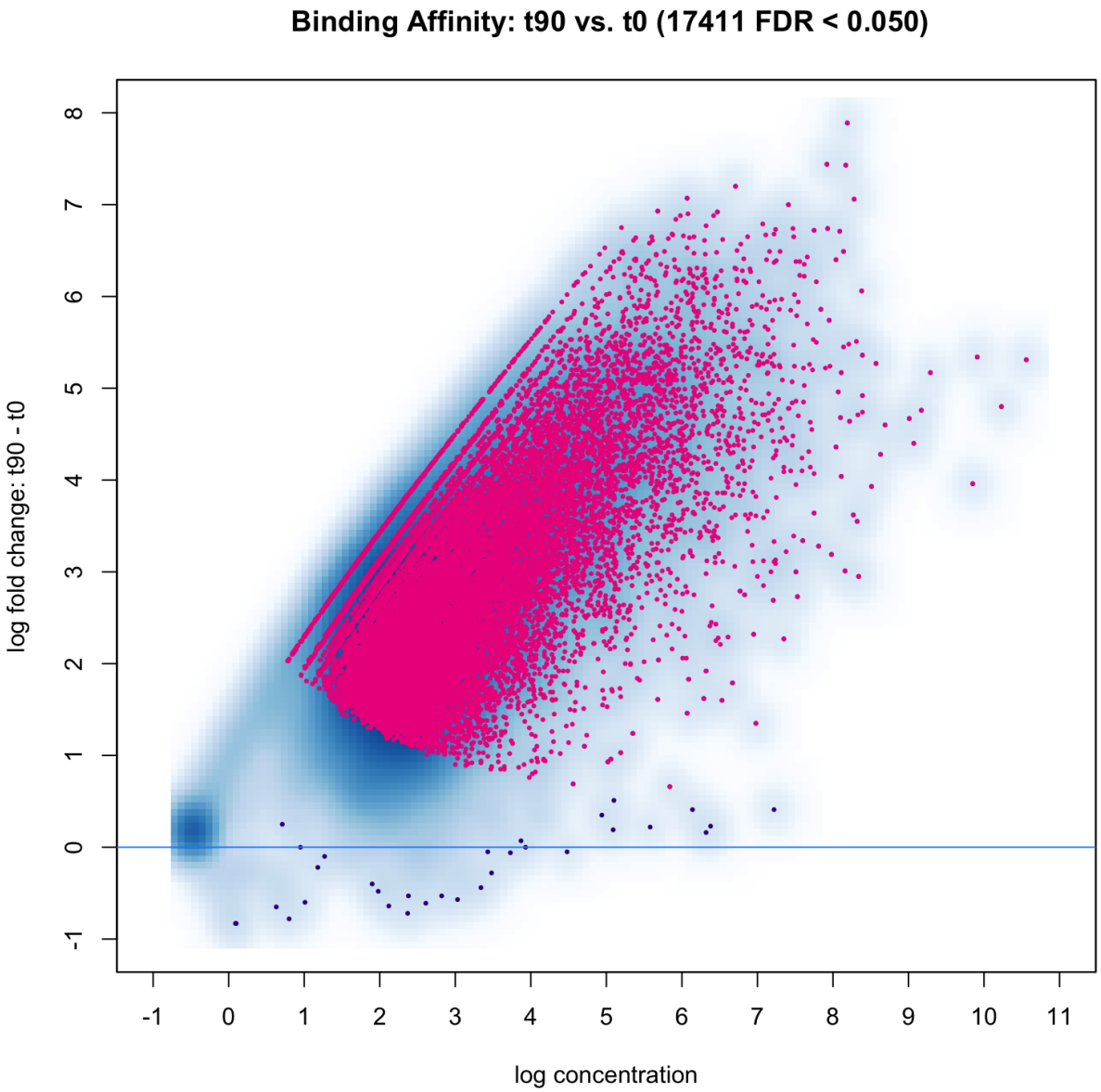


Figure 8: MA plot for Contrast 90mins vs 00mins, highlighting each individual peak. Significant peaks are highlighted in red

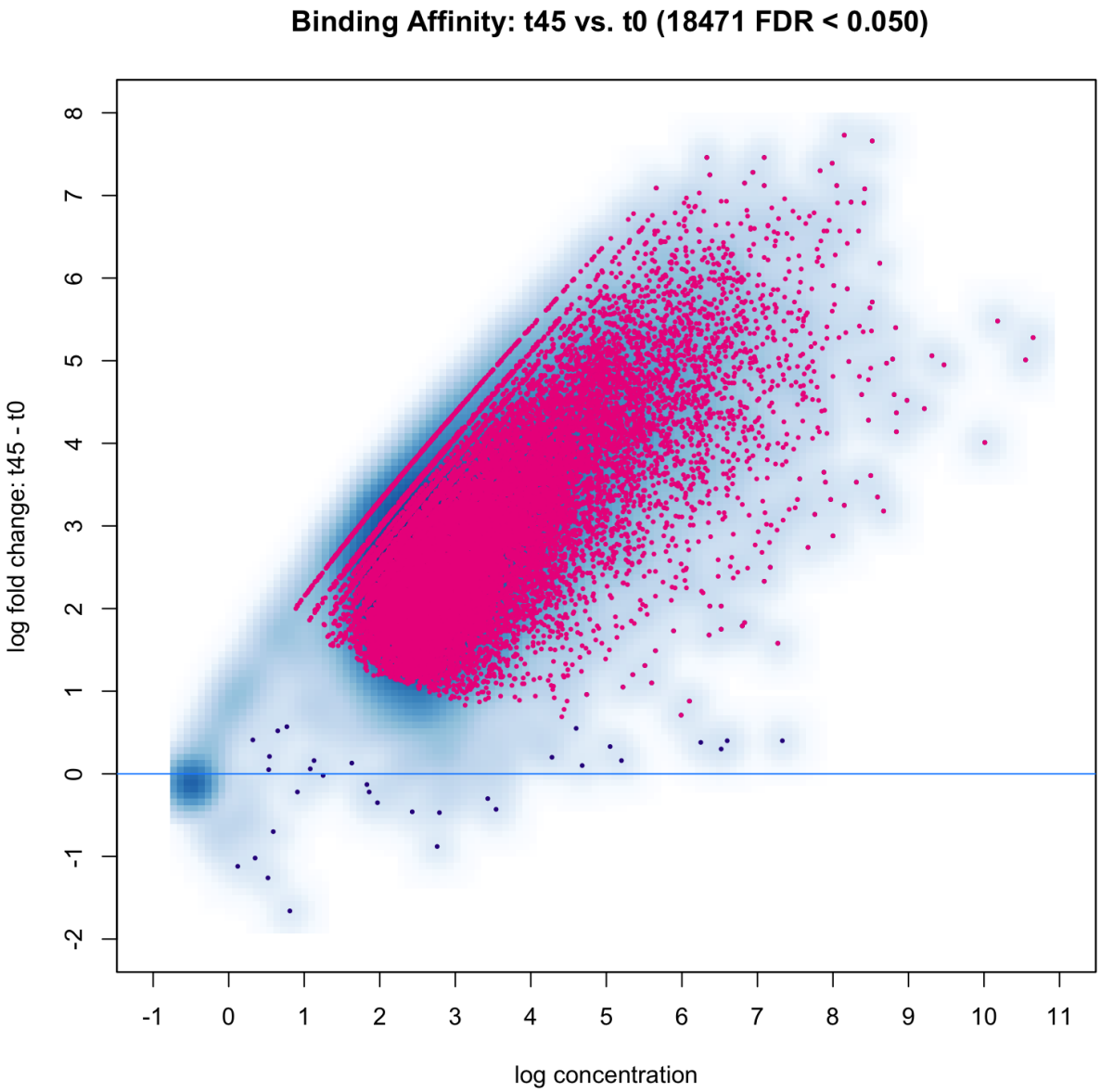


Figure 9: MA plot for Contrast 45mins vs 00mins, highlighting each individual peak. Significant peaks are highlighted in red

```
names(vobj$samples)
```

```
## [1] "t90" "t45" "t0"
```

3.4.1 Running the core Vulcan function

The final Vulcan pipeline step requires three input objects:

- The annotated and normalized chipseq data (/texit{vobj})
- A specific binding signature defined by a contrast between two sample groups
- A regulatory network

```
fname<-"results/002_vobj_networks.rda"
list_eg2symbol<-as.list(org.Hs.egSYMBOL[mappedkeys(org.Hs.egSYMBOL)])
if(!file.exists(fname)){
  vobj_tcga_90<-vulcan(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  vobj_tcga_45<-vulcan(vobj,network=tcga_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)
  vobj_metabric_90<-vulcan(vobj,network=metabric_regulon,contrast=c("t90","t0"),annotation=list_eg2symbo
  vobj_metabric_45<-vulcan(vobj,network=metabric_regulon,contrast=c("t45","t0"),annotation=list_eg2symbo
  vobj_negative_90<-vulcan(vobj,network=laml_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  vobj_negative_45<-vulcan(vobj,network=laml_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)
  save(
    vobj_tcga_90,
    vobj_tcga_45,
    vobj_metabric_90,
    vobj_metabric_45,
    vobj_negative_90,
    vobj_negative_45,
    file=fname
  )
} else {
  load(fname)
}
```

3.4.2 Visualize the relative TF activity

Vulcan has now generated activity scores for each Transcription Factor, which specify the global ER binding strength on the promoters of their targets.

3.4.2.1 METABRIC results

The following line plot shows the relative network activity for every TF at two time points. On the y-axis, the Normalized Enrichment Score of Vulcan is provided. Dashed lines indicate a p=0.05 significance threshold for up- and down-regulation.

```
threshold<-p2z(0.05)
metabric_90=vobj_metabric_90$mrs[,"NES"]
metabric_45=vobj_metabric_45$mrs[,"NES"]
tfs<-names(metabric_45)
metabricmat<-cbind(rep(0,length(metabric_45)),metabric_45[tfs],metabric_90[tfs])
colnames(metabricmat)<-c("T0","T45","T90")
## All TFs
matplot(t(metabricmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="All TFs",xlim=c(1,3.3
```

```
axis(1,at=c(1:3),labels=colnames(metabricmat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(t(metabricmat["ESR1",])),type="l",col="red",lty=1,lwd=2,add=TRUE)
text(3,metabricmat["ESR1",3],label="ESR1",pos=4,cex=0.6,font=2,col="red3")
mtext("METABRIC network")
```

These TFs can be grouped in classes. For example, TFs whose activity is already significant after 45mins and remains significant at 90 minutes after estradiol treatment can be dubbed **early responders**.

```
threshold<-p2z(0.05)
tfclass<-tfs[metabricmat[,"T45"]>=threshold&metabricmat[,"T90"]>=threshold]
matplot(t(metabricmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Early responders",xli
axis(1,at=c(1:3),labels=colnames(metabricmat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(metabricmat[tfclass,]),type="l",col="red3",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(metabricmat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
text(3,newcoords["ESR1"],label="ESR1",pos=4,cex=0.6,font=2,col="red3")
mtext("METABRIC network")
```

TFs whose *repressed* targets are bound will yield a negative activity score. These **repressed TFs** are symmetrically opposite to early responder TFs.

```
threshold<-p2z(0.05)
tfclass<-tfs[metabricmat[,"T45"]<=-threshold&metabricmat[,"T90"]<=-threshold]
matplot(t(metabricmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Repressed TFs",xlim=c
axis(1,at=c(1:3),labels=colnames(metabricmat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(metabricmat[tfclass,]),type="l",col="cornflowerblue",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-c(tfclass,"GRHL2")
oricoords<-sort(setNames(metabricmat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("METABRIC network")
```

Some TFs appear to have their targets bound at 45 minutes, but then unoccupied at 90 minutes. This "updown" behavior is consistent to what observed in previous literature about the cyclic properties of certain components of the ER DNA-binding complex, and therefore we dubbed them **candidate cyclic TFs**.

```
threshold<-p2z(0.05)
tfclass<-tfs[metabricmat[,"T45"]>=threshold&metabricmat[,"T90"]<=threshold]
matplot(t(metabricmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Candidate Cyclic TFs"
axis(1,at=c(1:3),labels=colnames(metabricmat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(metabricmat[tfclass,]),type="l",col="seagreen",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(metabricmat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("METABRIC network")
```

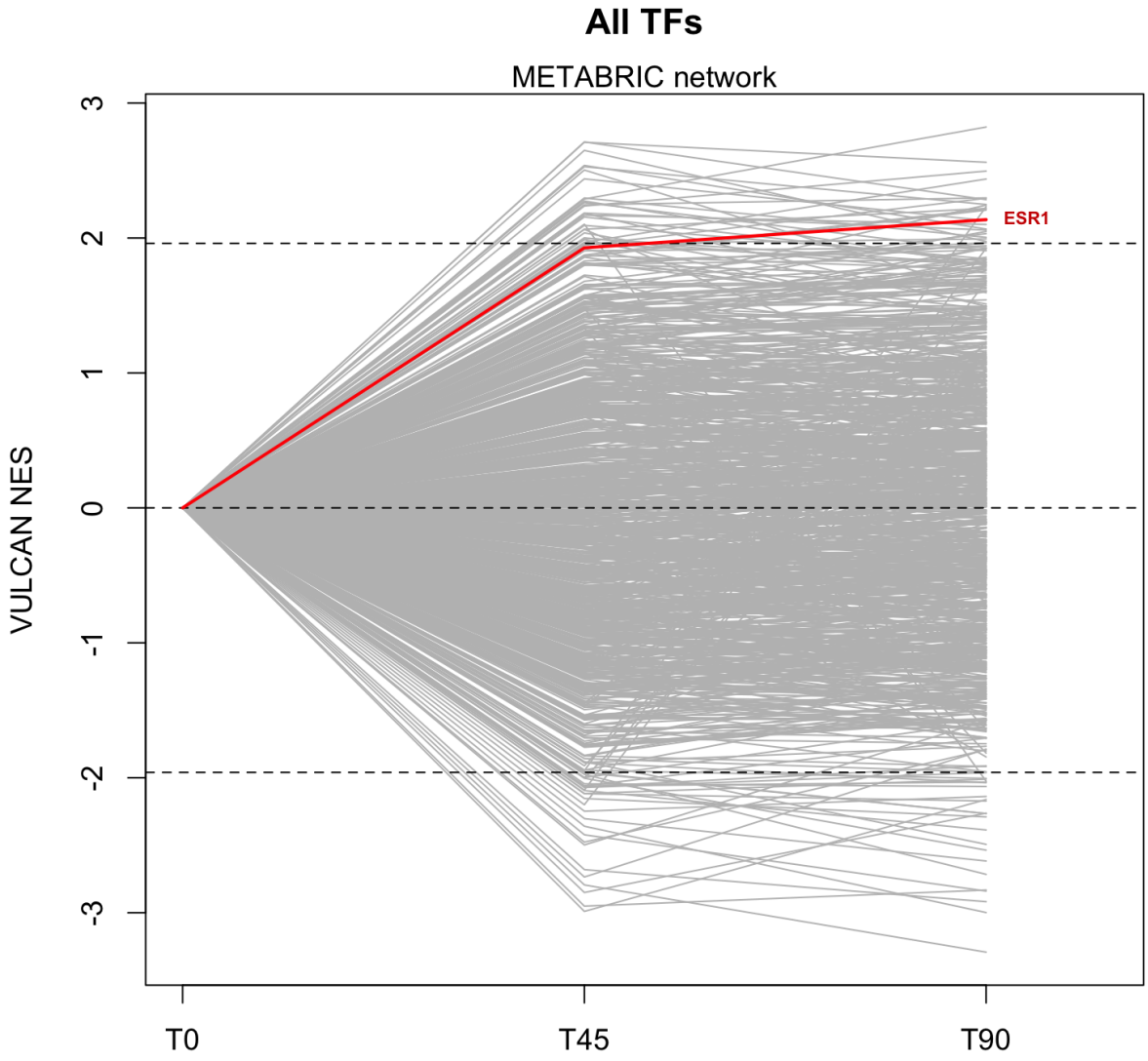


Figure 10: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the METABRIC network, highlighting the ESR1 TF as an example

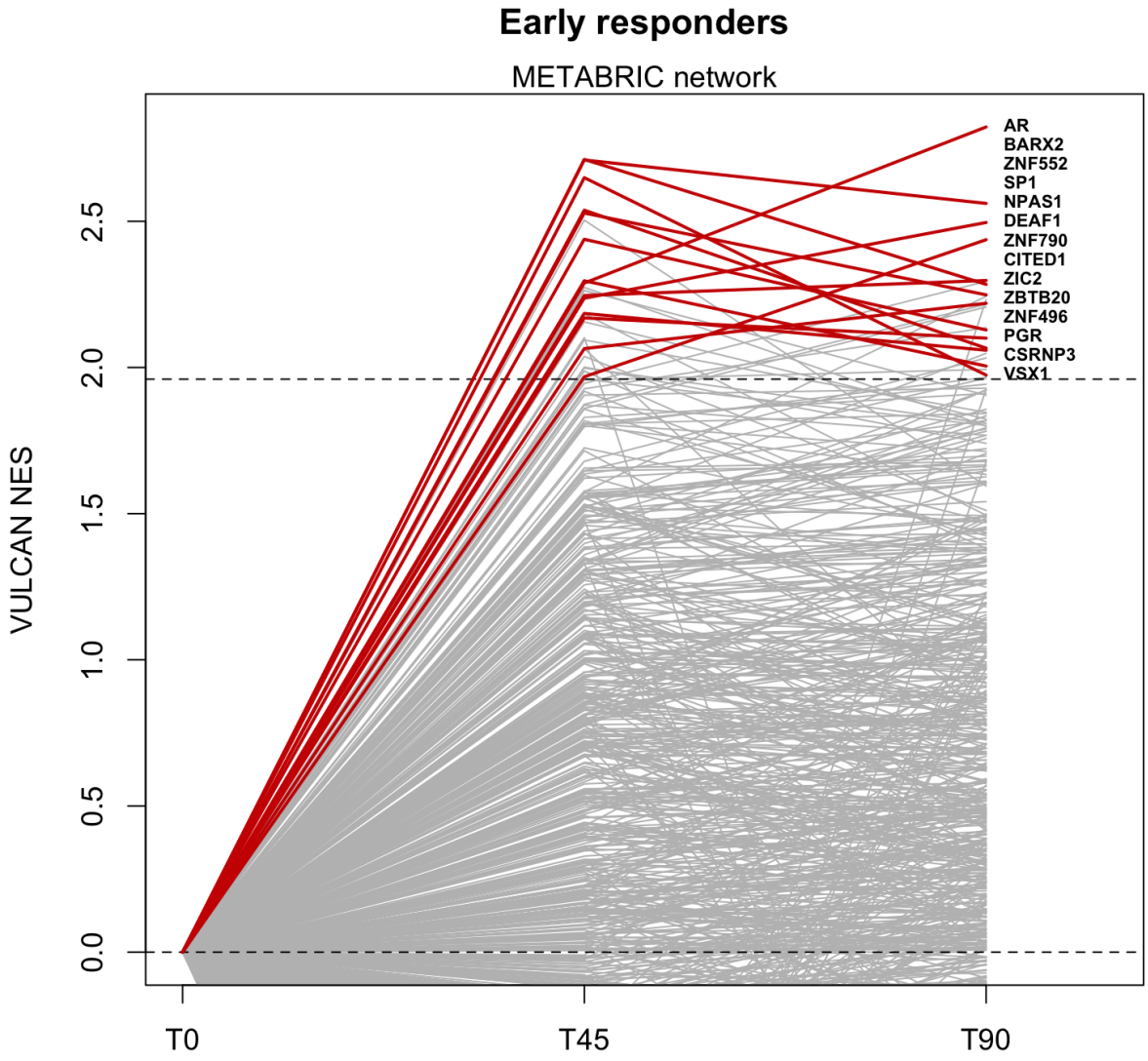


Figure 11: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the METABRIC network, highlighting TFs significantly upregulated at 45 minutes and 90 minutes

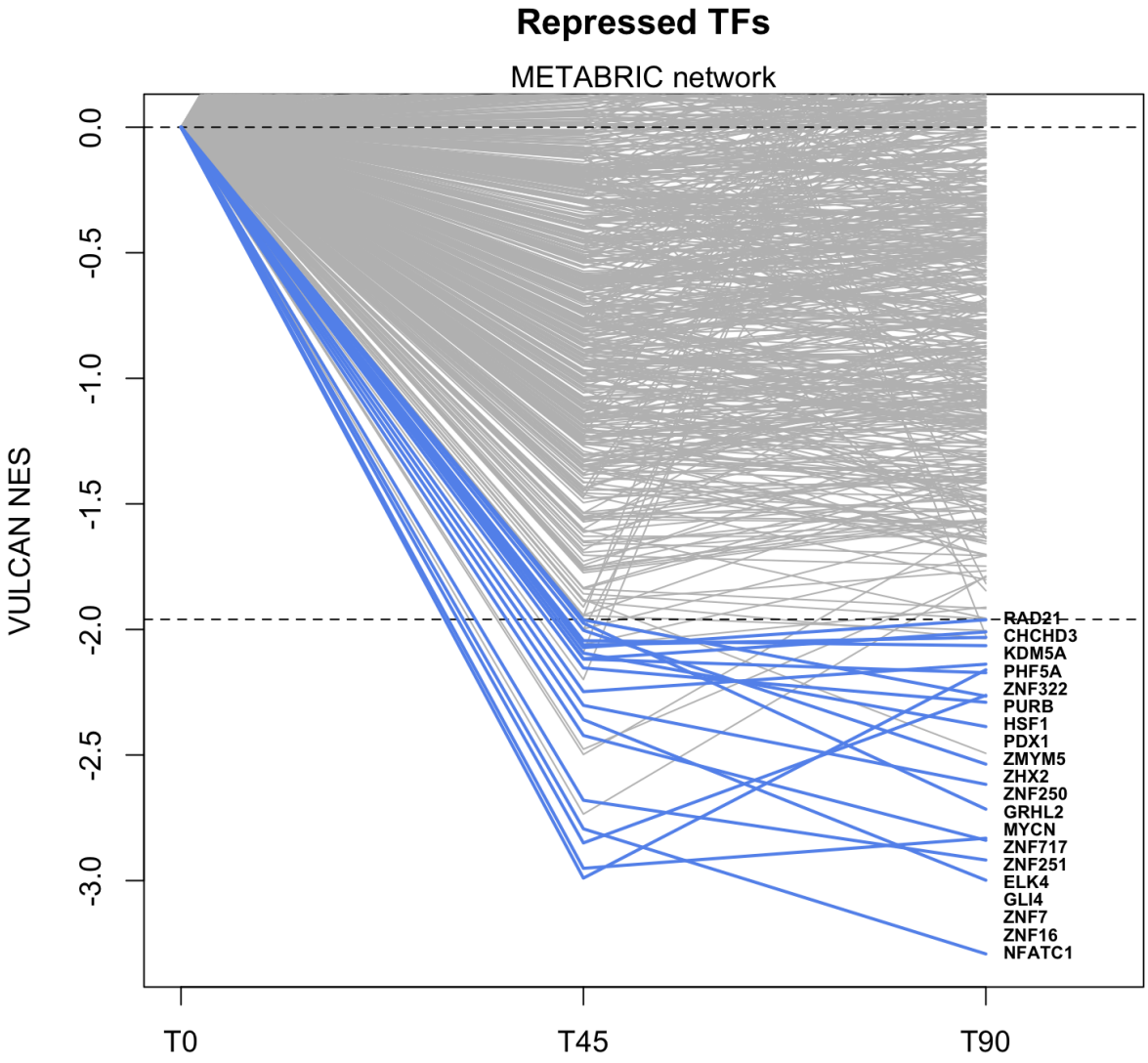


Figure 12: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the METABRIC network, highlighting TFs significantly downregulated at 45 minutes and 90 minutes

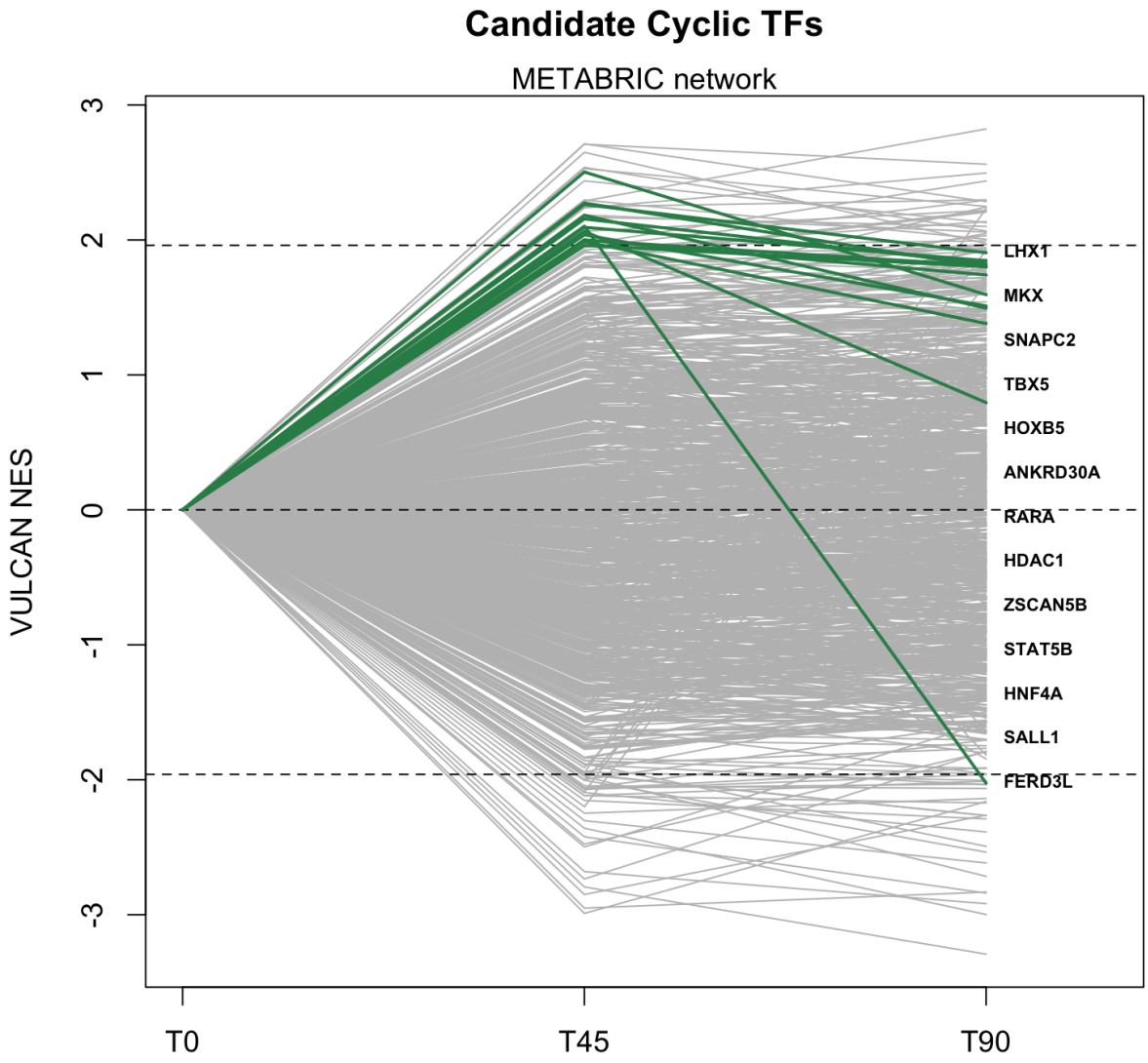


Figure 13: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the METABRIC network, highlighting TFs significantly upregulated at 45 minutes but not at 90 minutes

Finally, a category of TFs appear to be activated but at 90 minutes only. We call this category of TFs **late responders**.

```
threshold<-p2z(0.05)
tfclass<-tfs[metabricmat[,"T45"]<=threshold&metabricmat[,"T45"]>0&metabricmat[,"T90"]>=threshold]
matplot(t(metabricmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Late responders",xlim=
axis(1,at=c(1:3),labels=colnames(metabricmat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(metabricmat[tfclass,]),type="l",col="darkorange",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(metabricmat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),min(oricoords)*1.2,length.out=length(oricoords)),names(oricoords)
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("METABRIC network")
```

3.4.2.2 TCGA results

All the TFs and the four categories of TFs were inferred also using the TCGA network.

```
tcga_90=vobj_tcga_90$mrs[,"NES"]
tcga_45=vobj_tcga_45$mrs[,"NES"]
tfs<-names(tcga_45)
tcgamat<-cbind(rep(0,length(tcga_45)),tcga_45[tfs],tcga_90[tfs])
colnames(tcgamat)<-c("T0","T45","T90")
## All TFs
matplot(t(tcgamat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="All TFs",xlim=c(1,3.3),yli
axis(1,at=c(1:3),labels=colnames(tcgamat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(t(tcgamat["ESR1",])),type="l",col="red",lty=1,lwd=2,add=TRUE)
text(3,tcgamat["ESR1",3],label="ESR1",pos=4,cex=0.6,font=2,col="red3")
mtext("TCGA network")
```

```
threshold<-p2z(0.05)
tfclass<-tfs[tcgamat[,"T45"]>=threshold&tcgamat[,"T90"]>=threshold]
matplot(t(tcgamat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Early responders",xlim=c(
axis(1,at=c(1:3),labels=colnames(tcgamat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(tcgamat[tfclass,]),type="l",col="red3",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(tcgamat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
text(3,newcoords["ESR1"],label="ESR1",pos=4,cex=0.6,font=2,col="red3")
mtext("TCGA network")
```

```
threshold<-p2z(0.05)
tfclass<-tfs[tcgamat[,"T45"]<=-threshold&tcgamat[,"T90"]<=-threshold]
matplot(t(tcgamat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Repressed TFs",xlim=c(1,3
axis(1,at=c(1:3),labels=colnames(tcgamat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(tcgamat[tfclass,]),type="l",col="cornflowerblue",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
```

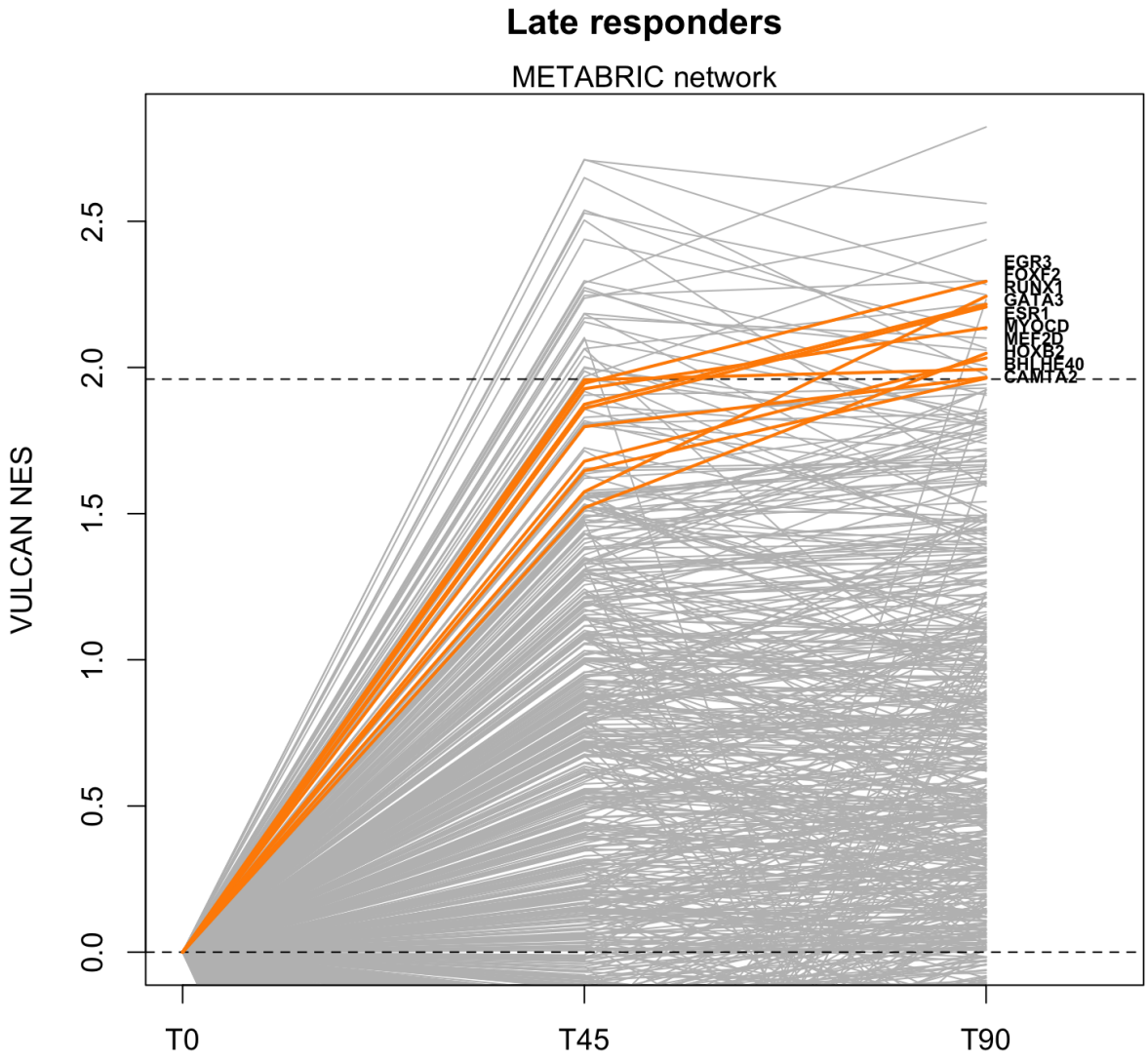


Figure 14: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the METABRIC network, highlighting TFs significantly upregulated at 90 minutes but not at 45 minutes

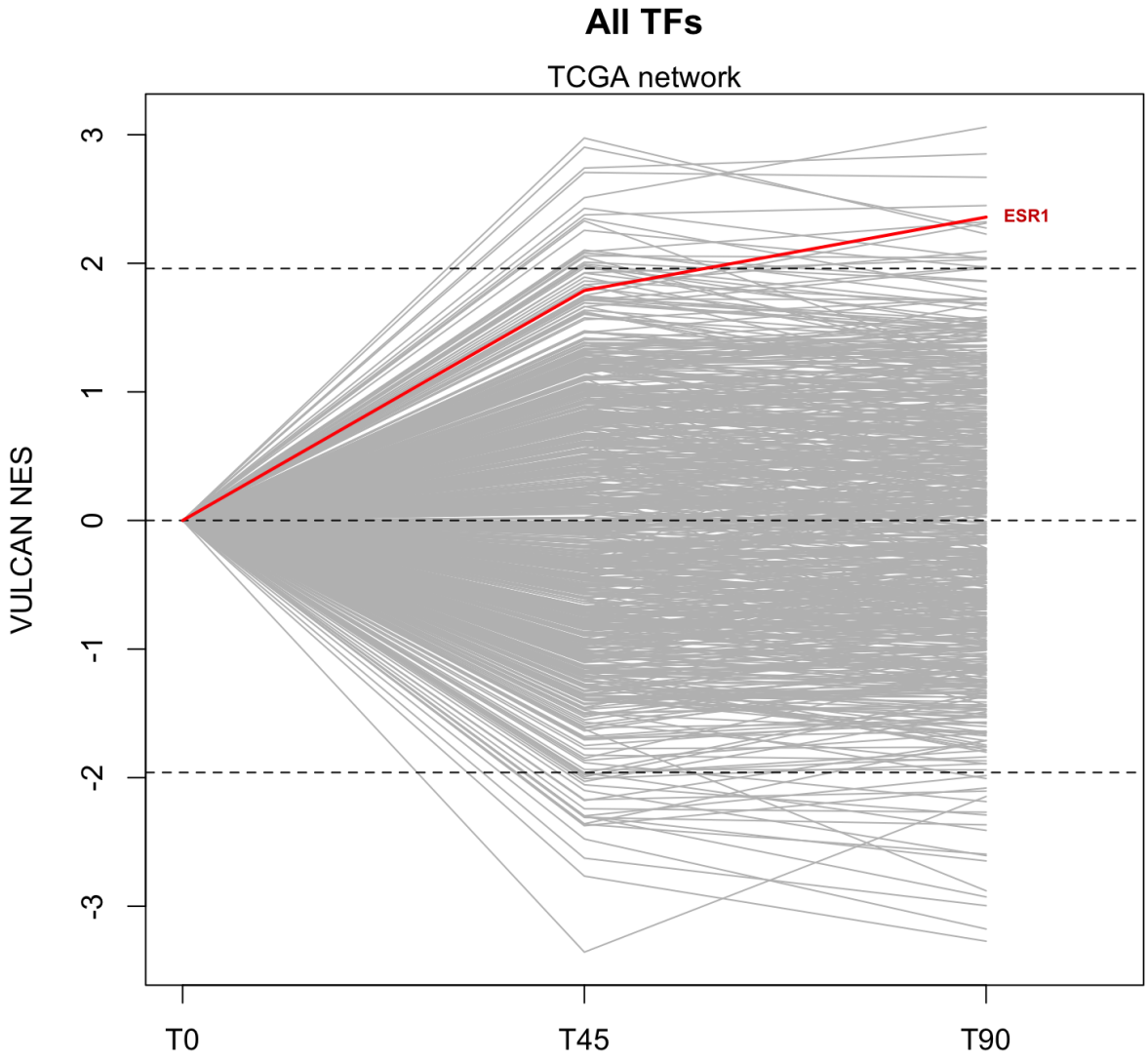


Figure 15: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the TCGA network, highlighting the ESR1 TF as an example

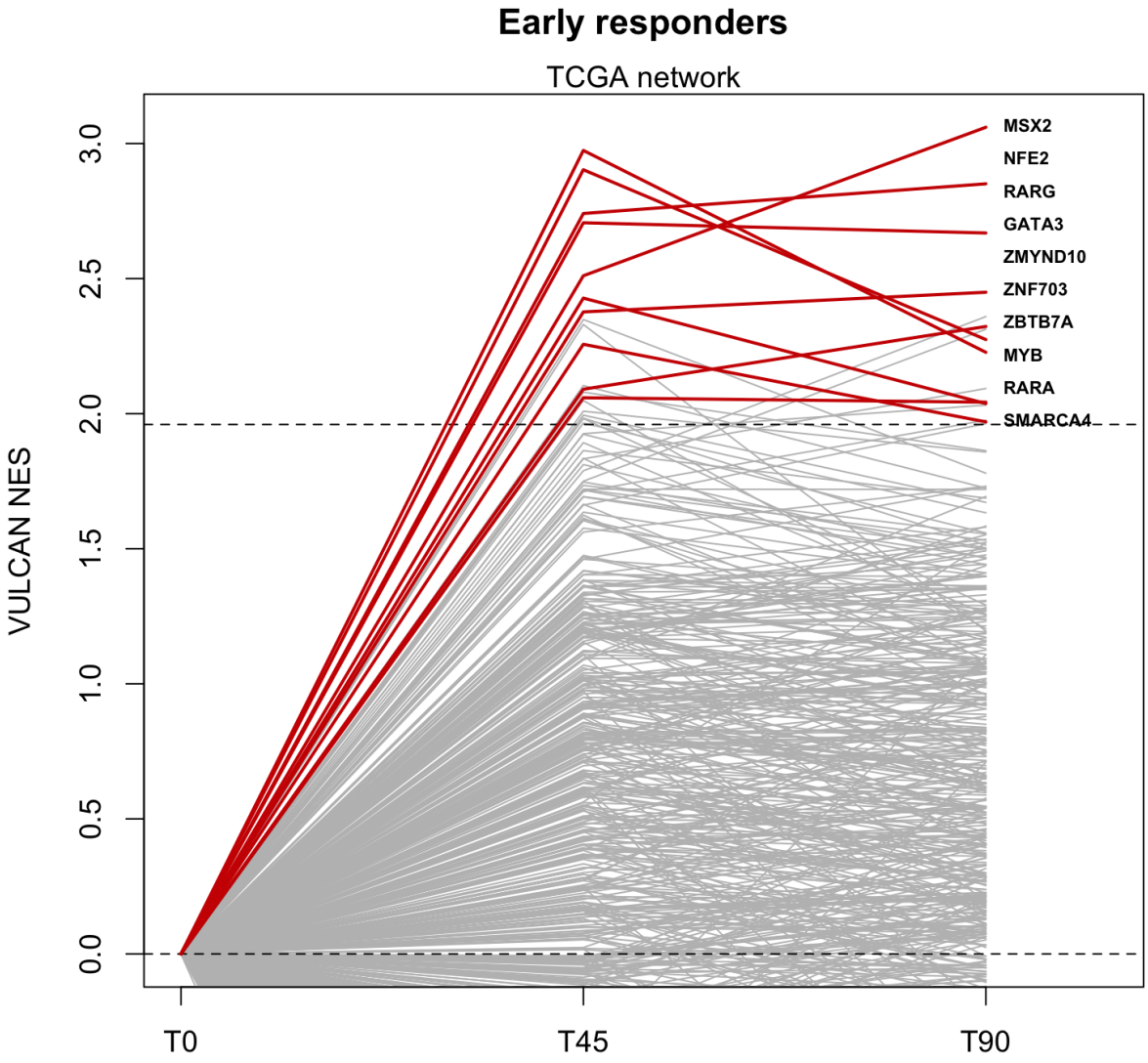


Figure 16: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the TCGA network, highlighting TFs significantly upregulated at 45 minutes and 90 minutes

```
oricoords<-sort(setNames(tcgamat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("TCGA network")
```

```
threshold<-p2z(0.05)
tfclass<-tfs[tcgamat[,"T45"]>=threshold&tcgamat[,"T90"]<=threshold]
matplot(t(tcgamat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Candidate Cyclic TFs",xli
axis(1,at=c(1:3),labels=colnames(tcgamat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(tcgamat[tfclass,]),type="l",col="seagreen",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(tcgamat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),max(oricoords),length.out=length(oricoords)),names(oricoords))
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("TCGA network")
```

```
## Dn-Up TF class
threshold<-p2z(0.05)
tfclass<-tfs[tcgamat[,"T45"]<=threshold&tcgamat[,"T45"]>0&tcgamat[,"T90"]>=threshold]
matplot(t(tcgamat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main="Late responders",xlim=c(1
axis(1,at=c(1:3),labels=colnames(tcgamat))
abline(h=c(0,threshold,-threshold),lty=2)
matplot(t(tcgamat[tfclass,]),type="l",col="darkorange",lty=1,lwd=2,add=TRUE)
# Repel a bit
plabels<-tfclass
oricoords<-sort(setNames(tcgamat[plabels,3],plabels))
newcoords<-setNames(seq(min(oricoords),min(oricoords)*1.2,length.out=length(oricoords)),names(oricoords
text(3,newcoords,label=names(oricoords),pos=4,cex=0.6,font=2)
mtext("TCGA network")
```

3.4.2.3 Comparing TCGA and METABRIC results

The following scatterplots compare the activity inferred by VULCAN using two alternative networks, derived from the TCGA and METABRIC breast cancer datasets using ARACNe-AP (Giorgi et al., 2016) with default parameters.

```
common<-intersect(rownames(tcgamat),rownames(metabricmat))
set.seed(1)
x<-tcgamat[common,"T45"]
y<-metabricmat[common,"T45"]
plot(x,y,xlab="TCGA, 45mins",ylab="METABRIC, 45mins",pch=20,col="grey",xlim=c(min(x)*1.2,max(x)*1.2))
grid()
#toshow<-c("ESR1","GATA3","RARA","HSF1")
toshow<-names(which(rank(rank(-abs(x))+rank(-abs(y)))<=20))
#toshow<-unique(c(toshow,names(sort(rank(-abs(x))[1:10])),names(sort(rank(-abs(y)))[1:10])))
textplot2(x[toshow],y[toshow],words=toshow,new=FALSE,cex=0.8)
lm1<-lm(y~x)
abline(lm1$coef)
pcc<-cor.test(x,y)
mtext(paste0("PCC=",signif(pcc$estimate,2)," (p=",signif(pcc$p.value,2),")"))
```

```
x<-tcgamat[common,"T90"]
y<-metabricmat[common,"T90"]
```

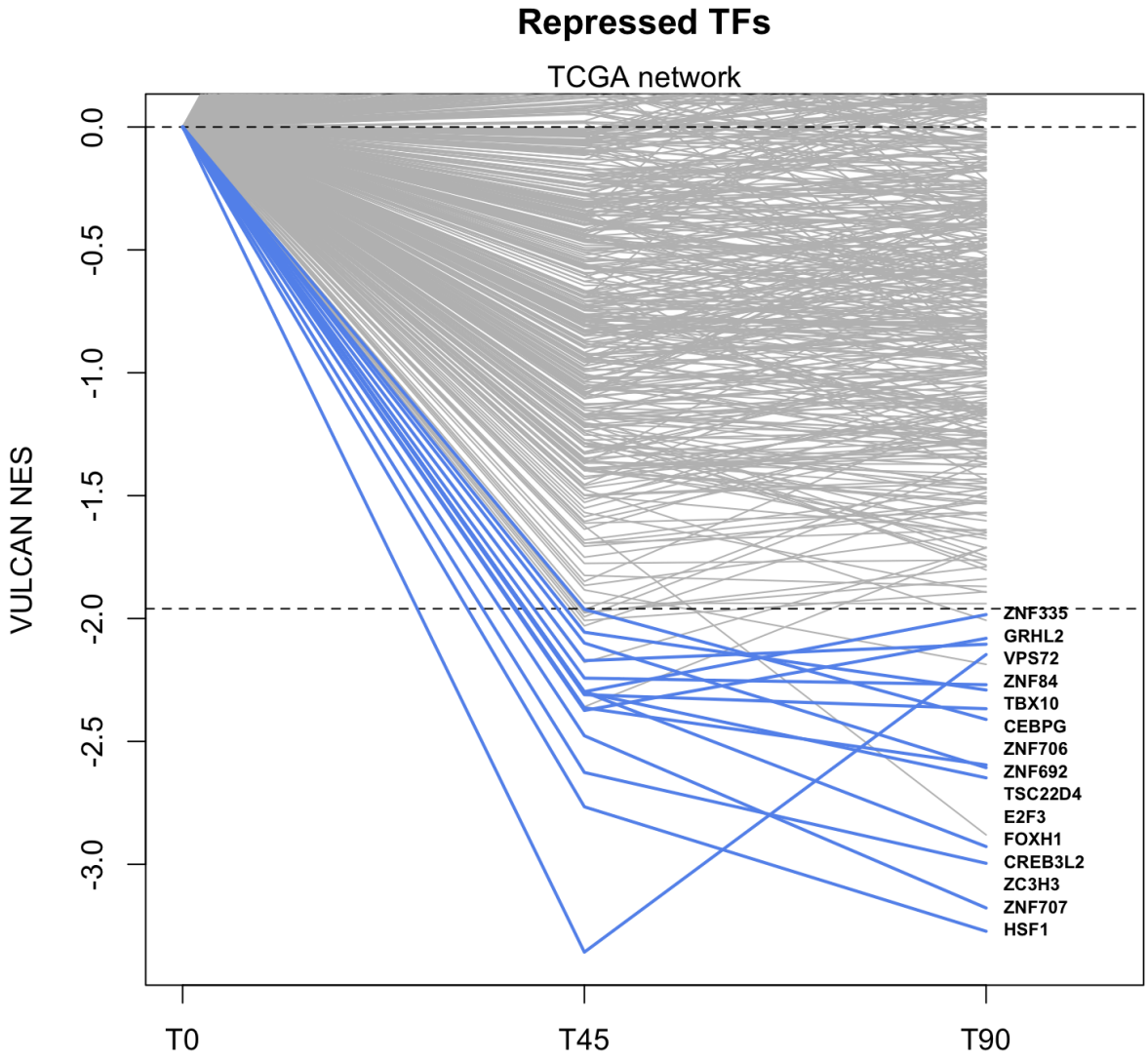


Figure 17: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the TCGA network, highlighting TFs significantly downregulated at 45 minutes and 90 minutes

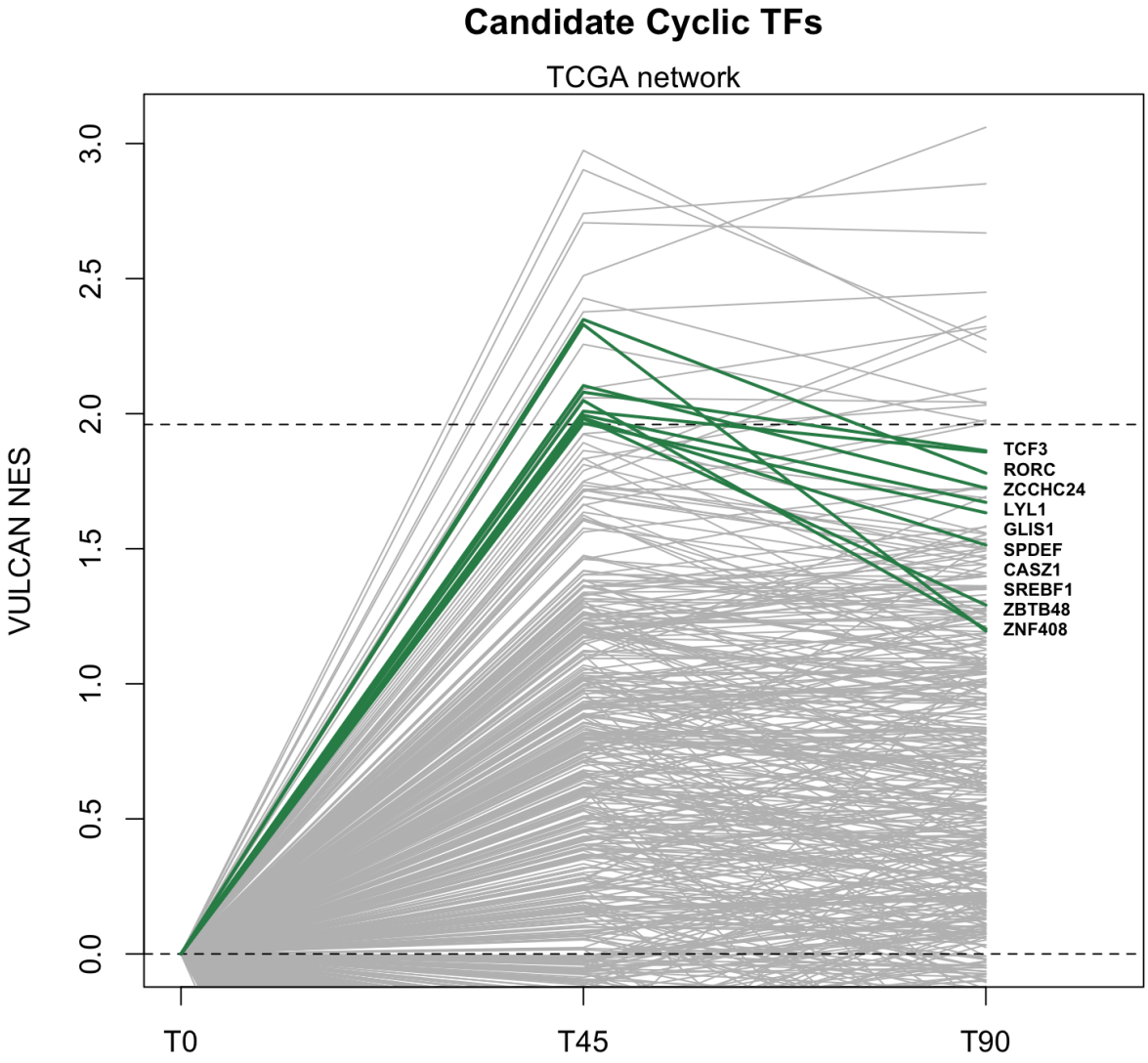


Figure 18: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the TCGA network, highlighting TFs significantly upregulated at 45 minutes but not at 90 minutes

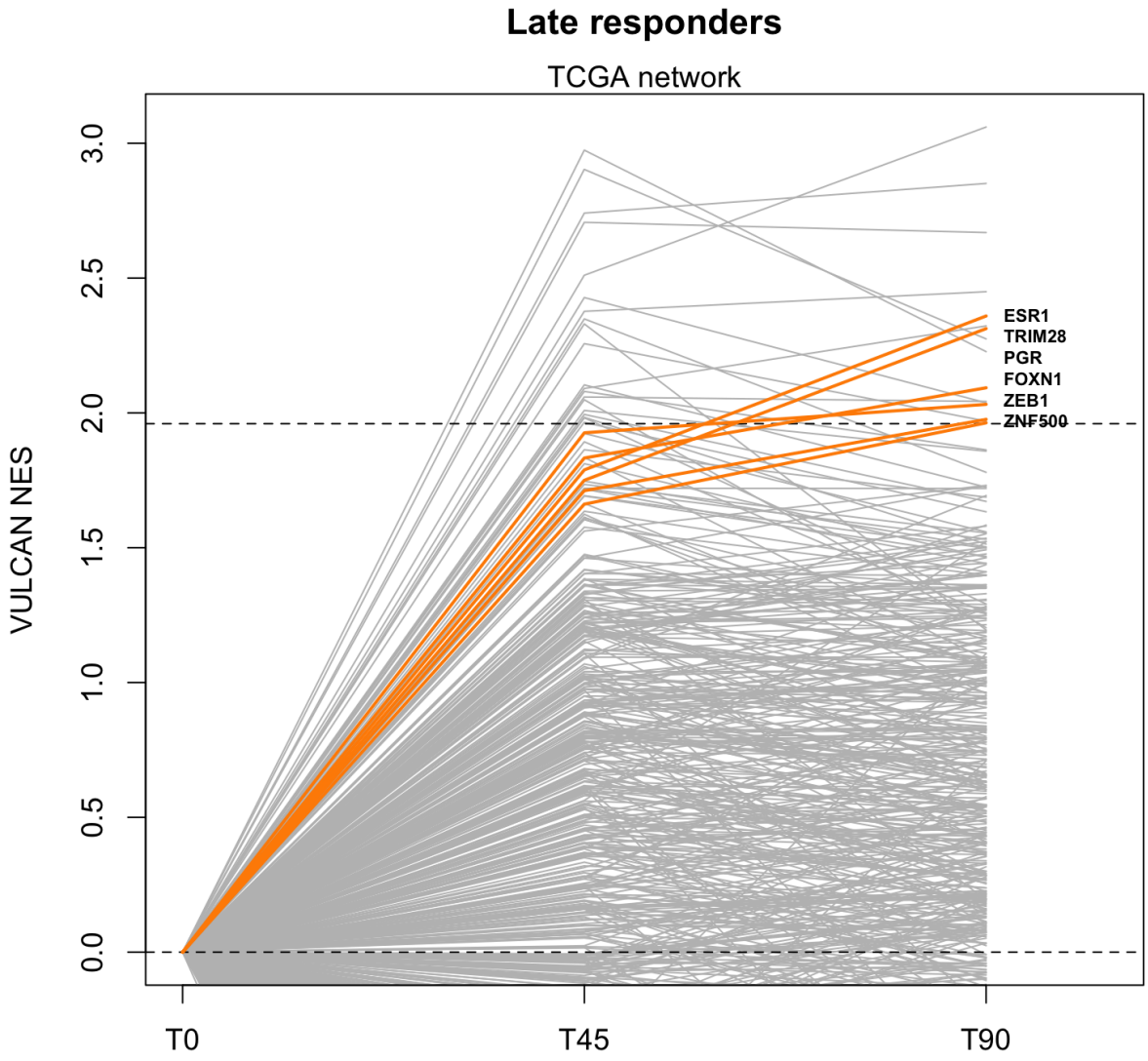


Figure 19: Global TF activity after Estradiol Treatment in MCF7 cells, inferred using the TCGA network, highlighting TFs significantly upregulated at 90 minutes but not at 45 minutes

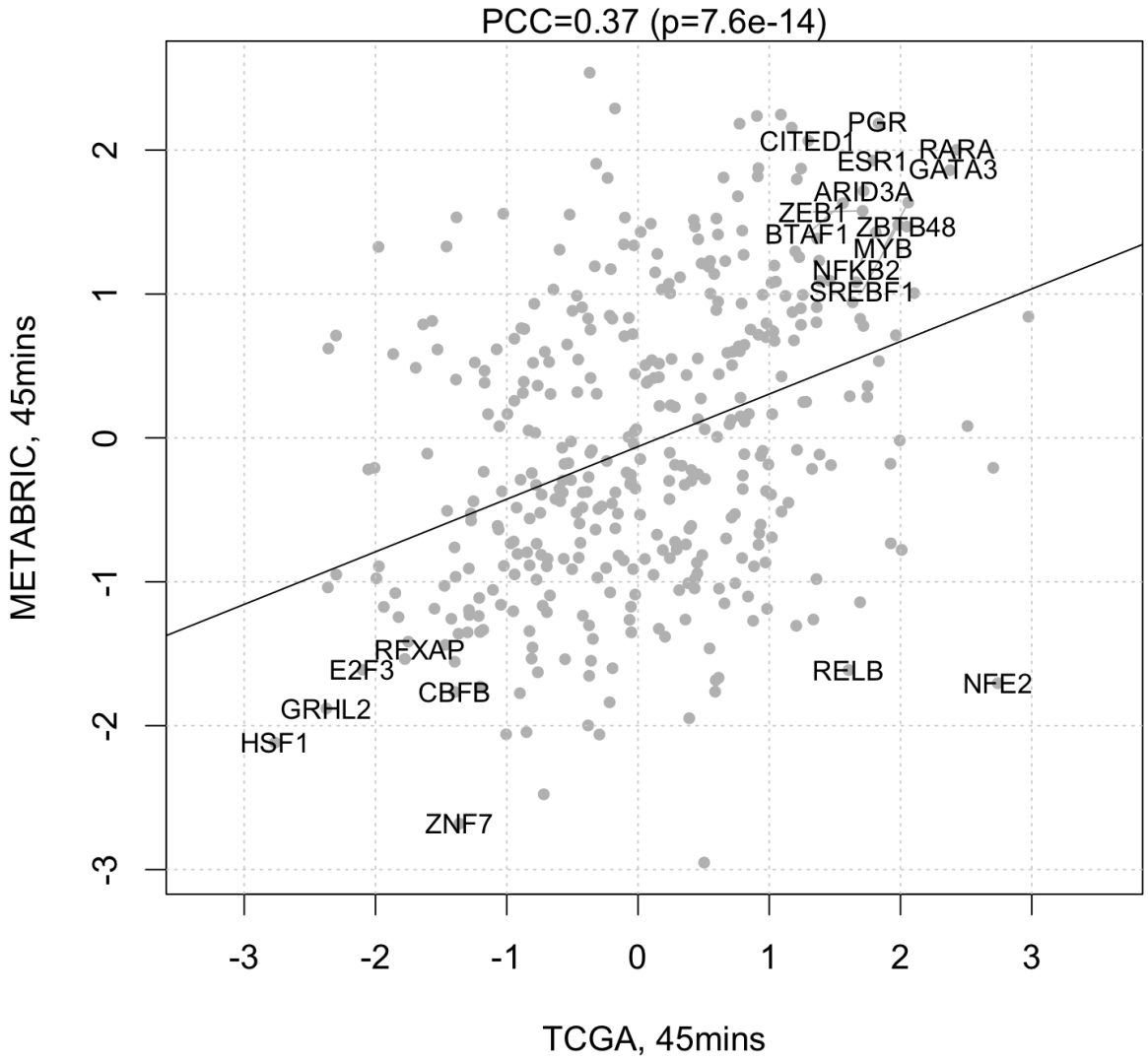


Figure 20: Comparing TF activities inferred by VULCAN using two different breast cancer dataset-derived networks on the ER 45 minutes signature. PCC indicates the Pearson Correlation Coefficient

```
plot(x,y,xlab="TCGA, 90mins",ylab="METABRIC, 90mins",pch=20,col="grey",xlim=c(min(x)*1.2,max(x)*1.2))
grid()
#toshow<-c("ESR1","GATA3","RARA","HSF1")
toshow<-names(which(rank(rank(-abs(x))+rank(-abs(y)))<=20))
#toshow<-unique(c(toshow,names(sort(rank(-abs(x))[1:10])),names(sort(rank(-abs(y)))[1:10])))
textplot2(x[toshow],y[toshow],words=toshow,new=FALSE,cex=0.8)
lm1<-lm(y~x)
abline(lm1$coef)
pcc<-cor.test(x,y)
mtext(paste0("PCC=",signif(pcc$estimate,2)," (p=",signif(pcc$p.value,2),")"))
abline(lm1$coef)
```

3.4.2.4 Testing with a different context network as a negative control

Regulatory networks can be tissue-specific, and so we built a third network using the same parameters as the two breast cancer data-based ones. The third network is built on leukemic samples from the TCGA AML dataset. Our results show a weaker activity for all TFs, which is only weakly correlated to that inferred via the TCGA breast cancer network.

```
negative_90=vobj_negative_90$mrs[,"NES"]
negative_45=vobj_negative_45$mrs[,"NES"]
tfs<-names(negative_45)
negativemat<-cbind(rep(0,length(negative_45)),negative_45[tfs],negative_90[tfs])
colnames(negativemat)<-c("T0","T45","T90")

common<-intersect(rownames(tcgamat),rownames(negativemat))

par(mfrow=c(1,2),oma=c(0,0,2,0))
# Scatterplot, negative vs tcga 45mins
x<-tcgamat[common,"T45"]
y<-negativemat[common,"T45"]
plot(x,y,xlab="TCGA, 45mins",ylab="negative, 45mins",pch=20,col="grey",xlim=c(min(x)*1.2,max(x)*1.2))
grid()
lm1<-lm(y~x)
abline(lm1$coef)
pcc<-cor.test(x,y)
mtext(paste0("PCC=",signif(pcc$estimate,2)," (p=",signif(p.adjust(pcc$p.value,method="bonferroni",n=100

abline(lm1$coef)


# Scatterplot, negative vs tcga 90mins
x<-tcgamat[common,"T90"]
y<-negativemat[common,"T90"]
plot(x,y,xlab="TCGA, 90mins",ylab="negative, 90mins",pch=20,col="grey",xlim=c(min(x)*1.2,max(x)*1.2))
grid()
lm1<-lm(y~x)
abline(lm1$coef)
pcc<-cor.test(x,y)
mtext(paste0("PCC=",signif(pcc$estimate,2)," (p=",signif(p.adjust(pcc$p.value,method="bonferroni",n=100
abline(lm1$coef)

title("Comparison with Negative Network", outer=TRUE)
```

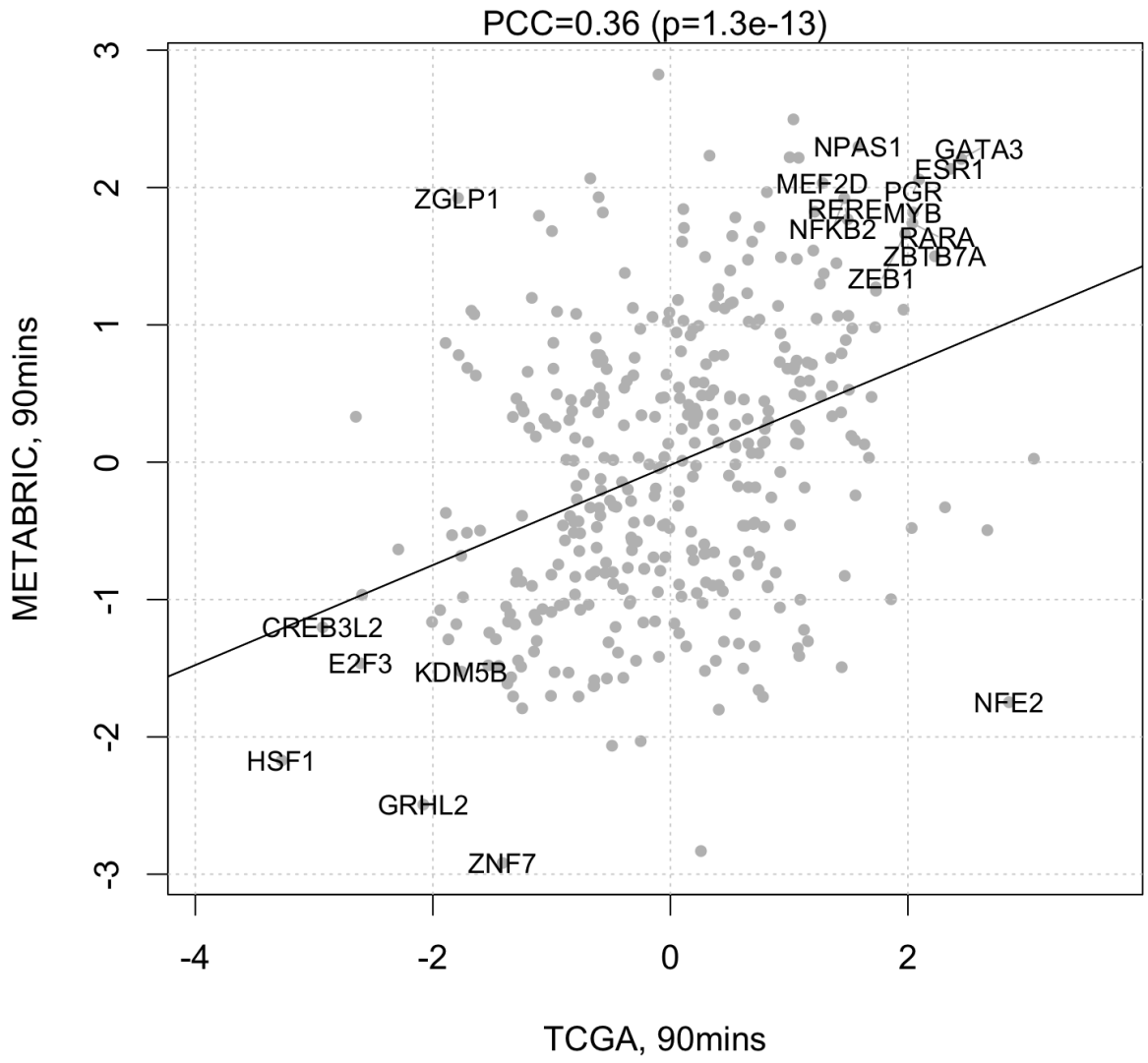


Figure 21: Comparing TF activities inferred by VULCAN using two different breast cancer dataset-derived networks on the ER 90 minutes signature. PCC indicates the Pearson Correlation Coefficient

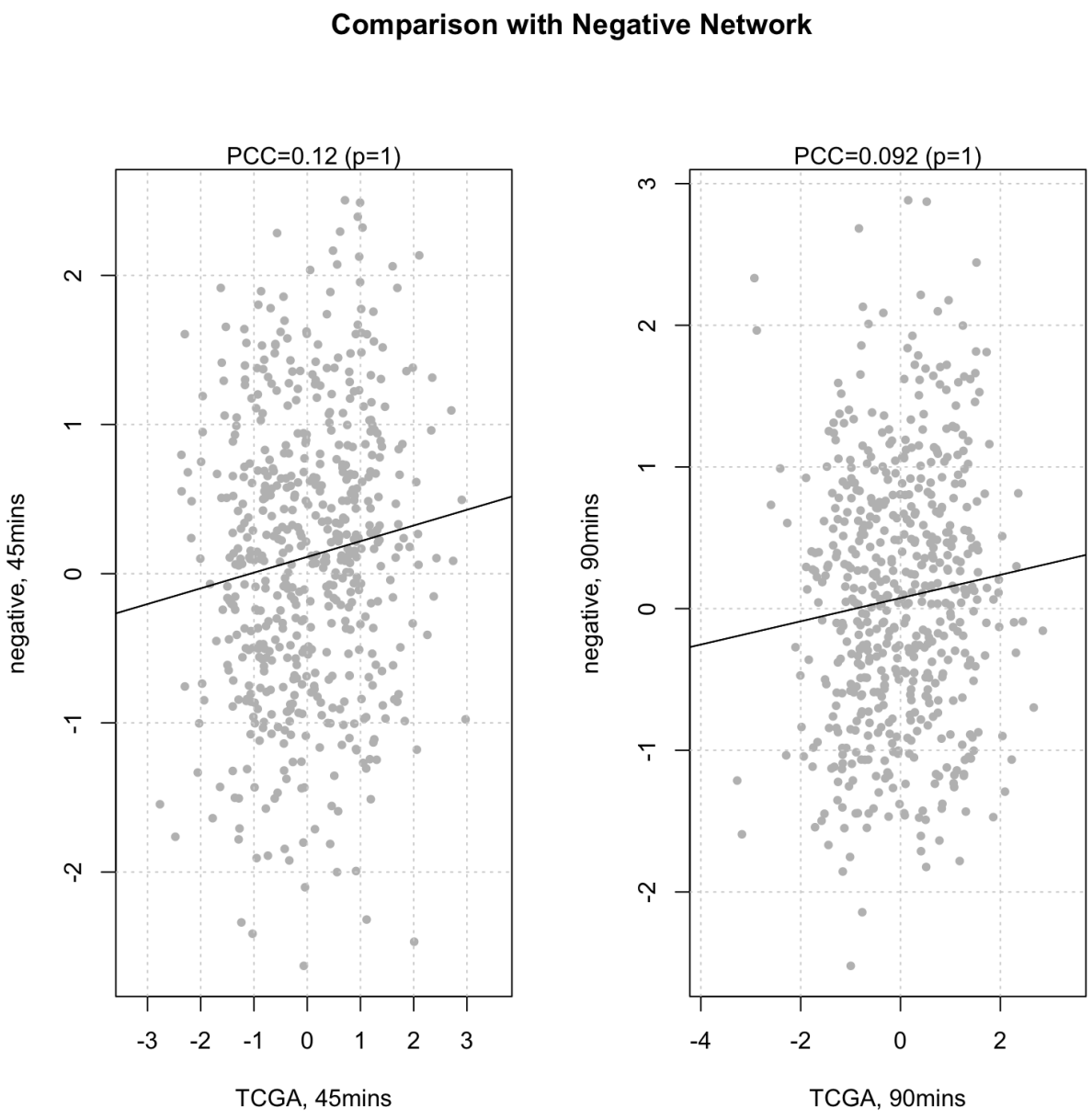


Figure 22: Comparison between activities inferred through a breast cancer TCGA dataset and the AML dataset. PCC indicates the Pearson Correlation Coefficient

3.5 Pathway enrichment analysis

In the previous paragraphs we generated signatures of differential binding centered around the peaks extracted by VULCAN using the *vulcan.annotate* function. We apply two methods that assess the enrichment of gene pathways over continuous signatures: GSEA (Subramaniam et al., 2004) and aREA (Alvarez et al., 2016). First, we will load the VULCAN object containing our dataset:

```
# Load imported vulcan object
load("results/001_vobj.rda")
```

Then, we need gene pathways. A manually annotated and very exhaustive list is available from the MsiGDB database by the Broad Institute

```
# Load MsigDB
load("msigdb/MSigDB_v5.0_human.rda")
pathways<-msigDBentrez$c2.all.v5.0.entrez.gmt
```

First, we compare aREA and GSEA. Despite being similar in purpose, the two methods differ in their core algorithm and implementations. Notably, aREA processes the entire analysis in one go, by virtue of matrix multiplications. We expect aREA to be faster, while using more RAM than GSEA. Both GSEA and aREA are implemented in the VULCAN package. Here we calculate GSEA on the 90' vs 00' signature:

```
fname<-"results/003_pathwayComparison_GSEA_90.rda"
if(!file.exists(fname)){
  start<-Sys.time()
  results_gsea_90<-vulcan.pathways(vobj,pathways,contrast=c("t90","t0"),method="GSEA",np=1000)
  end<-Sys.time()
  time_gsea<-end-start
  save(results_gsea_90,time_gsea,file=fname)
} else {
  load(fname)
}
```

Then, we calculate aREA on the same 90' vs 00' signature:

```
fname<-"results/003_pathwayComparison_REA_90.rda"
if(!file.exists(fname)){
  start<-Sys.time()
  results_rea_90<-vulcan.pathways(vobj,pathways,contrast=c("t90","t0"),method="REA")[,1]
  end<-Sys.time()
  time_rea<-end-start
  save(results_rea_90,time_rea,file=fname)
} else {
  load(fname)
}
```

We then compare GSEA and aREA:

```
plot(results_rea_90,results_gsea_90,xlab="aREA enrichment score",ylab="GSEA enrichment score",pch=20,ma
     ylim=c(-max(abs(results_gsea_90)),max(abs(results_gsea_90))),col="grey")
th<-p2z(0.1)
abline(h=c(th,-th),v=c(th,-th),lty=2)
lm1<-lm(results_gsea_90~results_rea_90)
abline(lm1$coef,lwd=2)
pcc<-cor(results_rea_90,results_gsea_90,method="p")
mtext(paste0("PCC=",signif(pcc,3),", run on ",length(pathways)," pathways"))
# Estrogen
keywords<-c("ESTROGEN","ESTRADIOL","ESR1")
```

```
keypaths<-c()
for(keyword in keywords){
  keypaths<-c(keypaths,grep(keyword,names(pathways),value=TRUE))
}
keypaths<-unique(keypaths)
# Breast Cancer
keywords<-c("BREAST_CANCER")
keypaths2<-c()
for(keyword in keywords){
  keypaths2<-c(keypaths2,grep(keyword,names(pathways),value=TRUE))
}
keypaths2<-unique(keypaths2)

points(results_rea_90[keypaths],results_gsea_90[keypaths],pch=16,col="red3")
points(results_rea_90[keypaths2],results_gsea_90[keypaths2],pch=16,col="navy")
legend("topleft",pch=16,col=c("red3","navy"),cex=1,legend=c("Estrogen-Related Pathways","Breast Cancer
legend("bottomright",
       legend=c(
         paste0("REA: ",signif(time_rea,2)," ",units(time_rea)),
         paste0("GSEA: ",signif(time_gsea,2)," ",units(time_gsea))
       ),
       title="Runtime"
)
```

aREA appears to be faster. Therefore, we will use it to calculate the 45' signature as well:

```
fname<-"results/003_pathwayComparison_REA_45.rda"
if(!file.exists(fname)){
  start<-Sys.time()
  results_rea_45<-vulcan.pathways(vobj,pathways,contrast=c("t45","t0"),method="REA")[,1]
  end<-Sys.time()
  time_rea<-end-start
  save(results_rea_45,time_rea,file=fname)
} else {
  load(fname)
}
```

The results of aREA in both 90'vs00' and 45'vs00' signatures show the presence of known Estrogen-related pahtways:

```
topdn<-sort(results_rea_90)[1:20]
topup<-sort(results_rea_90,dec=TRUE)[1:20]

topdn<-cbind(topdn,z2p(topdn))
topup<-cbind(topup,z2p(topup))
colnames(topdn)<-colnames(topup)<-c("NES, 90' vs 0'","pvalue")

rownames(topdn)<-tolower(rownames(topdn))
rownames(topup)<-tolower(rownames(topup))
rownames(topdn)<-gsub("_"," ",rownames(topdn))
rownames(topup)<-gsub("_"," ",rownames(topup))
topdn[,1]<-signif(topdn[,1],3)
topup[,1]<-signif(topup[,1],3)
topdn[,2]<-signif(topdn[,2],2)
topup[,2]<-signif(topup[,2],2)
```

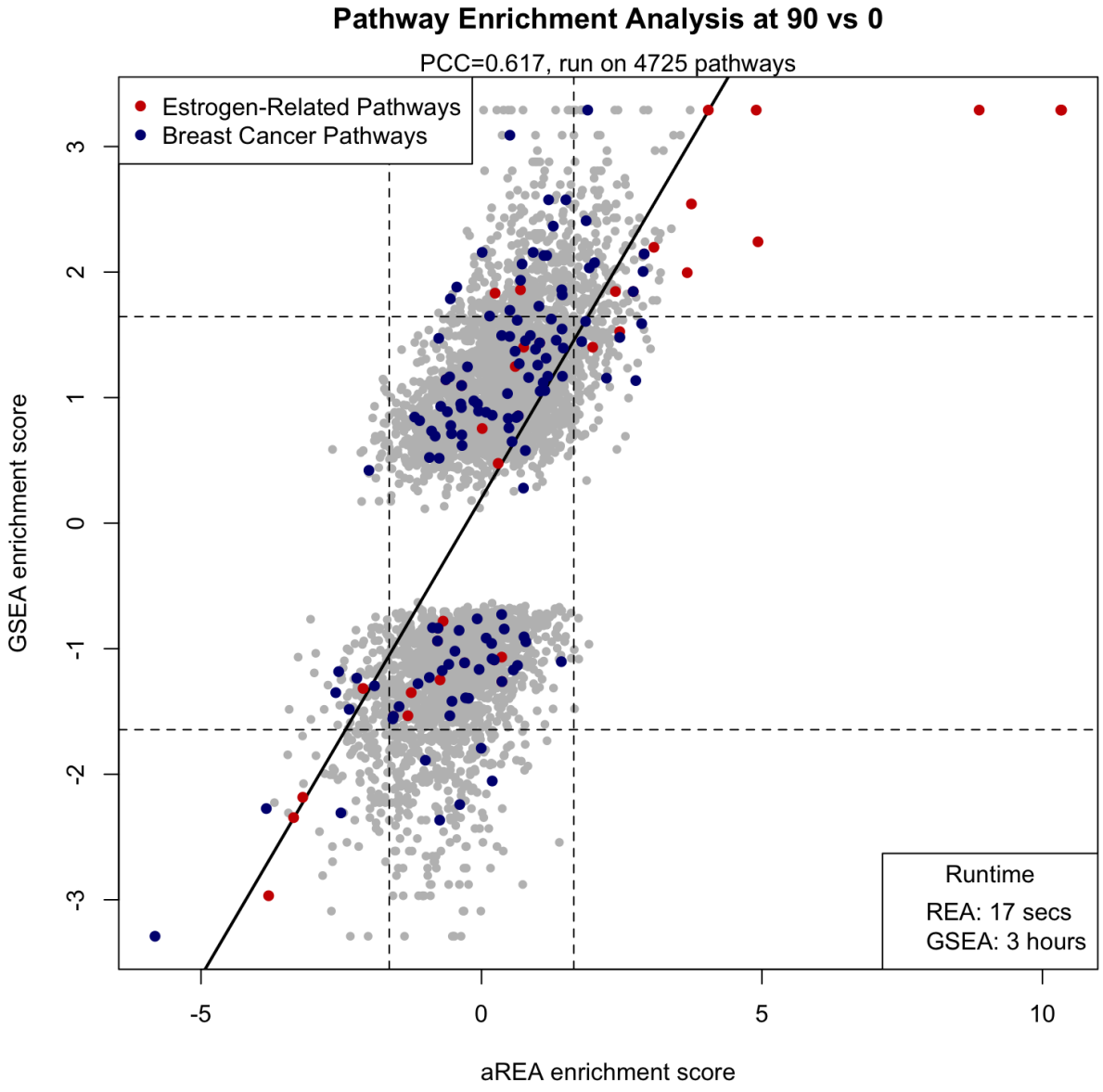


Figure 23: Comparison of GSEA and aREA on a differential binding signature

```
firstup <- function(x) {
  substr(x, 1, 1) <- toupper(substr(x, 1, 1))
  x
}
rownames(topdn)<-firstup(rownames(topdn))
rownames(topup)<-firstup(rownames(topup))
```

```
# Plot your table with table Grob in the library(gridExtra)
grid.newpage()
grid.table(topup)
```

```
grid.newpage()
grid.table(topdn)
```

```
topdn<-sort(results_rea_45)[1:20]
topup<-sort(results_rea_45,dec=TRUE)[1:20]

topdn<-cbind(topdn,z2p(topdn))
topup<-cbind(topup,z2p(topup))
colnames(topdn)<-colnames(topup)<-c("NES, 45' vs 0'","pvalue")

rownames(topdn)<-tolower(rownames(topdn))
rownames(topup)<-tolower(rownames(topup))
rownames(topdn)<-gsub("_"," ",rownames(topdn))
rownames(topup)<-gsub("_"," ",rownames(topup))
topdn[,1]<-signif(topdn[,1],3)
topup[,1]<-signif(topup[,1],3)
topdn[,2]<-signif(topdn[,2],2)
topup[,2]<-signif(topup[,2],2)

firstup <- function(x) {
  substr(x, 1, 1) <- toupper(substr(x, 1, 1))
  x
}
rownames(topdn)<-firstup(rownames(topdn))
rownames(topup)<-firstup(rownames(topup))
```

```
# Plot your table with table Grob in the library(gridExtra)
grid.newpage()
grid.table(topup)
```

```
grid.newpage()
grid.table(topdn)
```

3.6 The GRHL2 Transcription Factor

Our analysis shows that the genes repressed by the Transcription Factor GRHL2 are occupied by the ER complex, using both TCGA-derived and METABRIC-derived regulatory models.

	NES, 90' vs 0'	pvalue
Bhat esr1 targets not via akt1 up	10.3	4.6e-25
Bhat esr1 targets via akt1 up	10.3	5.3e-25
Dutertre estradiol response 6hr up	8.87	7.1e-19
Frasor response to estradiol up	4.93	8.3e-07
Dutertre estradiol response 24hr up	4.9	9.7e-07
Stein esr1 targets	4.04	5.3e-05
Stein esrra targets responsive to estrogen dn	3.74	0.00018
Creighton endocrine therapy resistance 1	3.72	2e-04
Stossi response to estradiol	3.67	0.00024
Zwang egf interval dn	3.55	0.00039
Creighton endocrine therapy resistance 4	3.42	0.00062
Pedrioli mir31 targets up	3.38	0.00071
Massarweh tamoxifen resistance dn	3.24	0.0012
Lein pons markers	3.19	0.0014
Reactome hs gag degradation	3.19	0.0014
Jiang tip30 targets dn	3.17	0.0015
Kegg glycerophospholipid metabolism	3.16	0.0016
Geserick tert targets dn	3.13	0.0018
Gross hypoxia via hif1a dn	3.09	0.002
Valk aml cluster 6	3.09	0.002

Figure 24: Table S2. aREA results: upregulated MsigDBpathways at 90mins

	NES, 90' vs 0'	pvalue
Nikolsky breast cancer 8q23 q24 amplicon	-5.82	5.9e-09
Nikolsky breast cancer 1q21 amplicon	-3.84	0.00012
Dutertre estradiol response 24hr dn	-3.79	0.00015
Streicher lsm1 targets up	-3.69	0.00022
Kenny ctnnb1 targets dn	-3.45	0.00055
Reactome activation of chaperone genes by xbp1s	-3.44	0.00058
Verrecchia response to tgfb1 c4	-3.43	0.00061
Frasor response to estradiol dn	-3.35	0.00081
Pid myc repress pathway	-3.27	0.0011
Sesto response to uv c4	-3.19	0.0014
Dutertre estradiol response 6hr dn	-3.19	0.0014
Reactome degradation of the extracellular matrix	-3.17	0.0015
Lamb ccnd1 targets	-3.05	0.0023
Guo targets of irs1 and irs2	-3.04	0.0024
Vakabayashi adipogenesis pparg rxra bound with h4k20me1 mark	-2.98	0.0029
Woo liver cancer recurrence up	-2.96	0.0031
Pid myc pathway	-2.95	0.0032
Hu angiogenesis dn	-2.89	0.0039
Reactome unfolded protein response	-2.83	0.0046
Yamashita liver cancer with epcam dn	-2.8	0.005

Figure 25: Table S3. aREA results: downregulated MsigDBpathways at 90mins

	NES, 45' vs 0'	pvalue
Bhat esr1 targets via akt1 up	10.8	3e-27
Bhat esr1 targets not via akt1 up	10.4	2.8e-25
Dutertre estradiol response 6hr up	8.1	5.5e-16
Frasor response to estradiol up	4.54	5.6e-06
Stein esrra targets responsive to estrogen dn	4.21	2.6e-05
Dutertre estradiol response 24hr up	4.15	3.4e-05
Stein esr1 targets	4.03	5.6e-05
Vantveer breast cancer esr1 up	3.84	0.00012
Geserick tert targets dn	3.69	0.00022
Pedrioli mir31 targets up	3.61	3e-04
Reactome glycerophospholipid biosynthesis	3.39	0.00069
Zwang egf interval dn	3.38	0.00072
Naba ecm affiliated	3.31	0.00093
Stossi response to estradiol	3.29	0.00099
Lien breast carcinoma metaplastic vs ductal dn	3.25	0.0011
Kegg glycerophospholipid metabolism	3.25	0.0012
Creighton endocrine therapy resistance 1	3.24	0.0012
Reactome hs gag degradation	3.21	0.0013
Valk aml cluster 6	3.2	0.0014
Massarweh tamoxifen resistance dn	3.1	0.0019

Figure 26: Table S5. aREA results: upregulated MsigDBpathways at 45mins

	NES, 45' vs 0'	pvalue
Nikolsky breast cancer 8q23 q24 amplicon	-5.39	7.2e-08
Dutertre estradiol response 24hr dn	-3.97	7.2e-05
Kenny ctnnb1 targets dn	-3.89	1e-04
Nikolsky breast cancer 1q21 amplicon	-3.8	0.00014
Streicher lsm1 targets up	-3.7	0.00021
Wakabayashi adipogenesis pparg rxra bound 8d	-3.45	0.00055
Boyault liver cancer subclass g23 up	-3.16	0.0016
Reactome activation of chaperone genes by xbp1s	-3.13	0.0017
Dutertre estradiol response 6hr dn	-3.04	0.0023
Sesto response to uv c4	-3.01	0.0026
Shin b cell lymphoma cluster 2	-2.96	0.003
Yamashita liver cancer with epcam dn	-2.91	0.0037
Ichiba graft versus host disease 35d up	-2.82	0.0048
Chen hoxa5 targets 9hr up	-2.77	0.0057
Kenny ctnnb1 targets up	-2.74	0.0061
Kegg sulfur metabolism	-2.72	0.0065
Laiho colorectal cancer serrated up	-2.72	0.0066
Reactome unfolded protein response	-2.68	0.0074
Heidenblad amplicon 8q24 up	-2.67	0.0077
Nikolsky breast cancer 20q12 q13 amplicon	-2.65	0.008

Figure 27: Table S6. aREA results: downregulated MsigDBpathways at 45mins

3.6.1 Network similarity between ESR1 and GRHL2

The following analysis highlights, with a Venn Diagram for both the METABRIC and TCGA ARACNe-derived regulatory models, the overlap between the ESR1 (Estrogen Receptor) and GRHL2 networks

```
# Load networks
load("networks/brca-tf-regulon.rda")
tcga_regulon<-regul
rm(regul)

load("networks/metabric-regulon-tfs.rda")
metabric_regulon<-regulon
rm(regulon)


# Select TFs
tf1<-"ESR1"
tf2<-"GRHL2"


par(mfrow=c(1,2))

# Venn diagram in TCGA
reg1<-tcga_regulon[[list_symbol2eg[[tf1]]]]
reg2<-tcga_regulon[[list_symbol2eg[[tf2]]]]
targets1<-names(reg1$tfmode)
targets2<-names(reg2$tfmode)
vennlist<-list()
vennlist[[tf1]]<-targets1
vennlist[[tf2]]<-targets2
venn(vennlist)
title("Network overlap in TCGA")


# Venn diagram in METABRIC
reg1<-metabric_regulon[[list_symbol2eg[[tf1]]]]
reg2<-metabric_regulon[[list_symbol2eg[[tf2]]]]
targets1<-names(reg1$tfmode)
targets2<-names(reg2$tfmode)
vennlist<-list()
vennlist[[tf1]]<-targets1
vennlist[[tf2]]<-targets2
venn(vennlist)
title("Network overlap in METABRIC")
```

```
par<-par.backup
```

3.6.2 Correlation between ESR1 and GRHL2 expression in breast cancer datasets

The following analysis assesses the inverse correlation between GRHL2 and ESR1 expressions in TCGA and METABRIC, and highlights the phenomenon for specific tumor subtypes, according to the PAM50 nomenclature (Perou et al., 2000)

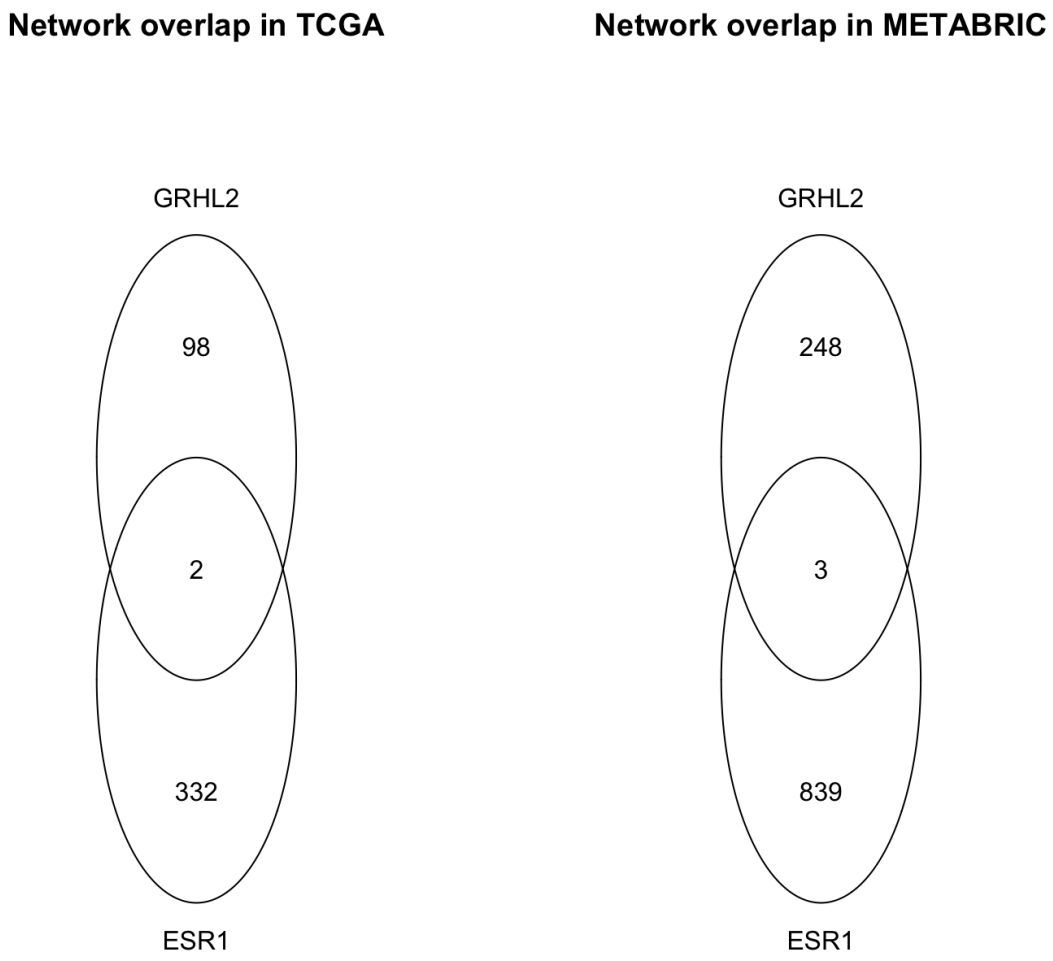


Figure 28: Overlap between ESR1 and GRH2 networks in TCGA (left) and METABRIC (right) networks

```r
# TCGA Color annotation
load("data/brca-expmat.rda")
load("data/brca-subtypes.rda")
expmat<-expmat[,names(subtypes)]
colors<-setNames(rep("black",length(subtypes)),subtypes)
colors[subtypes=="Basal"]<-"#FF0000AA"
colors[subtypes=="LumA"]<-"#00FFFFAA"
colors[subtypes=="LumB"]<-"#0000FFAA"
colors[subtypes=="Her2"]<-"#FFFF00AA"
colors[subtypes=="Tumor, Normal-Like"]<-"#00FF00AA"
colors[subtypes=="Normal, Tumor-Like"]<-"#00FF00AA"
colors[subtypes=="Normal"]<-"#00FF00AA"
colors_tcga<-colors
tcga_expmat<-expmat

# METABRIC Color annotation
load("data/metabric-expmat.rda")
load("data/metabric-subtypes.rda")
expmat<-expmat[,names(subtypes)]
colors<-setNames(rep("black",length(subtypes)),subtypes)
colors[subtypes=="Basal"]<-"#FF0000AA"
colors[subtypes=="LumA"]<-"#00FFFFAA"
colors[subtypes=="LumB"]<-"#0000FFAA"
colors[subtypes=="Her2"]<-"#FFFF00AA"
colors[subtypes=="Tumor, Normal-Like"]<-"#00FF00AA"
colors[subtypes=="Normal, Tumor-Like"]<-"#00FF00AA"
colors[subtypes=="Normal"]<-"#00FF00AA"
colors_metabric<-colors
metabric_expmat<-expmat

x<-tcga_expmat[list_symbol2eg[[tf1]],]
y<-tcga_expmat[list_symbol2eg[[tf2]],]
plot(x,y,xlab=tf1,ylab=tf2,pch=20,main="Correlation between GRHL2 and ESR1 expression in TCGA",col=colo
pcc<-cor(x,y)
mtext(paste0("PCC=",signif(pcc,3)))
grid()
legend("bottomright",col=c("red","cyan","blue","yellow","green"),legend=c("Basal","LumA","LumB","Her2",
```

```r
x<-metabric_expmat[list_symbol2eg[[tf1]],]
y<-metabric_expmat[list_symbol2eg[[tf2]],]
plot(x,y,xlab=tf1,ylab=tf2,pch=20,main="Correlation between GRHL2 and ESR1 expression in METABRIC",col=
pcc<-cor(x,y)
mtext(paste0("PCC=",signif(pcc,3)))
grid()
legend("bottomright",col=c("red","cyan","blue","yellow","green"),legend=c("Basal","LumA","LumB","Her2",
```

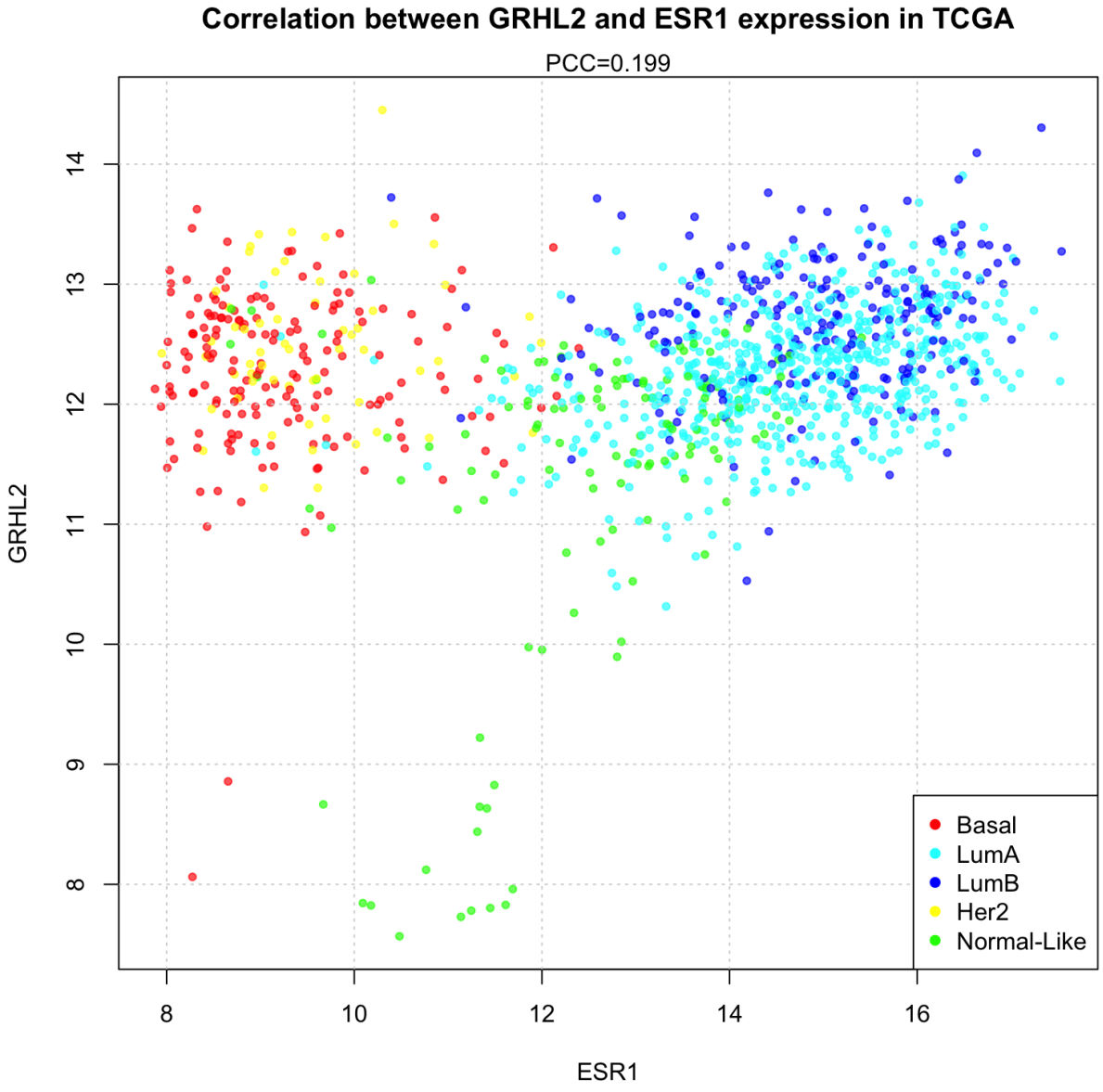
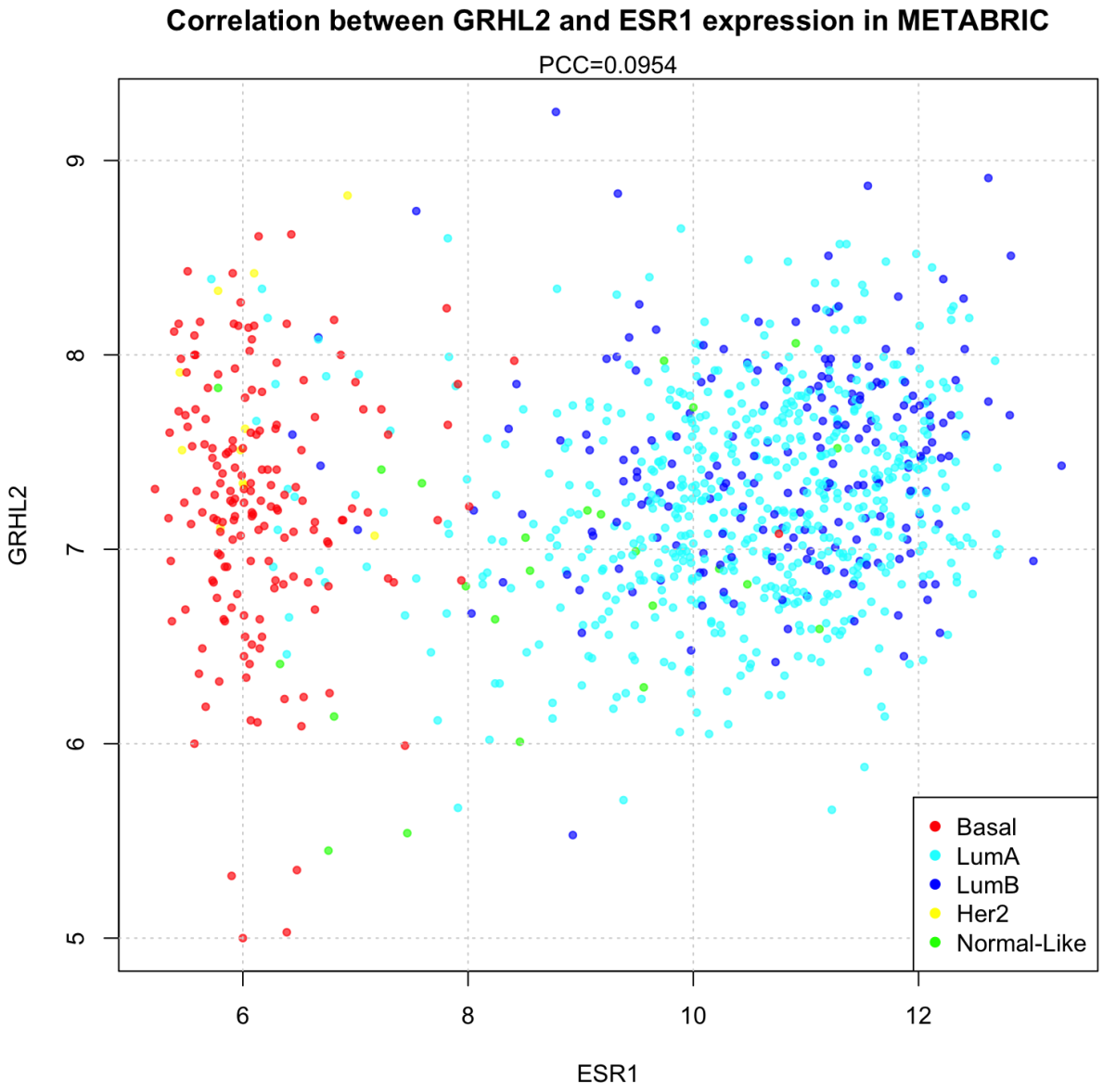


Figure 29: Correlation between GRHL2 and ESR1 expression in the TCGA breast cancer dataset



3.6.3 Enrichment of GRHL2 network for gene pathways

In order to analyze the GRHL2 network, we perform a simple hypergeometric test to overlap its targets with all pathways contained in hte manually curated Reactome collection of MsigDB pathways (C2 v5.0).

```
tf<-"GRHL2"
regulon<-tcga_regulon

# Pathways
wass<-load("msigdb/MSigDB_v5.0_human.rda") # "msigDBentrez" "msigDBsymbol"
ngenes<-length(unique(unlist(msigDBentrez$c2.all.v5.0.entrez.gmt)))

### Hypergeometric test implementation
```

```r
# Hypergeometric enrichment function
enrich<-function(list1,list2,pathway){
  l1<-length(list1)
  l2<-length(list2)
  overlap<-length(intersect(list1,list2))
  q<-overlap-1
  m<-l1
  n<-ngenes-l1
  k<-l2
  p<-phyper(q,m,n,k,lower.tail=FALSE,log.p=FALSE)
  return(p)
}

# Fisher P integration function
fisherp<-function (ps) {
  Xsq <- -2 * sum(log(ps))
  p.val <- pchisq(Xsq, df = 2 * length(ps), lower.tail = FALSE)
  return(p.val)
}

# Regulon vs. pathway
networkUP<-names(regulon[[s2e(tf)]]$tfmode)[regulon[[s2e(tf)]]$tfmode>=0]
networkDN<-names(regulon[[s2e(tf)]]$tfmode)[regulon[[s2e(tf)]]$tfmode<0]

### Enrichment Loop C2
fname<-paste0("results/004_results_hypergeom_C2_",tf,".rda")
#pathways<-msigDBentrez$c2.cp.biocarta.v5.0.entrez.gmt
pathways<-msigDBentrez$c2.all.v5.0.entrez.gmt
if(!file.exists(fname)){
  results<-list()
  ntests<-0
  tfs<-names(regulon)
  pb<-txtProgressBar(0,length(tfs),style=3)
  pbi<-1
  for(tf in tfs){
    plimit<-0.05
    setTxtProgressBar(pb,pbi)
    sublistUP<-setNames(rep(NA,length(pathways)),names(pathways))
    sublistDN<-setNames(rep(NA,length(pathways)),names(pathways))
    i<-1
    for(pathway in pathways){
      networkUP<-names(regulon[[tf]]$tfmode)[regulon[[tf]]$tfmode>=0]
      networkDN<-names(regulon[[tf]]$tfmode)[regulon[[tf]]$tfmode<0]
      pUP<-enrich(networkUP,pathway,ngenes)
      pDN<-enrich(networkDN,pathway,ngenes)
      ntests<-ntests+1
      sublistUP[i]<-pUP
      sublistDN[i]<-pDN
      i<-i+1
    }
    sublistUP<-sublistUP[sublistUP<=plimit]
    sublistDN<-sublistDN[sublistDN<=plimit]
    results[[tf]]<-list(up=sublistUP,dn=sublistDN)
```

```
    pbi<-pbi+1
  }
  save(results,ntests,file=fname)
} else {
  load(fname)
}

### Now, plotting
# Set parameters
topn<-20
subsetting<-"REACTOME_"

## Start plot
# Prepare input results
res<-results[[s2e(tf)]]

# Correct pvalues and integrate
resUP<-p.adjust(res$up,method="none",n=ntests) # convert to bonferroni when deploying
resDN<-p.adjust(res$dn,method="none",n=ntests)
union<-union(names(resUP),names(resDN))
tmptable<-cbind(resUP[union],resDN[union])
tmptable[is.na(tmptable)]<-1
rownames(tmptable)<-union
integrated<-apply(tmptable,1,fisherp)

# Optional subsetting
toshow<-names(sort(integrated))
toshow<-toshow[grep(subsetting,toshow)]
toshow<-toshow[1:topn]


# Prepare input
resUP<--log10(res$up[toshow])
resDN<--log10(res$dn[toshow])
names(resUP)<-toshow
names(resDN)<-toshow
resUP[is.na(resUP)]<-0
resDN[is.na(resDN)]<-0
```

```
# Canvas
upperlim<-ceiling(max(c(resUP,resDN)))
par(mar=c(0,0,0,0))
plot(0,col="white",xlim=c(-10,12),ylim=c(-2,topn+3),xaxt="n",yaxt="n",type="n",frame.plot=FALSE,
     xlab="",ylab="",xaxs="i",yaxs="i")
# text(c(0:10),c(0:10),labels=c(0:10),pos=1,offset=0)
# text(c(1,2,2,1),c(1,2,1,2),labels=c("1_1","2_2","2_1","1_2"),pos=1,offset=0)

# Title
regsize<-length(regulon[[s2e(tf)]]$tfmode)
text(5,topn+2.8,paste0("Pathway enrichment of ",tf," regulon"),pos=1,offset=0,font=2)
text(5,topn+2,paste0("regulon size: ",regsize),pos=1,offset=0,font=3,cex=0.8)

# Axis horizontal
abline(h=0)
```

```
segments(0:10,0,0:10,-0.3)
text(5,-1,"Hypergeometric -log10(pvalue)",pos=1,offset=0)
text(1:10,-0.5,cex=0.7,
     labels=round(signif(seq(0,upperlim,length.out=11)),2)[2:11]
)

# Axis vertical
abline(v=0)
#segments(0,topn:0,-0.3,topn:0)

# Significance line
abline(v=(-log10(0.05)/upperlim)*10,lty=2)

# Bars
for(i in topn:1){
  upval<-(resUP[i]/upperlim)*10
  dnval<-(resDN[i]/upperlim)*10
  rect(0,i+0.25,upval,i,col="salmon")
  rect(0,i,dnval,i-0.25,col="cornflowerblue")
}


# Pathway labels
pathlabels<-gsub(subsetting,"",toshow)
pathlabels<-tolower(pathlabels)
pathlabels<-gsub("_"," ",pathlabels)
for(i in topn:1){
  text(0,i,pathlabels[i],pos=2,offset=0.1,cex=0.8)
}


# Percentage of pathway overlap
r<-names(regulon[[s2e(tf)]]$tfmode)
percs<-c()
for(i in topn:1){
  p<-pathways[[toshow[i]]]
  intrsct<-intersect(p,r)
  perc<-100*length(intrsct)/length(p)
  percs<-c(percs,perc)
}
names(percs)<-rev(toshow)
spercs<-(percs/max(percs))*4
lines(spercs,topn:1,lty=1,lwd=2)

# Ruler of pathway overlap
segments(0,topn+1,4,topn+1)
text(2,topn+1.5,pos=1,offset=0,font=3,cex=0.7,labels="% pathway")
segments(0:4,topn+1,0:4,topn+0.8)
text(1:4,topn+0.65,cex=0.5,
     labels=round(seq(0,max(percs),length.out=5),1)[2:5]
)
```

```
par<-par.backup
```

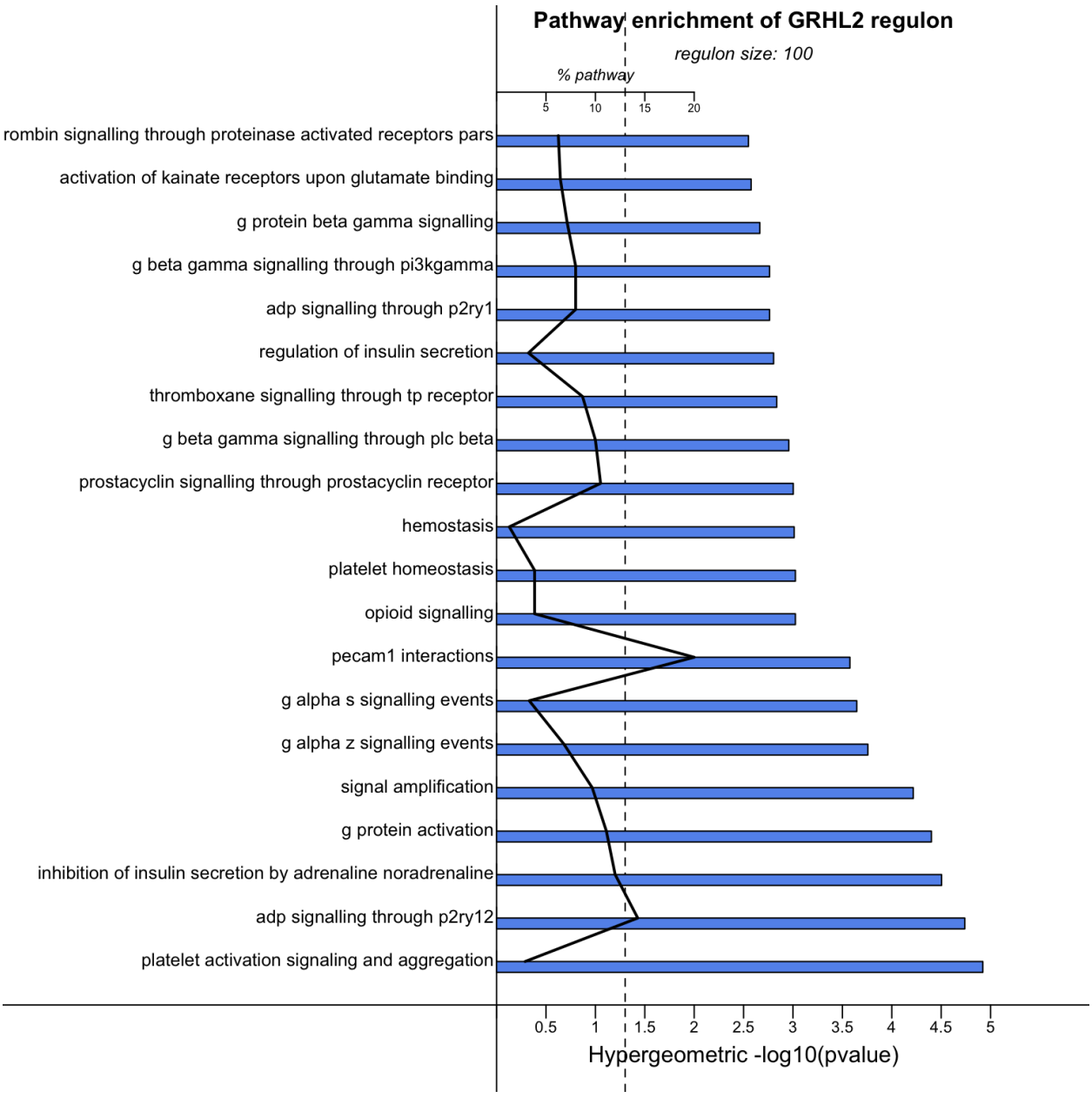


Figure 30: Pathways most significantly overlapping the GRHL2 network

3.7 Comparing VULCAN results with other tools and approaches

Our VULCAN results were obtained using independent gene regulatory models and multiple replicates to generate ER-response signatures.

3.7.1 VULCAN and qPLEX-RIME

We set to testing the performance of VULCAN against a complementary experimental approach called qPLEX-RIME (Holding et al., submitted), which aims at identifyin interactors of ER within the ER-chromatin complex at estrogen-responding promoters. We have QRIME results for MCF7 cells treated with estradiol at both 45 and 90 minutes, the same experimental setup for the VULCAN input dataset.

```
# Load imported vulcan object
load("results/001_vobj.rda")

#### Vulcan Analysis (multiple networks)
## TCGA network, ARACNe with proteomics centroids
load("aracne/regulon-tcga-centroids.rda")
tcga_regulon<-regulon
rm(regulon)

## METABRIC network, ARACNe with proteomics centroids
load("aracne/regulon-metabric-centroids.rda")
metabric_regulon<-regulon
rm(regulon)

## Specify contrast and network
fname<-"results/005_vobj_networks.rda"
list_eg2symbol<-as.list(org.Hs.egSYMBOL[mappedkeys(org.Hs.egSYMBOL)])
if(!file.exists(fname)){
  vobj_tcga_90<-vulcan(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  vobj_tcga_45<-vulcan(vobj,network=tcga_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)
  vobj_metabric_90<-vulcan(vobj,network=metabric_regulon,contrast=c("t90","t0"),annotation=list_eg2symbo
  vobj_metabric_45<-vulcan(vobj,network=metabric_regulon,contrast=c("t45","t0"),annotation=list_eg2symbo
  save(
    vobj_tcga_90,
    vobj_tcga_45,
    vobj_metabric_90,
    vobj_metabric_45,
    file=fname
  )
} else {
  load(fname)
}

# Merge proteomics and TF aracne results
vobj2_tcga_90<-vobj_tcga_90
vobj2_tcga_45<-vobj_tcga_45
vobj2_metabric_90<-vobj_metabric_90
vobj2_metabric_45<-vobj_metabric_45

fname<-"results/002_vobj_networks.rda"
load(fname)
vobj_tcga_90$mrs<-rbind(vobj_tcga_90$mrs,vobj2_tcga_90$mrs[setdiff(rownames(vobj2_tcga_90),rownames(vobj
```

```
vobj_tcga_45$mrs<-rbind(vobj_tcga_45$mrs,vobj2_tcga_45$mrs[setdiff(rownames(vobj2_tcga_45),rownames(vobj
vobj_metabric_90$mrs<-rbind(vobj_metabric_90$mrs,vobj2_metabric_90$mrs[setdiff(rownames(vobj2_metabric_
vobj_metabric_45$mrs<-rbind(vobj_metabric_45$mrs,vobj2_metabric_45$mrs[setdiff(rownames(vobj2_metabric_

# Metabric Results
metabric_90=vobj_metabric_90$mrs[,"pvalue"]
metabric_45=vobj_metabric_45$mrs[,"pvalue"]
tfs<-names(metabric_45)
metabricmat<-cbind(metabric_45[tfs],metabric_90[tfs])
colnames(metabricmat)<-c("T45","T90")

# TCGA Results
tcga_90=vobj_tcga_90$mrs[,"pvalue"]
tcga_45=vobj_tcga_45$mrs[,"pvalue"]
tfs<-names(tcga_45)
tcgamat<-cbind(tcga_45[tfs],tcga_90[tfs])
colnames(tcgamat)<-c("T45","T90")


## QRIME results
raw45<-read.delim("qrime/Results/ER_45min_vs_ER_0minExcludeMissingValues_QuantileNormalization.txt",as.
qrime45<-setNames(raw45$P.Value,raw45$Gene)
qrime45<-qrime45[!is.na(qrime45)]

raw90<-read.delim("qrime/Results/ER_90min_vs_ER_0minExcludeMissingValues_QuantileNormalization.txt",as.
qrime90<-setNames(raw90$P.Value,raw90$Gene)
qrime90<-qrime90[!is.na(qrime90)]
```

Here, we will compare ER-binding pvalues obtained with QRIME, with the VULCAN results aimed at identifying TFs upstream of the observed differential promoter binding.

```
### Comparisons, pvalues
par(mfrow=c(2,2))
# TCGA 45
main<-"TCGA network, 45 mins"
common<-intersect(rownames(tcgamat),names(qrime45))
x<-qrime45[common]
x<--log10(x)
y<-tcgamat[common,"T45"]
y<--log10(y)
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="-log10(p) VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
grid()

# TCGA 90
main<-"TCGA network, 90 mins"
common<-intersect(rownames(tcgamat),names(qrime90))
x<-qrime90[common]
```

```
x<--log10(x)
y<-tcgamat[common,"T90"]
y<--log10(y)
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="-log10(p) VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
grid()


# METABRIC 45
main<-"METABRIC network, 45 mins"
common<-intersect(rownames(metabricmat),names(qrime45))
x<-qrime45[common]
x<--log10(x)
y<-metabricmat[common,"T45"]
y<--log10(y)
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="-log10(p) VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
grid()

# METABRIC 90
main<-"METABRIC network, 90 mins"
common<-intersect(rownames(metabricmat),names(qrime90))
x<-qrime90[common]
x<--log10(x)
y<-metabricmat[common,"T90"]
y<--log10(y)
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="-log10(p) VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
grid()
```

```
# TCGA Results
tcganes_90=vobj_tcga_90$mrs[,"NES"]
tcganes_45=vobj_tcga_45$mrs[,"NES"]
tfs<-names(tcganes_45)
tcganesmat<-cbind(tcganes_45[tfs],tcganes_90[tfs])
colnames(tcganesmat)<-c("T45","T90")
```

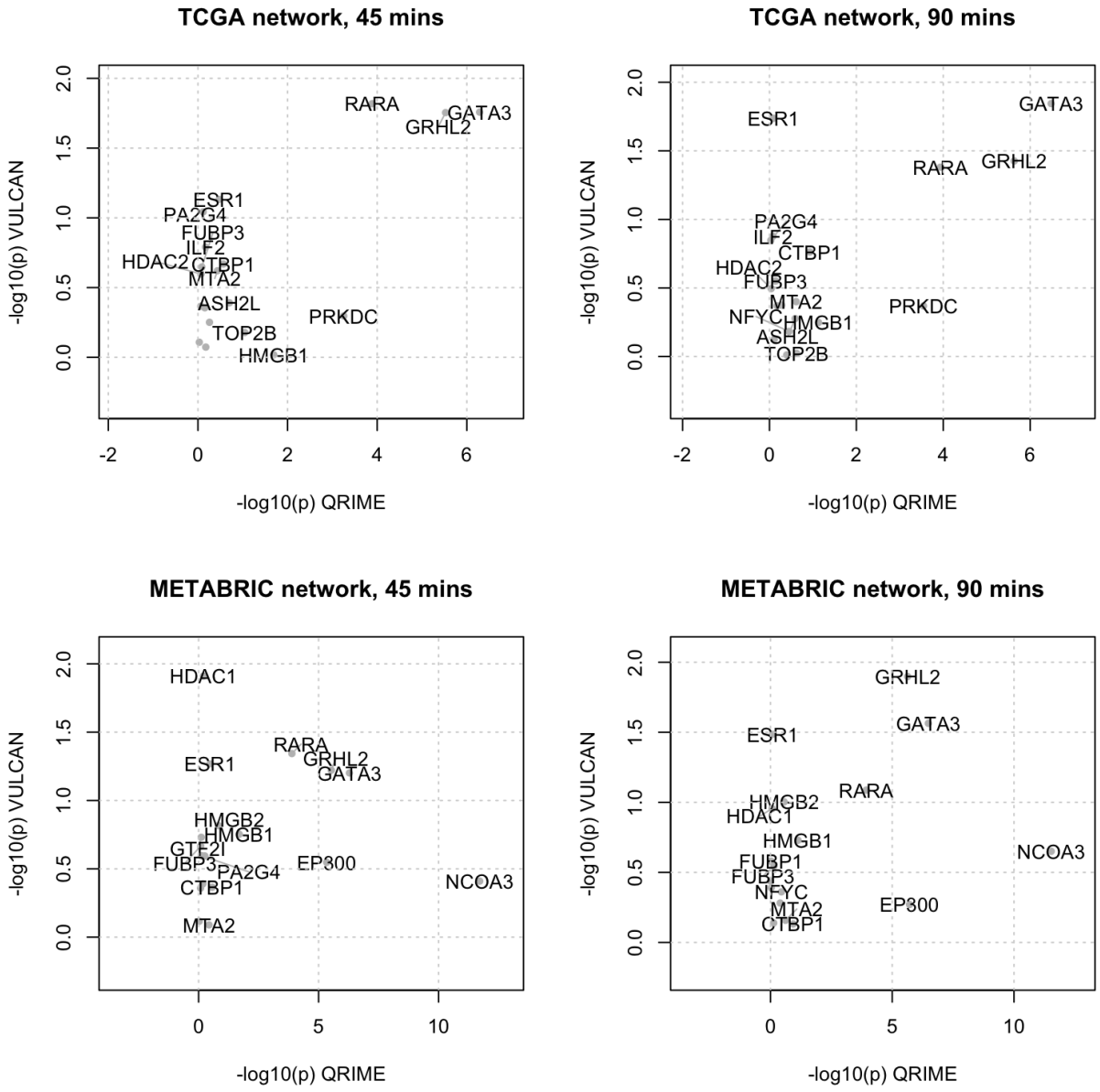


Figure 31: Comparison of significance between the QRIME method (x-axis) and the VULCAN method (y-axis) at two time points using two regulatory networks for VULCAN

```
# METABRIC Results
metabricnes_90=vobj_metabric_90$mrs[,"NES"]
metabricnes_45=vobj_metabric_45$mrs[,"NES"]
tfs<-names(metabricnes_45)
metabricnesmat<-cbind(metabricnes_45[tfs],metabricnes_90[tfs])
colnames(metabricnesmat)<-c("T45","T90")




par(mfrow=c(2,2))
# TCGA 45
main<-"TCGA network, 45 mins"
common<-intersect(rownames(tcganesmat),names(qrime45))
x<-qrime45[common]
x<--log10(x)
y<-tcganesmat[common,"T45"]
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="NES VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
abline(h=c(-p2z(0.05),p2z(0.05)),lty=3)
grid()

# TCGA 90
main<-"TCGA network, 90 mins"
common<-intersect(rownames(tcganesmat),names(qrime90))
x<-qrime90[common]
x<--log10(x)
y<-tcganesmat[common,"T90"]
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="NES VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
abline(h=c(-p2z(0.05),p2z(0.05)),lty=3)
grid()


# METABRIC 45
main<-"METABRIC network, 45 mins"
common<-intersect(rownames(metabricnesmat),names(qrime45))
x<-qrime45[common]
x<--log10(x)
y<-metabricnesmat[common,"T45"]
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="NES VULCAN",col="grey",main=main,
```

```
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
abline(h=c(-p2z(0.05),p2z(0.05)),lty=3)
grid()

# METABRIC 90
main<-"METABRIC network, 90 mins"
common<-intersect(rownames(metabricnesmat),names(qrime90))
x<-qrime90[common]
x<--log10(x)
y<-metabricnesmat[common,"T90"]
plot(x,y,pch=20,xlab="-log10(p) QRIME",ylab="NES VULCAN",col="grey",main=main,
     xlim=c(min(x)-max(x)*0.3,max(x)*1.1),
     ylim=c(min(y)-max(y)*0.2,max(y)*1.1)
)
topx<-names(sort(x,dec=TRUE))[1:10]
topy<-names(sort(y,dec=TRUE))[1:10]
top<-union(topx,topy)
textplot2(x[top],y[top],words=top,new=FALSE)
abline(h=c(-p2z(0.05),p2z(0.05)),lty=3)
grid()
```

3.7.2 Comparing Mutual Information and Partial Correlation networks

As shown before in literature, partial correlation and mutual information networks are highly similar. While requiring a mutual information network due to the VIPER engine used by the VULCAN pipeline, we set out here to compare Mutual Information networks with partial correlation ones, using different correlation thresholds.

```
filename<-"results/006_comparison1_aracne.rda"
if(!file.exists(filename)){
  # Load aracne network
  load("aracne/regulon-tcga-centroids.rda")

  # Load expression matrix
  expmat<-as.matrix(read.delim("aracne/tcga-expmat.dat",as.is=TRUE,row.names=1))


  ### Better pipeline
  tfs<-names(regulon)
  genes<-rownames(expmat)


  # Remove zero-SD genes
  sds<-apply(expmat,1,sd)
  expmat<-expmat[sds>=0.5,]
  dim(expmat)
```

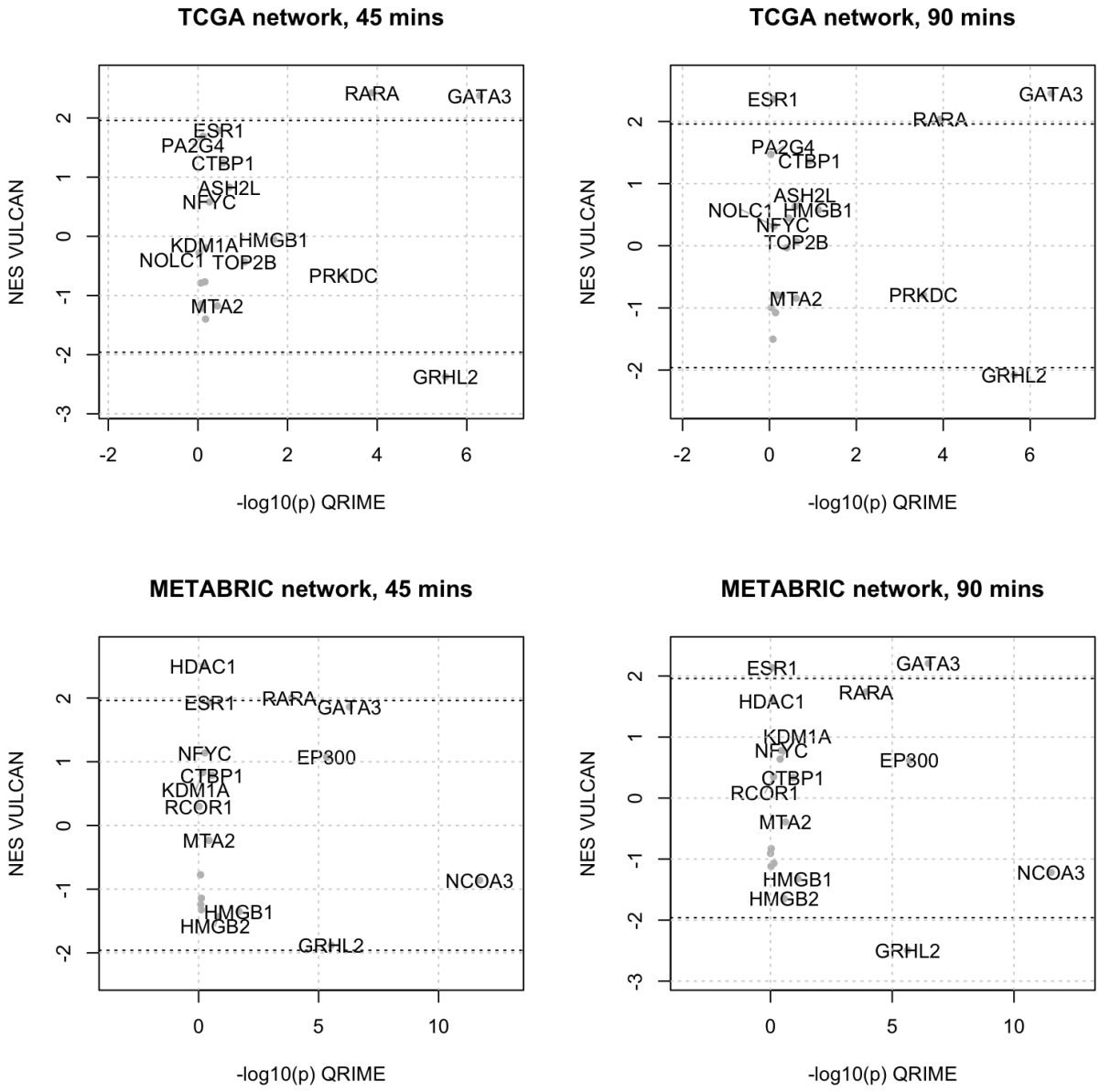


Figure 32: Comparison of Normalized Enrichment Score between the QRIME method (x-axis) and the VULCAN method (y-axis) at two time points using two regulatory networks for VULCAN

```
# Build Covariance Matrix
pcormat<-cov(t(expmat))
# Build pcormat
pcormat<-cor2pcor(pcormat)
rownames(pcormat)<-colnames(pcormat)<-rownames(expmat)


# Keep only TFs
diag(pcormat)<-0
tfs<-intersect(tfs,rownames(pcormat))
pcormat<-pcormat[tfs,]

# Transform my network into a vector of edges
myedges<-c()
for(centroid in names(regulon)){
  targets<-names(regulon[[centroid]]$tfmode)
  newedges<-paste0(centroid,"___",targets)
  myedges<-c(myedges,newedges)
}
length(myedges) # 268394

### Loop over rs
jis<-c()
nedges<-c()
rs<-seq(0.05,0.4,by=0.025)
for(r in rs){
  # Transform networks into vectors of edges
  matches<-which(abs(pcormat)>=r,arr.ind=TRUE)
  pcoredges<-paste0(rownames(pcormat)[matches[,1]],"___",colnames(pcormat)[matches[,2]])
  nedges<-c(nedges,nrow(matches))
  # Calculate Jaccard
  ji<-length(intersect(myedges,pcoredges))/length(union(myedges,pcoredges))
  jis<-c(jis,ji)
}
names(jis)<-names(nedges)<-rs



# Generate random network vectors
random<-list()
for(ii in 1:length(rs)){
  n<-nedges[ii]
  message("Doing ",rs[ii])
  pb<-txtProgressBar(0,1000,style=3)
  randomjis<-c()
  for(i in 1:1000){
    x<-sample(1:nrow(pcormat),n,replace=TRUE)
    y<-sample(1:ncol(pcormat),n,replace=TRUE)
    matches<-cbind(x,y)
    pcoredges<-paste0(rownames(pcormat)[matches[,1]],"___",colnames(pcormat)[matches[,2]])
    ji<-length(intersect(myedges,pcoredges))/length(union(myedges,pcoredges))
    randomjis<-c(randomjis,ji)
```

```
      setTxtProgressBar(pb,i)
    }
    random[[ii]]<-randomjis
  }
  names(random)<-as.character(rs)

  save(jis,rs,nedges,random,file=filename)

} else {
  load(filename)
}
```

We generate several partial correlation networks using the same input as the ARACNe network used by VULCAN (the TCGA breast cancer dataset). We tested the overlap of every partial correlation network with the ARACNe network using the Jaccard Index (JI) criterion.

```
par(mfrow=c(1,1))
bp<-barplot(jis,xlab="pcor r",ylab="JI",ylim=c(0,max(jis)*1.1))
text(bp[,1],jis,labels=kmgformat(nedges),pos=3,cex=0.7)
mtext("Number of overlapping edges between Pcor and Aracne",cex=0.8)
```

Finally, we show how the Jaccard Index between partial correlation networks and the ARACNe network is always significantly higher than expected by selecting random network edges.

```
par(mfrow=c(5,3))
for(i in 1:length(random)){
  dd<-density(random[[i]])
  plot(dd,xlim=c(min(random[[i]]),jis[i]*1.1),main=paste0("r=",rs[i]),xlab="Jaccard Index",lwd=2)
  arrows(jis[i],max(dd$y),jis[i],0,lwd=2)
}
```

3.7.3 Comparing alternative methods for target enrichment analysis

We developed three independent methods to compare VULCAN with. The first is common, the second is simple but crude, and the third is also common but too stringent.

```
### Classic vulcan result
# vobj_tcga_90<-vulcan(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
wass<-load("results/002_vobj_networks.rda")

### Prepare the workspace
# Load imported vulcan object
load("results/001_vobj.rda")

# Network
load("networks/brca-tf-regulon.rda")
tcga_regulon<-regul
rm(regul)

# Entrez to symbol
library(org.Hs.eg.db)
list_eg2symbol<-as.list(org.Hs.egSYMBOL[mappedkeys(org.Hs.egSYMBOL)])
```

1. A T-test based method, that takes the targets of a TF and integrates their pvalue in a specific contrast. Good but hard to keep control over the p-value (the classic integration step using the Fisher's Integration

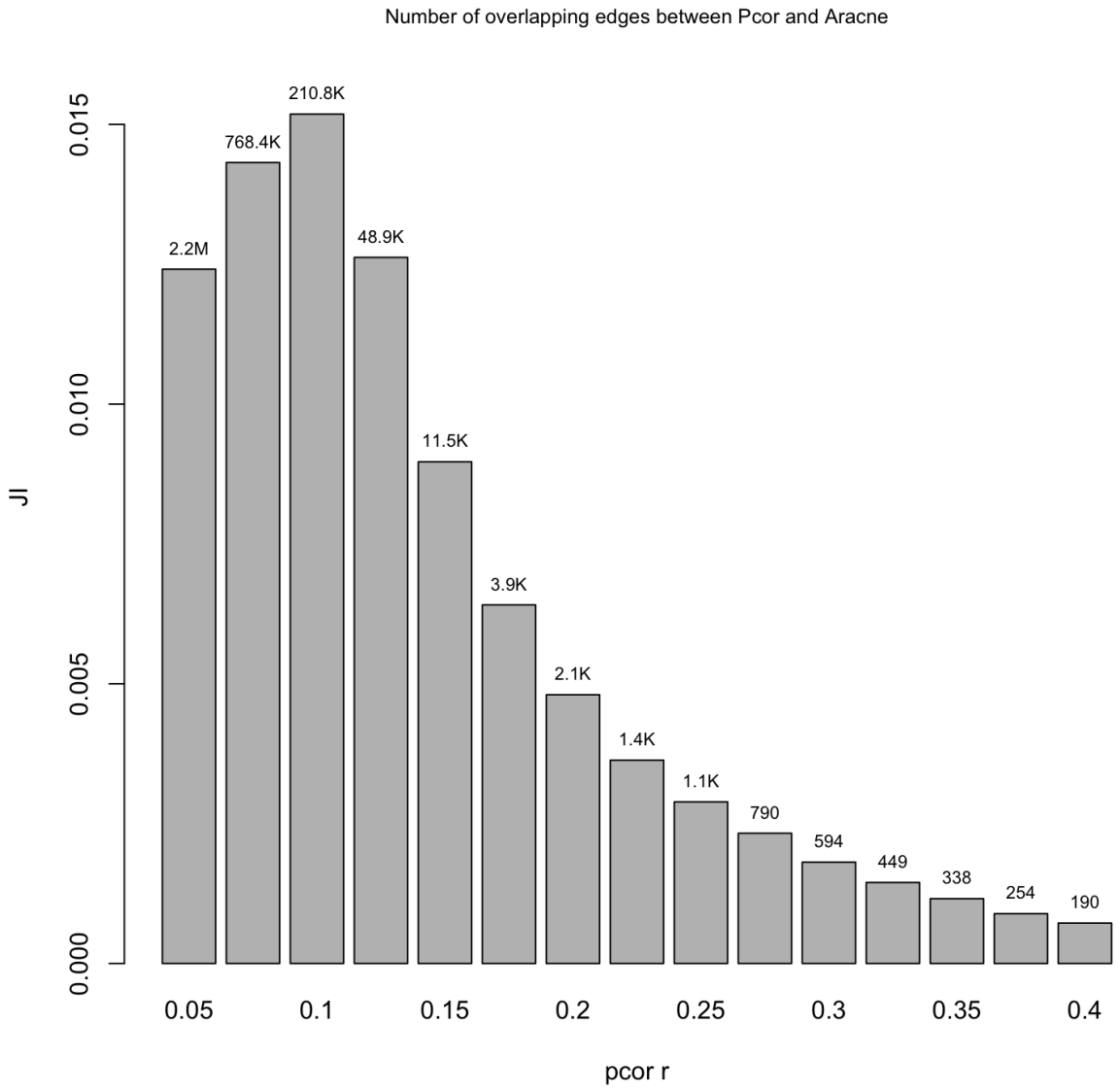


Figure 33: Sheer number of overlapping edges at different Pcor thresholds

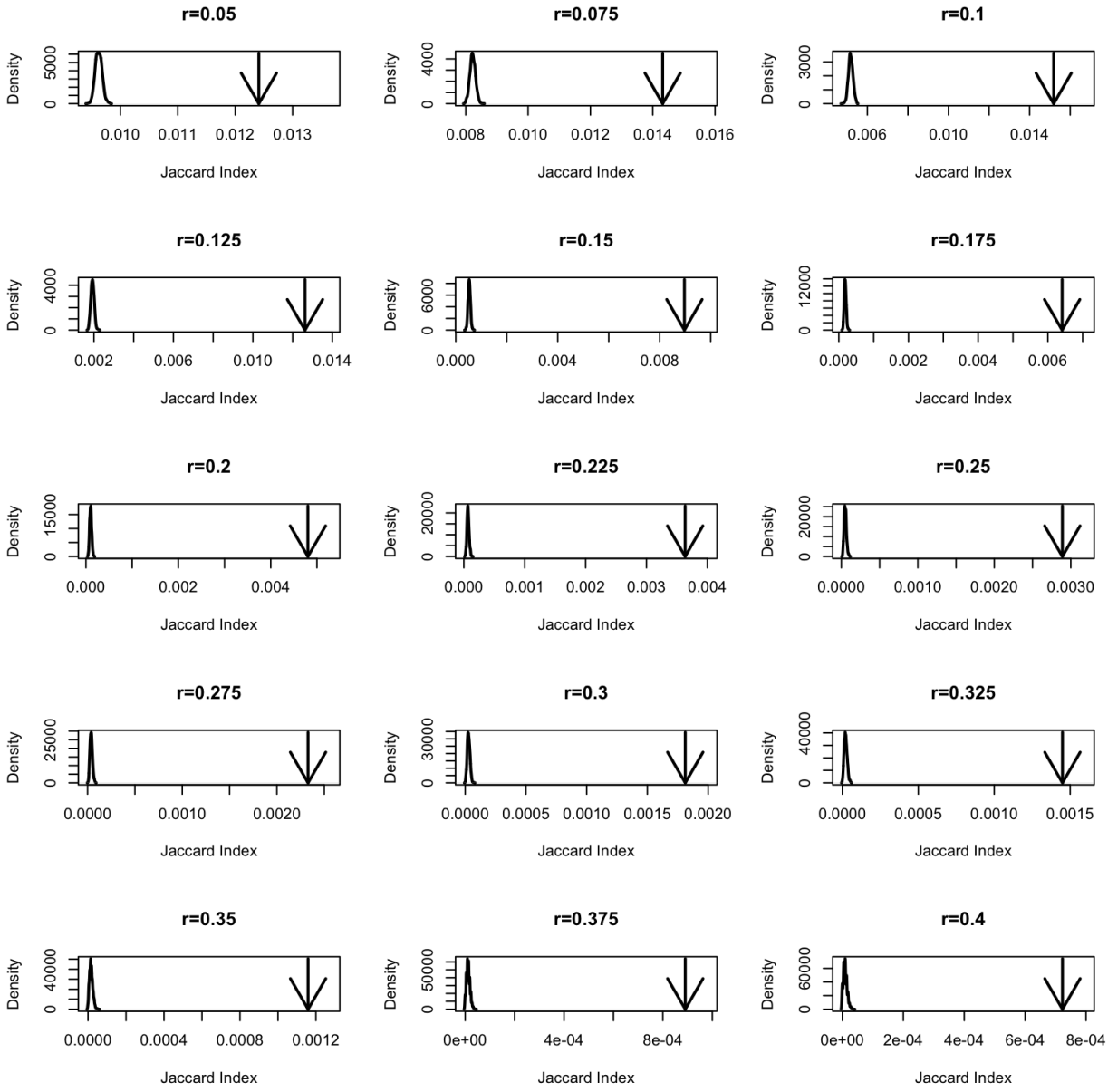


Figure 34: Comparing the Jaccard Index between the ARACNe network and Partial correlation networks (indicated as arrows) and random networks (indicated as distributions)

method vastly underestimates them). Also, GRHL2 doesn't shine with this method. Unless differently specified, the p-values are always Bonferroni-corrected

```
#########################################################
### TTEST INTEGRATION
#########################################################
if(TRUE){
  ### TTest functions
  msvipertt<-function(signature,network,minsize=10){
    # First, convert tscores into pvalues
    psignature<-2*pt(signature[,1], nrow(signature)-2, lower=FALSE)

    # For each regulon, take the targets and integrate their pvalue
    tfpvalues<-c()
    for(tf in names(network)){
      targets<-names(network[[tf]]$tfmode)
      ptargets<-psignature[intersect(targets,names(psignature))]
      if(length(ptargets)>=minsize){
        pt<-fisherp(ptargets)
        tfpvalues<-c(tfpvalues,pt)
        names(tfpvalues)[length(tfpvalues)]<-tf
      }
    }

    return(tfpvalues)
  }
  vulcantt<-function(vobj, network, contrast, annotation = NULL, minsize = 10) {
    tfs <- names(network)
    samples <- vobj$samples
    normalized <- vobj$normalized

    # Prepare output objects
    msvipers <- matrix(NA, ncol = 3, nrow = length(tfs))
    rownames(msvipers) <- tfs
    # Define contrast
    a <- samples[[contrast[1]]]
    b <- samples[[contrast[2]]]
    # Vulcan msviper implementation
    signature <- rowTtest(normalized[, a], normalized[, b])$statistic
    tfpvalues <- msvipertt(signature, network, minsize = minsize)
    if(!is.null(annotation)){
      names(tfpvalues)<-annotation[names(tfpvalues)]
    }
    return(tfpvalues)
  }

  ### Repeat the same analysis but with TT
  ttp_90<-p.adjust(vulcantt(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  ttp_45<-p.adjust(vulcantt(vobj,network=tcga_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)

  ### Compare vulcan and TT
  vulcanp_90<-vobj_tcga_90$msviper$es$p.value
  vulcanp_45<-vobj_tcga_45$msviper$es$p.value
```

```
  toshow<-c("ESR1","GATA3","GRHL2")

  par(mfrow=c(1,2))

  common<-intersect(names(ttp_45),names(vulcanp_45))
  x<--log10(vulcanp_45[common]+1e-320)
  y<--log10(ttp_45[common]+1e-320)
  plot(x,y,pch=20,main="Method Comparison, 45' vs 00'",xlab="VULCAN pvalue",ylab="Ttest integration pval
  grid()
  pcc<-cor.test(x,y)
  mtext(paste0("PCC=",signif(pcc$estimate,3)," (p=",signif(pcc$p.value,3),")"))
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

  common<-intersect(names(ttp_90),names(vulcanp_90))
  x<--log10(vulcanp_90[common]+1e-320)
  y<--log10(ttp_90[common]+1e-320)
  plot(x,y,pch=20,main="Method Comparison, 90' vs 00'",xlab="VULCAN pvalue",ylab="Ttest integration pval
  grid()
  pcc<-cor.test(x,y)
  mtext(paste0("PCC=",signif(pcc$estimate,3)," (p=",signif(pcc$p.value,3),")"))
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

  par(mfrow=c(1,1))

}
```

2. A fraction of targets method, defining for every TF the fraction of their targets that are also differentially bound. A crude alternative to VULCAN, which is ignoring the MI strength of interaction and the individual strengths of differential bindings.

```
#########################################################
### FRACTION TEST
#########################################################
if(TRUE){
  ### TTest functions
  msviperfrac<-function(signature,network,minsize=10){
    # First, convert tscores into pvalues
    psignature<-2*pt(signature[,1], nrow(signature)-2, lower=FALSE)

    # For each regulon, take the targets and integrate their pvalue
    tfpvalues<-c()
    for(tf in names(network)){
      targets<-names(network[[tf]]$tfmode)
      ptargets<-psignature[intersect(targets,names(psignature))]
      ptargets<-ptargets
        frac<-length(ptargets)/length(targets)
        tfpvalues<-c(tfpvalues,frac)
        names(tfpvalues)[length(tfpvalues)]<-tf
    }
    return(tfpvalues)
  }
  vulcanfrac<-function(vobj, network, contrast, annotation = NULL, minsize = 10) {
    tfs <- names(network)
    samples <- vobj$samples
```

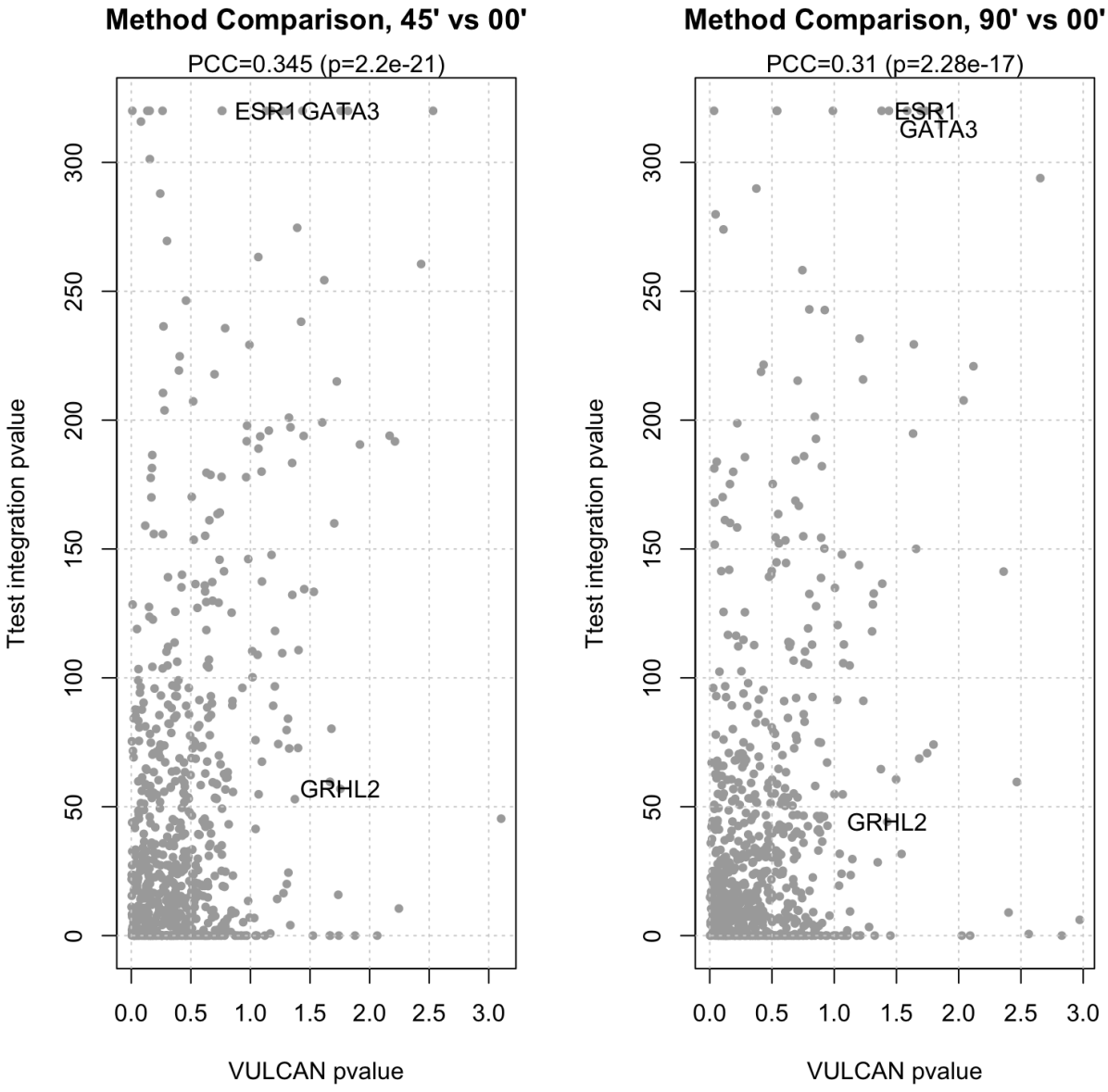


Figure 35: Comparison between VULCAN/VIPER and T-test integration

```
    normalized <- vobj$normalized

    # Prepare output objects
    msvipers <- matrix(NA, ncol = 3, nrow = length(tfs))
    rownames(msvipers) <- tfs
    # Define contrast
    a <- samples[[contrast[1]]]
    b <- samples[[contrast[2]]]
    # Vulcan msviper implementation
    signature <- rowTtest(normalized[, a], normalized[, b])$statistic
    tfpvalues <- msviperfrac(signature, network, minsize = minsize)
    if(!is.null(annotation)){
      names(tfpvalues)<-annotation[names(tfpvalues)]
    }
    return(tfpvalues)
  }

  ### Repeat the same analysis but with TT
  ttp_90<-vulcanfrac(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  ttp_45<-vulcanfrac(vobj,network=tcga_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)

  ### Compare vulcan and TT
  vulcanp_90<-vobj_tcga_90$msviper$es$p.value
  vulcanp_45<-vobj_tcga_45$msviper$es$p.value

  toshow<-c("ESR1","GATA3","GRHL2")

  par(mfrow=c(1,2))

  common<-intersect(names(ttp_45),names(vulcanp_45))
  x<--log10(vulcanp_45[common])
  y<-ttp_45[common]
  plot(x,y,pch=20,main="Method Comparison, 45' vs 00'",xlab="VULCAN pvalue",ylab="Fraction of Differenti
  grid()
  pcc<-cor.test(x,y)
  mtext(paste0("PCC=",signif(pcc$estimate,3)," (p=",signif(pcc$p.value,3),")"))
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

  common<-intersect(names(ttp_90),names(vulcanp_90))
  x<--log10(vulcanp_90[common])
  y<-ttp_90[common]
  plot(x,y,pch=20,main="Method Comparison, 90' vs 00'",xlab="VULCAN pvalue",ylab="Fraction of Differenti
  grid()
  pcc<-cor.test(x,y)
  mtext(paste0("PCC=",signif(pcc$estimate,3)," (p=",signif(pcc$p.value,3),")"))
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

  par(mfrow=c(1,1))

}
```

3. A Fisher's Exact Method (different from the Fisher's Integration!) which assesses the overlap between networks and significant differential binding. Here the problem is that the test is too stringent (as observed in the original VIPER paper) and even without p-value correction we get nothing significant

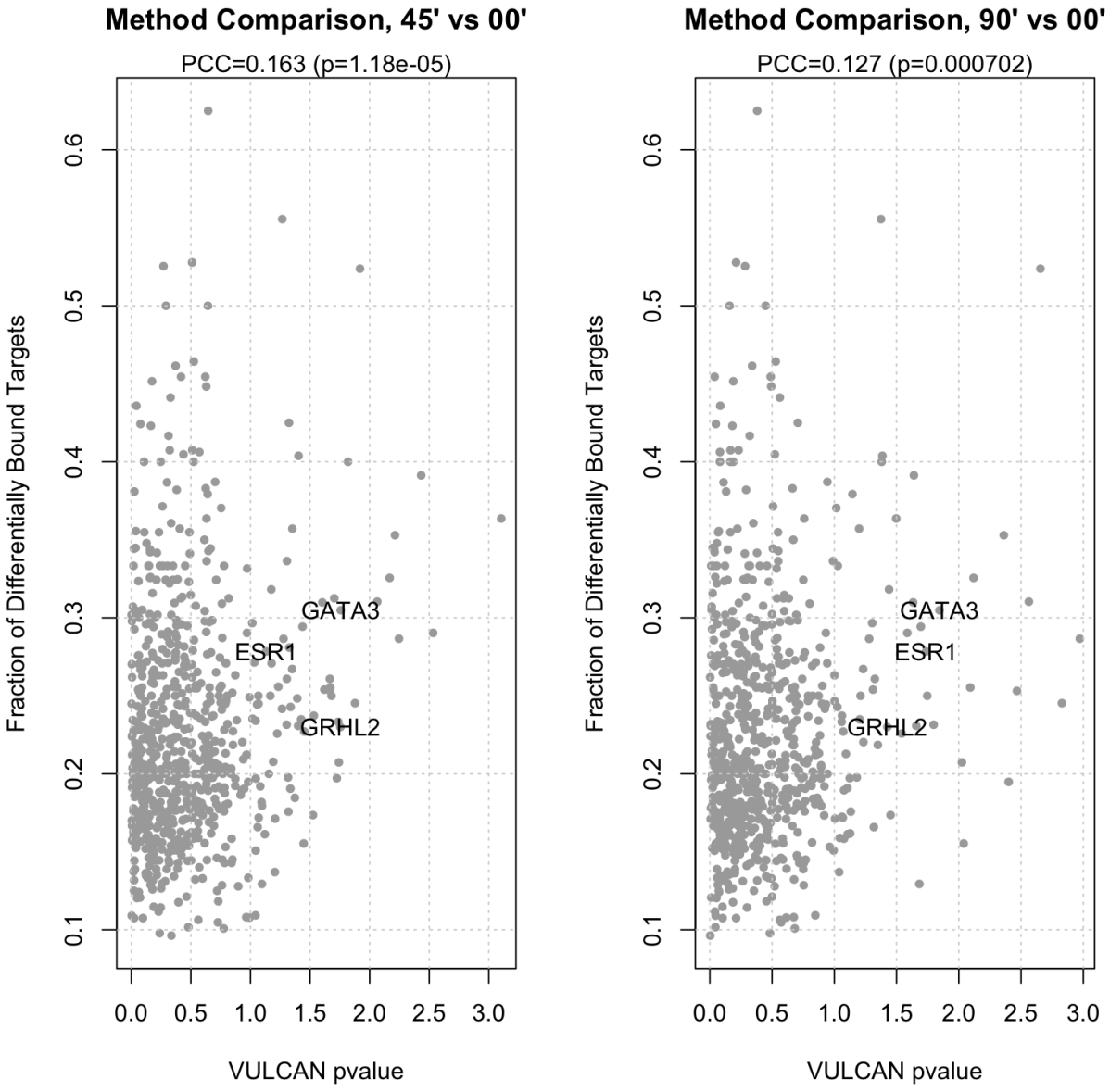


Figure 36: Comparison between VULCAN/VIPER and a fraction of targets found method

(I tried different threshold of differential binding significance). The plot is uninspiring highlights the non-significance of the results of this method:

```
#########################################################
### FISHER'S EXACT TEST
#########################################################
if(TRUE){
  ### TTest functions
  msviperfet<-function(signature,network,minsize=10){
    # First, convert tscores into pvalues
    psignature<-2*pt(signature[,1], nrow(signature)-2, lower=FALSE)

    # For each regulon, take the targets and calculate how many are significant
    tfpvalues<-c()
    for(tf in names(network)){
      targets<-names(network[[tf]]$tfmode)
      ptargets<-psignature[intersect(targets,names(psignature))]
      if(length(ptargets)>=minsize){
        # Prepare the contingency table
        ul<-names(ptargets)[ptargets<=0.05]
        ur<-setdiff(targets,ul)
        dl<-setdiff(names(psignature)[psignature<=0.05],ul)
        ctable<-rbind(
          c(length(ul),length(ur)),
          c(length(dl),0)
        )
        #ctable[2,2]<-5000-sum(ctable)
        fet<-fisher.test(ctable,alternative="greater")
        pt<-fet$p.value
        tfpvalues<-c(tfpvalues,pt)
        names(tfpvalues)[length(tfpvalues)]<-tf
      }
    }

    return(tfpvalues)
  }
  vulcanfet<-function(vobj, network, contrast, annotation = NULL, minsize = 10) {
    tfs <- names(network)
    samples <- vobj$samples
    normalized <- vobj$normalized

    # Prepare output objects
    msvipers <- matrix(NA, ncol = 3, nrow = length(tfs))
    rownames(msvipers) <- tfs
    # Define contrast
    a <- samples[[contrast[1]]]
    b <- samples[[contrast[2]]]
    # Vulcan msviper implementation
    signature <- rowTtest(normalized[, a], normalized[, b])$statistic
    tfpvalues <- msviperfet(signature, network, minsize = minsize)
    if(!is.null(annotation)){
      names(tfpvalues)<-annotation[names(tfpvalues)]
    }
    return(tfpvalues)
```

```
  }

  ### Repeat the same analysis but with FET
  ttp_90<-vulcanfet(vobj,network=tcga_regulon,contrast=c("t90","t0"),annotation=list_eg2symbol)
  ttp_45<-vulcanfet(vobj,network=tcga_regulon,contrast=c("t45","t0"),annotation=list_eg2symbol)

  ### Compare vulcan and TT
  vulcanp_90<-vobj_tcga_90$msviper$es$p.value
  vulcanp_45<-vobj_tcga_45$msviper$es$p.value

  toshow<-c("ESR1","GATA3","GRHL2")

  par(mfrow=c(1,2))

  common<-intersect(names(ttp_45),names(vulcanp_45))
  x<--log10(vulcanp_45[common]+1e-320)
  y<--log10(ttp_45[common]+1e-320)
  plot(x,y,pch=20,main="Method Comparison, 45' vs 00'",xlab="VULCAN pvalue",ylab="Classic Fisher's Exac
  grid()
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

  common<-intersect(names(ttp_90),names(vulcanp_90))
  x<--log10(vulcanp_90[common]+1e-320)
  y<--log10(ttp_90[common]+1e-320)
  plot(x,y,pch=20,main="Method Comparison, 90' vs 00'",xlab="VULCAN pvalue",ylab="Classic Fisher's Exac
  grid()
  textplot2(x[toshow],y[toshow],words=toshow,new=FALSE)

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

3.7.4 Comparing VULCAN with online tools

We will generate BED files for the peaks, as reported by DiffBind, we will then test the pathway enrichment for each of these peaks using the Great software v3.0.0 (http://bejerano.stanford.edu/great/public/html/), the ChIP-Enrich pipeline (http://chip-enrich.med.umich.edu/) and the ISMARA tool (http://ismara.unibas.ch). Parameters for GREAT: defaults (Basal plus extension, 5.0kb upstream, 1.0 kb downstream) Parameters for ChIP-Enrich: defaults (Nearest TSS, pathways size<=2000, Biocarta, KEGG, Reactome, TFs). Parameters for ISMARA: defaults (ChIP-Seq mode, hg19)

The VULCAN analysis shows a significant overlap in terms of significant pathways with the GREAT method. ChIP-enrich computes enrichment for a number of TFs which are amongst the most significant in VULCAN, but it surprisingly fails at identifing ESR1 as the top Transcription Factor affected by our experiment. ISMARA succeeds at identifying ESR1 using a motif-based analysis, but doesn't identify other candidate binding TFs, as expected, being the experiment targeted at the estrogen receptor.

```
### Number of tested pathways (for statistical comparison later)
load("msigdb/MSigDB_v5.0_human.rda")
raw<-msigDBentrez$c2.all.v5.0.entrez.gmt
universe<-names(raw)[grep("BIOCARTA_|REACTOME_|KEGG_|PID_|ST_",names(raw))]

### DiffBind object (containing peak location and intensity)
wass<-load("results/001_diffbind.rda")
```

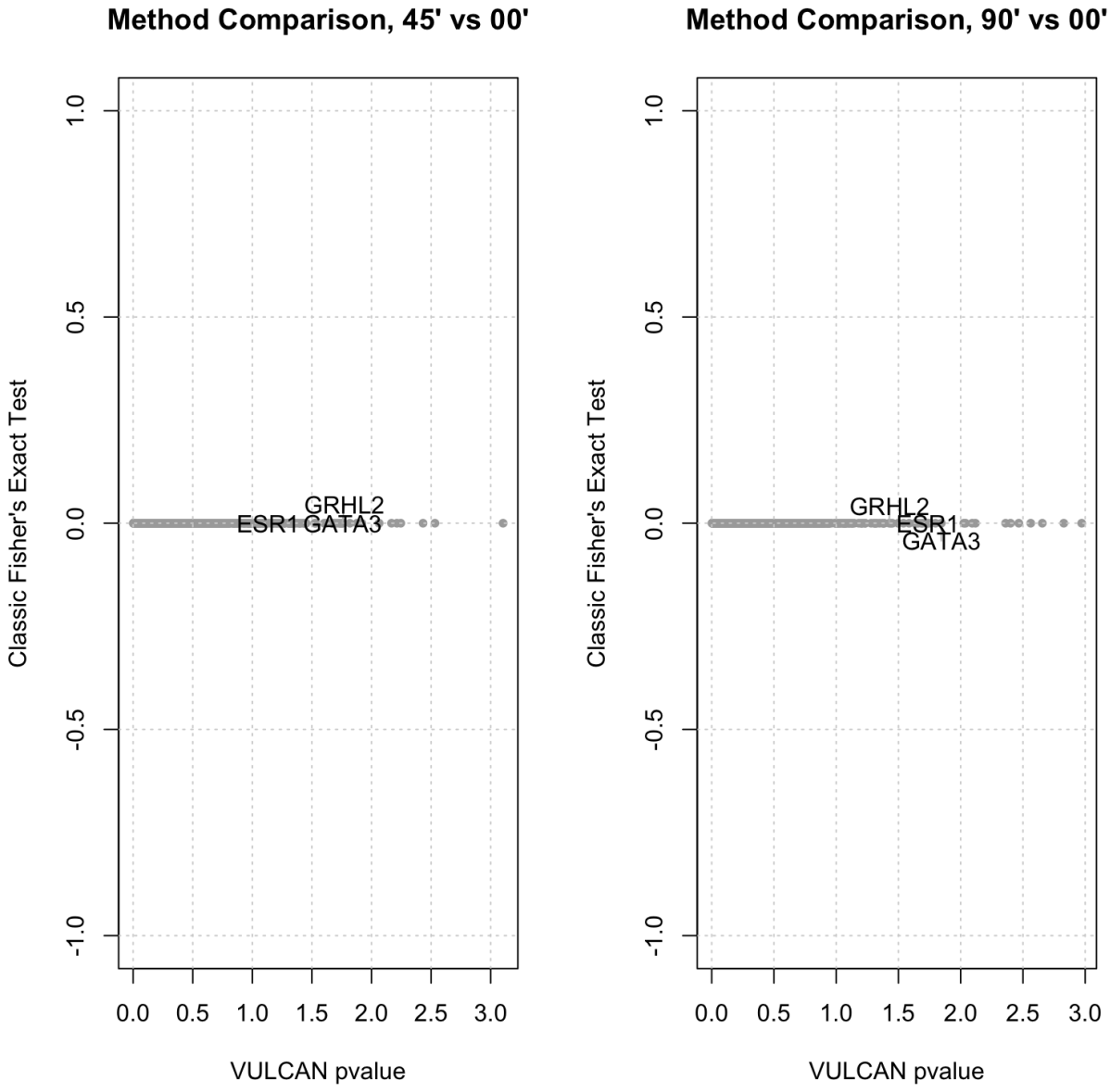


Figure 37: Comparison between VULCAN/VIPER and Fisher's Exact Test method

```
### Generate input BEDs for chip enrich and GREAT (ismara takes BAMs as input)
fname<-paste0("results/008_comparison_contrast90vs00_UP_p1.bed")
if(!file.exists(fname)){
  contrasts<-c("contrast45vs00","contrast90vs00","contrast90vs45")
  for (contrast in contrasts) {
    c<-switch(contrast,contrast45vs00=3,contrast90vs00=2,contrast90vs45=1)
    for(pgreat in c(1,0.05)){
      bed<-as.data.frame(dba.report(dbaobj,contrast=c,method=DBA_DESEQ2,bNormalized=TRUE, bCounts=TRUE,
      if(nrow(bed)>0) { # Nothing individually significant if no rows were produced
        bedup<-bed[bed$Fold>0,1:3]
        beddn<-bed[bed$Fold<0,1:3]
        if(nrow(bedup)>0){
          write.table(bedup,
                      file=paste0("results/008_comparison_",contrast,"_UP_p",pgreat,".bed"),
                      row.names=FALSE,col.names=FALSE,sep="\t",quote=FALSE
          )
        }
        if(nrow(beddn)>0){
          write.table(beddn,
                      file=paste0("results/008_comparison_",contrast,"_DN_p",pgreat,".bed"),
                      row.names=FALSE,col.names=FALSE,sep="\t",quote=FALSE
          )
        }
      }
    }
  }
}
```

```
### Pathway Comparison: VULCAN vs. GREAT
load_obj <- function(f){
  env <- new.env()
  nm <- load(f, env)[1]
  env[[nm]]
}
contrasts<-c("45","90")
par(mfrow=c(2,1))
for(c in contrasts){
  # Our REA pathways
  nes.tpathways<-load_obj(paste0("results/003_pathwayComparison_REA_",c,".rda")) # results_rea_45

  # GREAT pathways
  rawgreat<-read.delim(paste0("results/great/great_",c,"_p1_UP.tsv"),as.is=TRUE,skip=3)
  table(rawgreat[,1])
  rawgreat<-rawgreat[rawgreat[,1]=="MSigDB Pathway",]
  great<-setNames(rawgreat$BinomBonfP,rawgreat[,2])
  great<-great[great<0.1]

  ### Comparison GREAT/VULCAN
  vsig<-nes.tpathways[z2p(nes.tpathways)<0.1]
  venn(list(VULCAN=names(vsig),GREAT=names(great)))
  title(paste0("Comparison VULCAN/GREAT for pathways at ",c," minutes"))
  ctable<-rbind(c(0,0),c(0,0))
  ctable[1,1]<-length(intersect(names(vsig),names(great)))
```

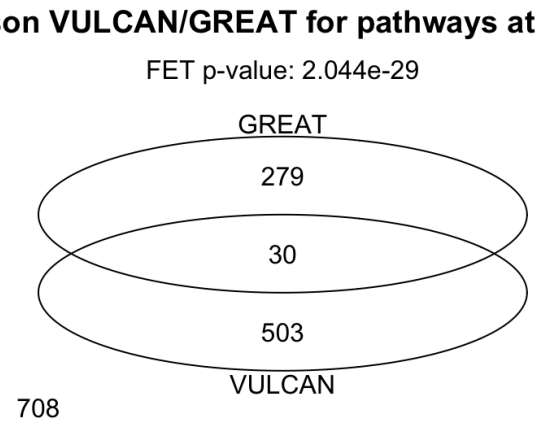
```
  ctable[1,2]<-length(setdiff(names(vsig),names(great)))
  ctable[2,1]<-length(setdiff(names(great),names(vsig)))
  ctable[2,2]<-length(universe)-ctable[1,1]-ctable[1,2]-ctable[2,1]
  fp<-signif(fisher.test(ctable)$p.value,4)
  mtext(paste0("FET p-value: ",fp))
  text(100,3,ctable[2,2])

  common<-intersect(names(nes.tpathways),names(great))
}
```

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

```
### TF and Pathway Comparison: VULCAN vs. ChIP enrich
load("results/002_vobj_networks.rda")
contrasts<-c("45","90")
par(mfrow=c(2,1))
for(c in contrasts){
  # Our VULCAN TF enrichment
  if(c=="45"){
    vulcannes<-vobj_tcga_45$msviper$es$nes
  }
  if(c=="90"){
    vulcannes<-vobj_tcga_90$msviper$es$nes
  }
  # ChIPEnrich pathways + TFs
  rawchipenrich<-read.delim(paste0("results/chipenrichr/",c,"_p1_UP_results.tab"),as.is=TRUE)
  chipenrich_tfs<-rawchipenrich[rawchipenrich[,1]=="Transcription Factors",]
  chipenrich_pathways<-rawchipenrich[rawchipenrich[,1]!="Transcription Factors",]
  t1<-gsub(" ","_",paste0(toupper(chipenrich_pathways[,1]),"_",toupper(chipenrich_pathways[,3])))
  t2<-chipenrich_pathways[,"FDR"]
  chipenrich_pathways<-setNames(t2,t1)

  # TF comparison
  cetfs<-setNames(chipenrich_tfs[,"FDR"],gsub("_.+","",chipenrich_tfs[,"Description"]))
  vutfs<-vulcannes[!is.na(vulcannes)]
  vutfs<-vutfs[vutfs>0]
  common<-intersect(names(vutfs),names(cetfs))

  plot(vutfs,-log10(z2p(vutfs)),xlab="VULCAN NES",ylab="VULCAN -log10(p)",pch=20,col="grey",
       main=paste0("Comparison VULCAN/ChIP-enrich at ",c)
  )
  mtext("Labels indicate significant TFs by ChIP-enrich binding site analysis",cex=0.8)
  set.seed(1)
  textplot2(vutfs[common],-log10(z2p(vutfs))[common],common,new=FALSE)
}
```

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

```
### Comparison: VULCAN vs. ISMARA
ismara<-t(read.delim("results/ismara/ismara_report/activity_table",as.is=TRUE))
vulcan45<-vobj_tcga_45$msviper$es$nes
vulcan90<-vobj_tcga_90$msviper$es$nes

# Compare VULCAN (x axis) at 45 and 90 minutes with ISMARA (yaxis) average + sd
```



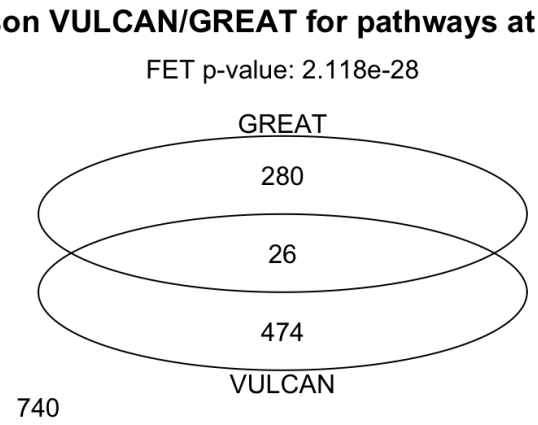


Figure 38: Comparison of results from the VULCAN and GREAT methods

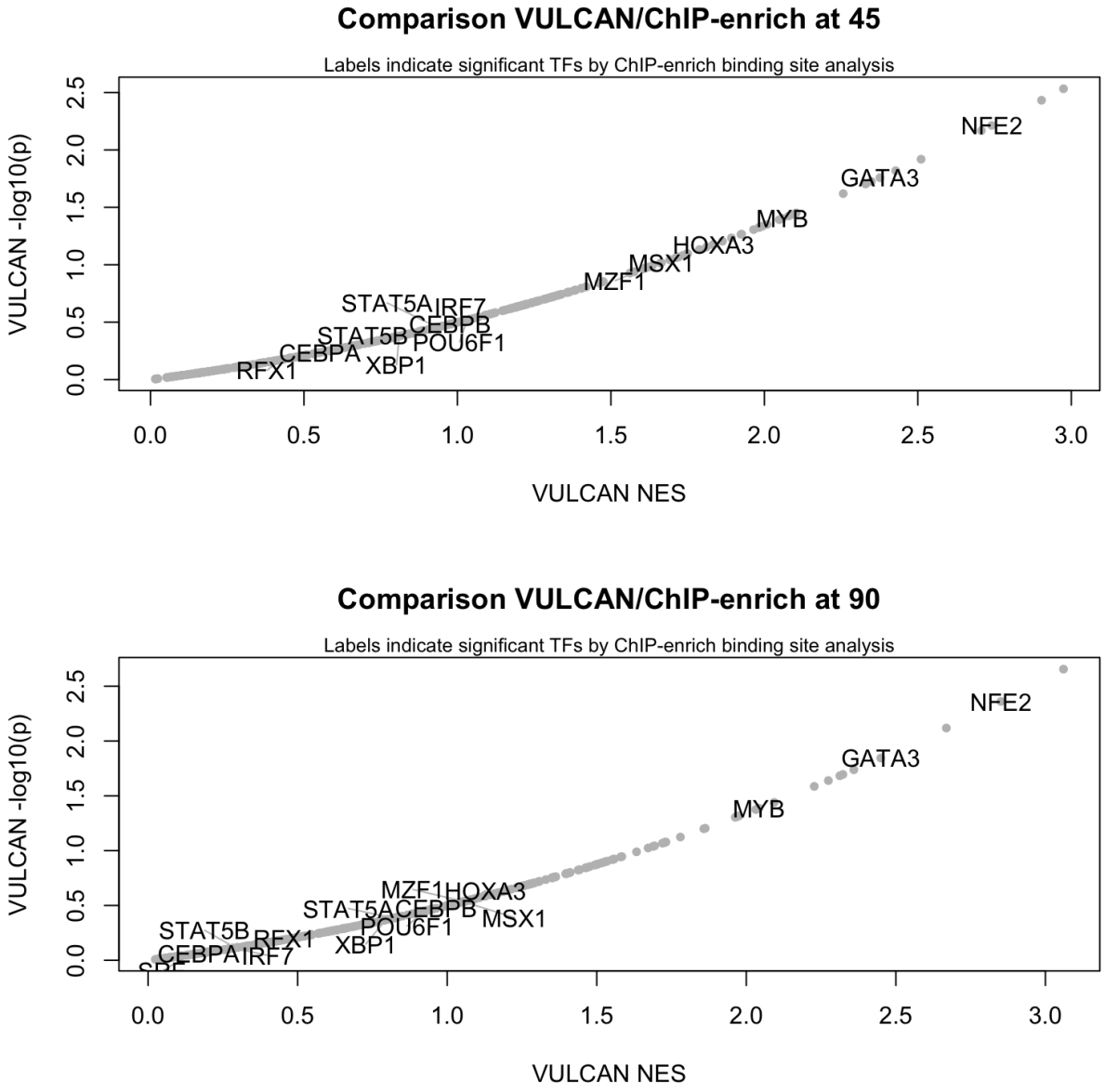


Figure 39: Comparison of results from the VULCAN and ChIP-enrich methods

```r
# Prepare ismara values
i45<-ismara[,grep("45",colnames(ismara))]
i90<-ismara[,grep("90",colnames(ismara))]
i45_mean<-apply(i45,1,mean)
i45_sd<-apply(i45,1,sd)
i90_mean<-apply(i90,1,mean)
i90_sd<-apply(i90,1,sd)

common<-intersect(names(vulcan45),rownames(ismara))
length(common) # 148
```

```
## [1] 148
```

```r
# Plots
par(mfrow=c(2,1))

# 45 minutes
x<-vulcan45[common]
y<-i45_mean[common]
uiw<-i45_sd[common]
top<-intersect(names(sort(x,dec=TRUE))[1:20],names(sort(y,dec=TRUE))[1:20])
plotCI(x,y,uiw=uiw,xlab="VULCAN NES",ylab="ISMARA Activity",main="Comparison VULCAN/ISMARA",pch=20,col=
mtext("45 minutes",cex=0.8)
textplot2(x[top],y[top],words=top,new=FALSE,font=2)
# 90 minutes
x<-vulcan90[common]
y<-i90_mean[common]
uiw<-i90_sd[common]
top<-intersect(names(sort(x,dec=TRUE))[1:25],names(sort(y,dec=TRUE))[1:25])
plotCI(x,y,uiw=uiw,xlab="VULCAN NES",ylab="ISMARA Activity",main="Comparison VULCAN/ISMARA",pch=20,col=
mtext("90 minutes",cex=0.8)
textplot2(x[top],y[top],words=top,new=FALSE,font=2)
```

4 Estrogen Receptor-alpha qPLEX-RIME Analysis

4.1 Introduction

This report is from an analysis of the RIME-TMT proteomics data for an experiment carried out by Andrew Holding to quantify specific interactors of estrogen receptor (ER) and forkhead box protein A1 (FOXA1) in MCF-7 cells.

The RIME (Rapid Immunoprecipitation of Endogenous Proteins) technique was used to pull down ER and FOXA1 and their interacting proteins, which were subsequently assessed and quantitated using mass spectrometry with tandem mass tags (TMT). TMT runs were carried out for 3 biological replicates, estrogen-starved MCF-7 cells from the same cell line at different times/passages, before and after addition of estrogen with samples taken at 3 different time points – 0, 45 and 90 minutes after estrogen addition. A pull down of Immunoglobulin G (IgG) was also run as a control.

The digests from two of the biological replicates (PR622 and PR650) were analyzed with 3 separate runs of the mass spectrometer to increase coverage; data from the repeated runs were combined prior to the analysis carried out here. The other sample (PR590) was run once only.

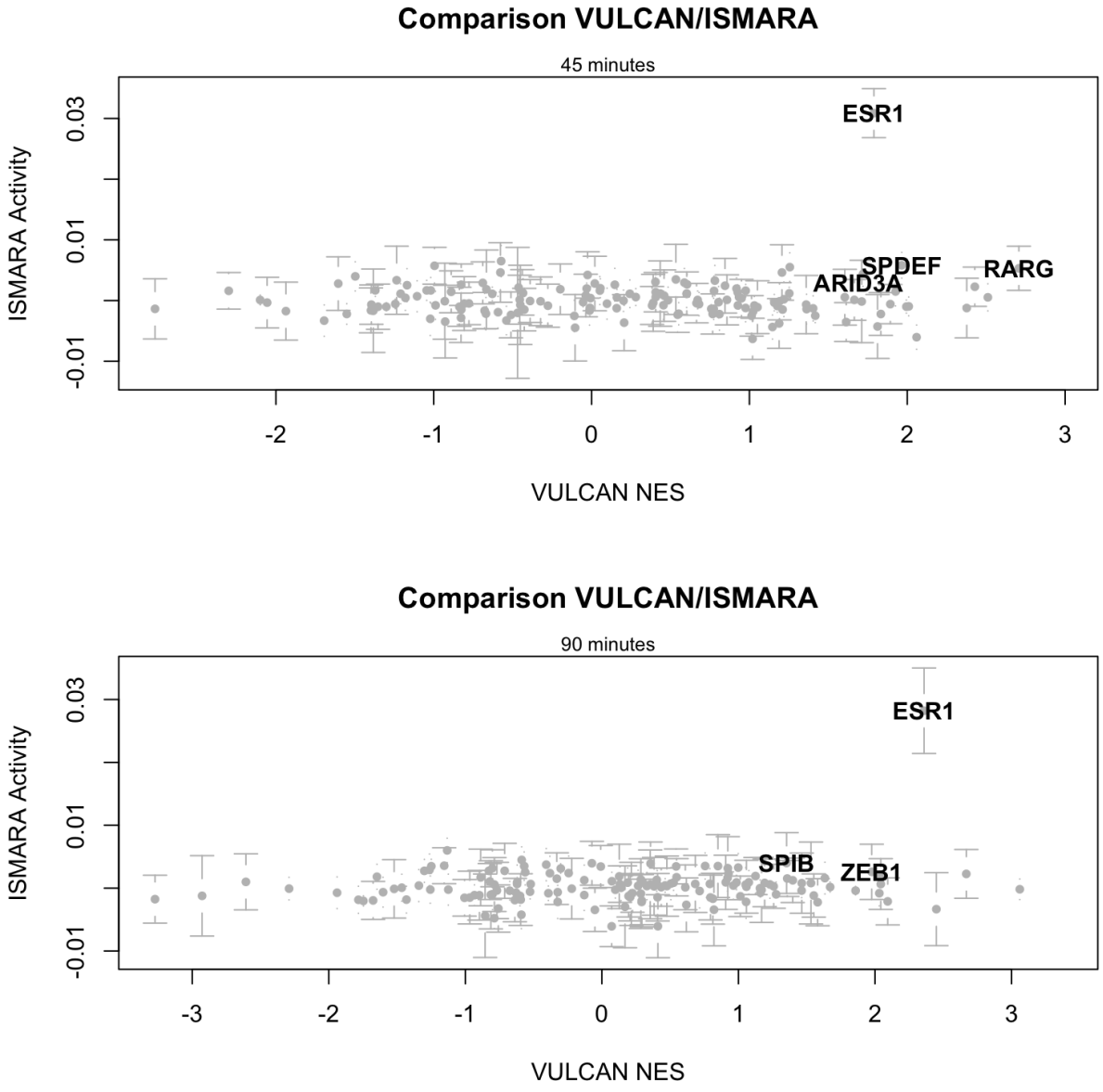


Figure 40: Comparison of results from the VULCAN and ISMARA methods

Table 1 shows the isobaric tags used for each sample within each of the runs.

Group	PR590	PR622	PR650
ER 0min	127N	127N	128N
ER 45min	128C	127C	128C
ER 90min	129N	128N	129N
FOXA1 0min	129C	129C	129C
FOXA1 45min	130N	130N	130N
FOXA1 90min	130C	130C	130C
IgG	126	126	131

Table 1: Isobaric tags used for each sample (group) and run.

4.2 Peptide intensity data

Raw spectra were processed using Proteome Discover 2.1 to produce peptide-level intensity data with a single set of intensity values per distinct peptide. Only peptides with unique high-confidence protein matches were included.

Multiple peptide-spectrum matches (PSMs) for the same peptide were combined using Proteome Discoverer by summing the PSM-level intensities. Peptide sequences with different modifications are treated as distinct peptides and the data provided contain intensities values for each modification identified for a peptide sequence. Several distinct peptide sequences (with modifications) may have been identified for any given protein.

Table 2 shows the numbers of distinct peptides and proteins obtained in each TMT run. Also given are the numbers of peptides and proteins after filtering peptides with missing intensity values in one or more TMT channels.

	PR590	PR622	PR650	All
Peptides	5822	6952	3183	9321
Peptides with no missing values	5702	6872	3118	9179
Proteins	1125	1227	656	1553
Proteins with no missing values	1112	1218	645	1541

Table 2: Numbers of peptides and proteins observed in each run.

Figure 1 shows the distribution of intensities for each sample within each run.

4.2.1 Missing intensity values

Missing intensity values can result even though ions are present at detectable concentrations due to the stochastic nature of the mass spectrometry acquisition method. It is reasonable to expect that these missing values are randomly distributed in the data. Alternatively, missing values may occur when there is a low abundance of ions, below the limit of detection of the instrument. These biologically relevant missing values are not randomly distributed, affecting only those proteins that are expressed at low levels in the samples analysed. The R/Bioconductor package `MSnBase` provides imputation methods for both types of missing value.

In this analysis, missing values have been handled in 3 different ways. The first approach is to exclude all peptide-level mesasurements where there is a missing value for one or more of the samples within a run. In addition, two imputation methods have been performed: one in which the missing values are set to the

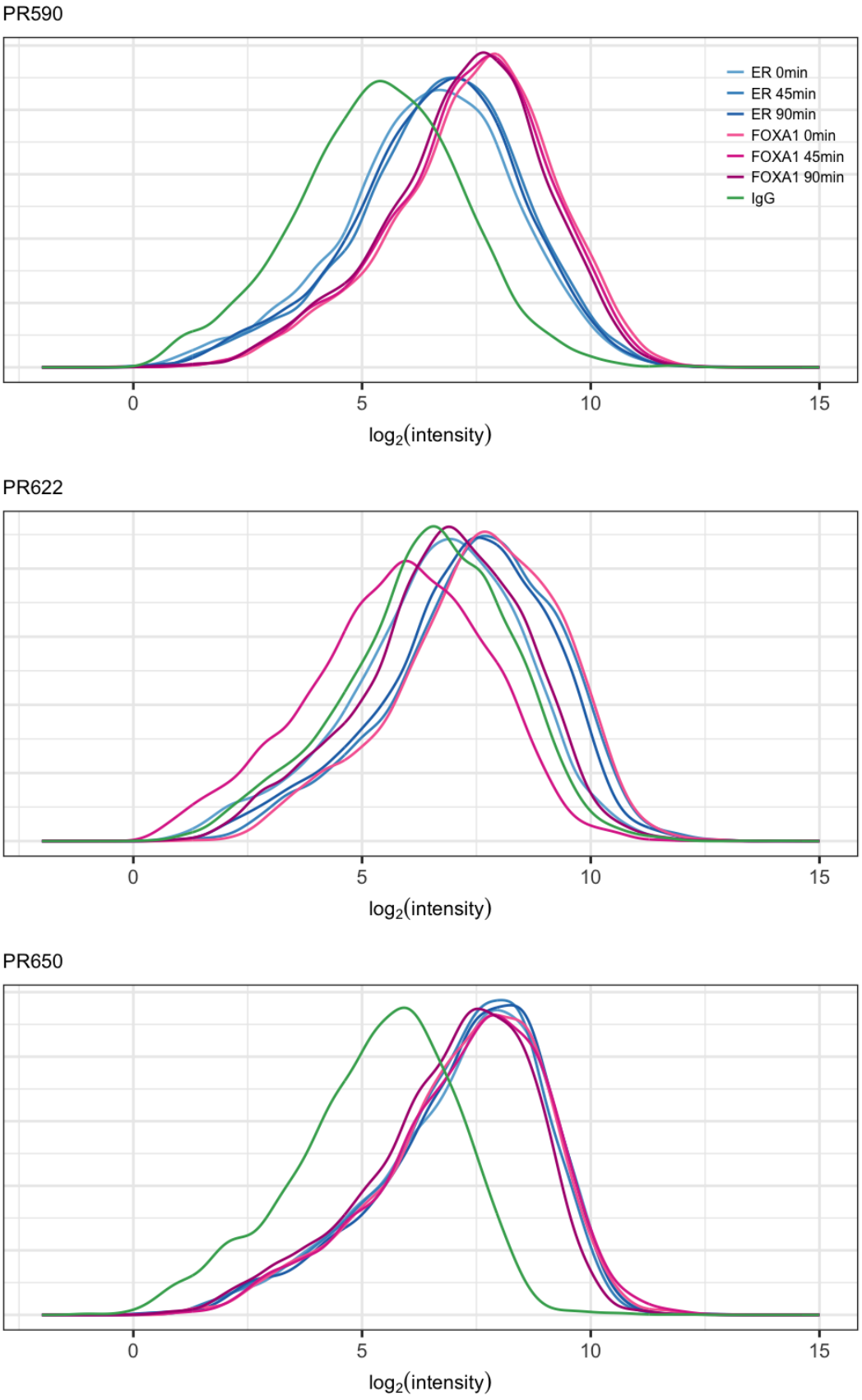


Figure 41: Density plots of intensities from each run.

smallest non-missing value in the data for that run and the other in which k-nearest neighbour (KNN) averaging is applied.

Figure 2 shows the distribution of intensities for each sample within each run following imputation using the smallest non-missing values. A small hump just below zero on the $\log_2$ scale is clearly visible for each run.

4.3 Normalization of intensity data

The intensity distribution for PR622 suggests that the labeled samples have been pooled with differing protein concentrations and that some normalization is required to properly assess differences between groups. Normalization techniques that are commonly applied to microarray expression data and mRNA sequencing data assume that only a few genes, e.g. a few hundred out of tens of thousands, are expressed at very different levels between samples. This assumption may not hold for a RIME experiment where a specific set of interacting proteins is targeted. For example, if the majority of interacting proteins were to bind less strongly at one time point relative to another, quantile normalization or scale normalization approaches would have the effect of removing the general trend. In this case, following normalization, those proteins that bind less strongly may appear to be largely unchanged, while proteins with very similar levels of binding at the different time points would have artificially increased binding levels. Therefore some care is required when interpreting the results of a differential binding analysis for RIME TMT experiments.

In this analysis, quantile normalization and scale normalization were applied for each run separately.

The IgG control was used to assess non-specific binding. The binding of a protein was considered to be non-specific if the binding level of that protein in the condition of interest, e.g. ER at 45 minutes, was not significantly above the level observed for the IgG control. Proteins detected with the IgG control are expressed at lower levels in the PR590 and PR650 runs. Normalization of the IgG control along with the other samples would make it more difficult to distinguish between specific and non-specific binding since the intensities of the IgG control would be scaled to a similar level of those of the other samples.

Among a number of normalization approaches tested here, normalizations were carried out that both include and exclude the IgG control. Figure 3 shows the normalized intensity distributions following scale normalization of all peptide intensities including those with missing values. Figure 4 shows the normalized intensity distributions following scale normalization for all samples except the IgG controls.

A further normalization approach was attempted that scales the data from all samples within a run based on a subset of peptides that are considered most likely to be at similar concentrations between the samples. The assumption is that the peptides with the highest intensity values in the IgG control are the result of non-specific binding and that that binding is consistent across all samples. Similar to the scale normalization applied in the first approach, the median intensity within each sample is computed to determine a scaling factor for each sample within a run but the median computed only uses the 10 peptides with the highest IgG measurements. As before, normalization is carried out within each run separately.

Figure 5 shows the resulting normalized intensity distributions. Strikingly, the IgG intensity distribution is shifted to lower values in run PR622, making it more consistent with the other 2 runs.

4.4 Protein-level quantification

Protein-level quantification was carried out for each run separately by summing normalized intensity values for all peptides matching to a particular protein. This is similar in principle to the gene-level quantification that is carried out during microarray analysis or the read-counting approach in RNA-Seq. Essentially, all signal attributable to a particular protein is assigned to that protein. Intensity values vary in magnitude depending on the elution profile for the peptide and when the peptide is sampled. Relative intensities for a peptide between the tags or samples within a run should be consistent but the signal may include some contribution from a contaminating, co-eluting peptide, the extent of which may differ for different PSMs. In summing intensities from multiple measurements for the same peptide, those peptides that have higher

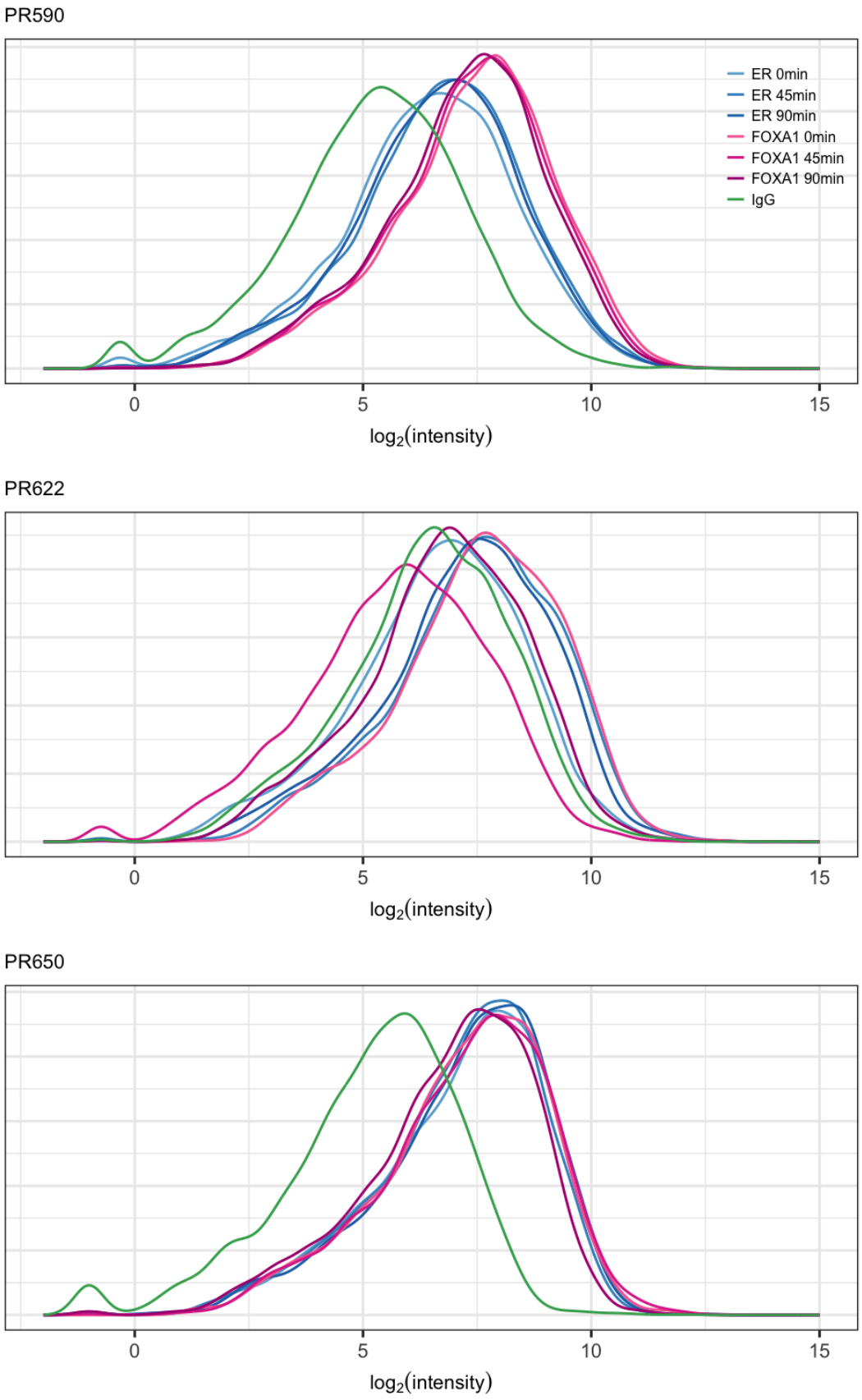


Figure 42: Density plots of intensities from each run following imputation of missing values using the smallest non-missing value in the data for each run.

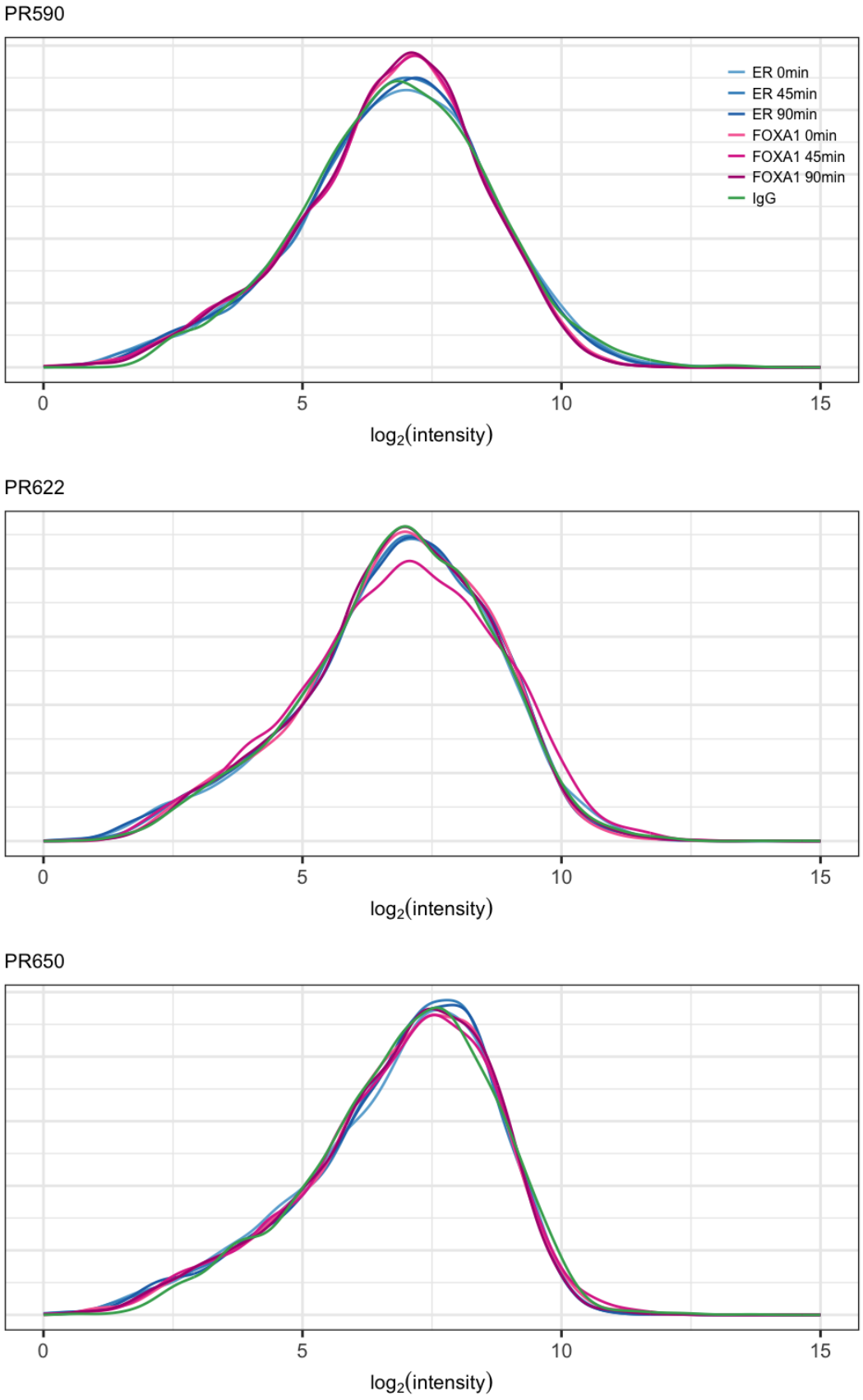


Figure 43: Density plots of normalized intensities from each TMT run where scale normalization was applied to peptide intensities that include misssing values.

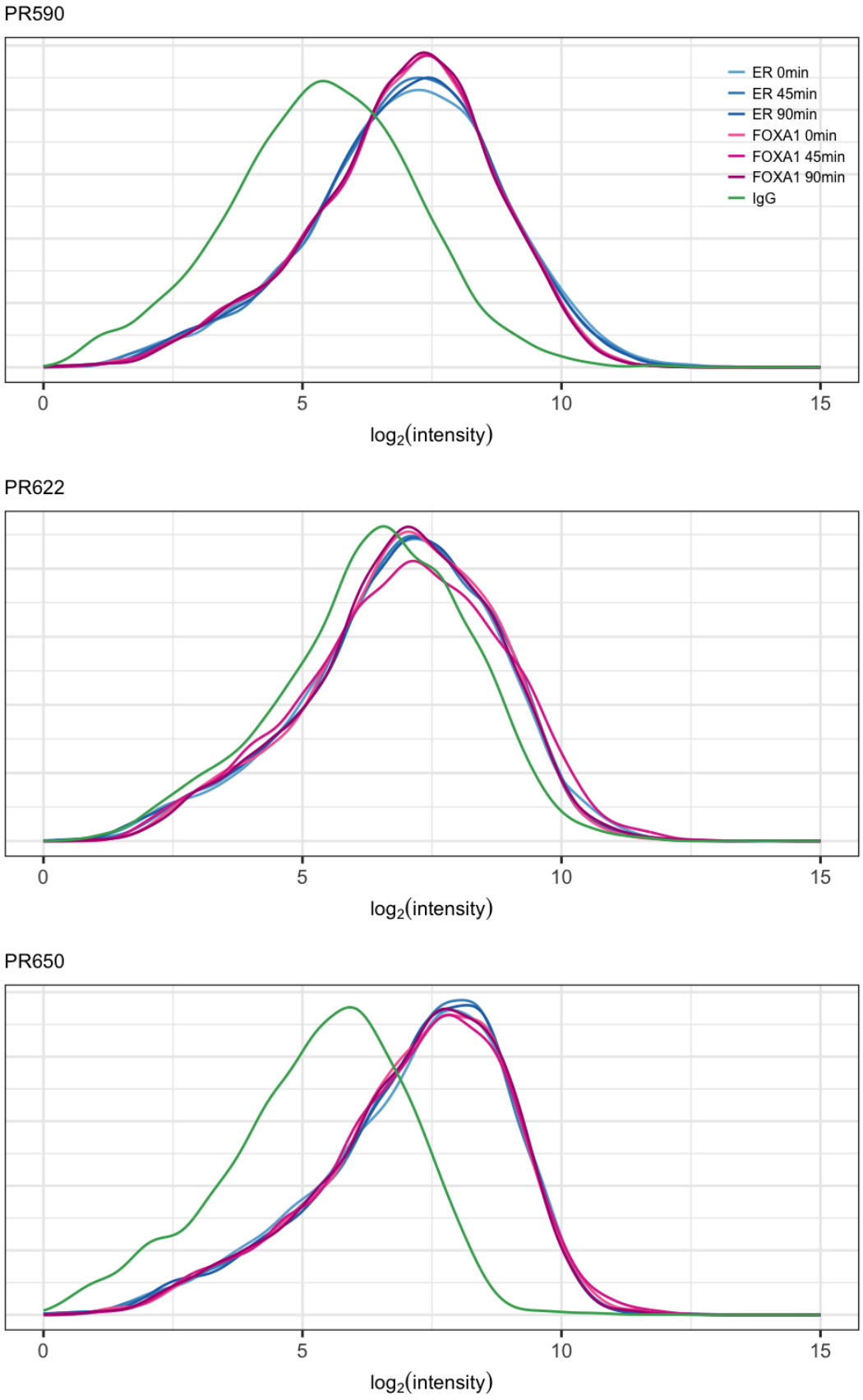


Figure 44: Density plots of normalized intensities from each TMT run where scale normalization was applied to peptide intensities for all samples except the IgG controls.

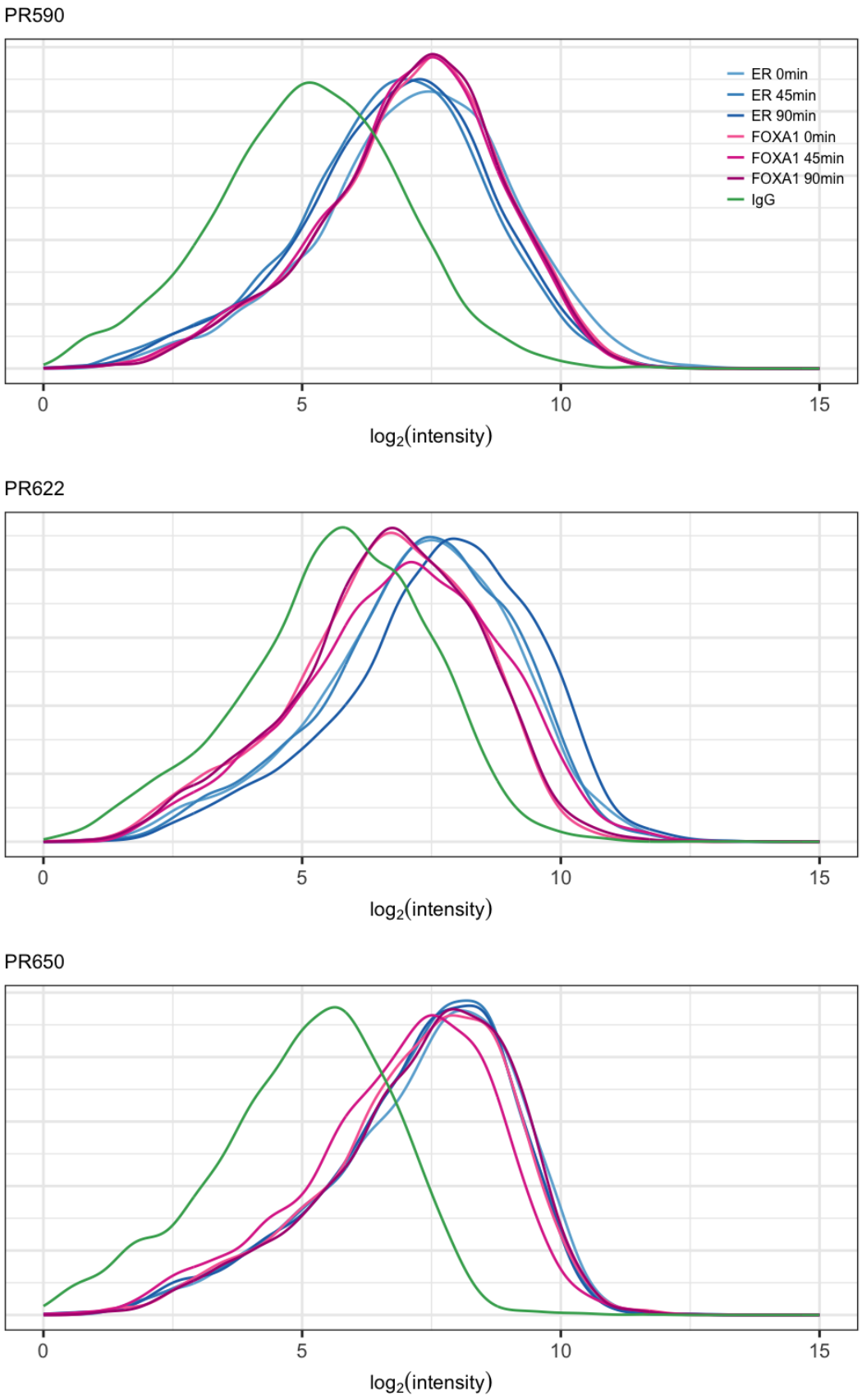


Figure 45: Density plots of normalized intensities from each TMT run where scale normalization is applied based on 10 peptides with the highest IgG intensities in each run.

levels of intensity values will have a greater weight in determining the overall intensity; higher intensity measurements are likely to more accurately reflect differences between samples so this may be advantageous.

When no imputation of missing values was carried out, peptides with missing values in one or more samples were excluded from the summation. Missing values that arise for technical reasons instead of being at undetectable levels in a sample would otherwise reduce the overall intensity in that sample relative to other samples.

The statistical analysis of differential binding between groups that follows requires at least 3 observations per group. Table 3 gives the number of protein identified in just a single run, two runs or all three runs. A total of proteins were identified across all runs of which 527 were observed in all three runs, allowing for a statistical analysis. $\log_2$ fold changes are still computed for proteins identified only in one or two of the runs, but no measure of the statistical significance is given in the differential binding analysis.

Runs/replicates	1	2	3	total
Including missing values	625	401	527	1553
Excluding missing values	625	398	518	1541

Table 3: Numbers of proteins identified in differing numbers of runs and total number of proteins identified in all runs.

4.5 Principal Component Analysis

Protein-level intensities were scaled to sum to 1.0 for each protein within a run, allowing for comparison across runs in a principal component analysis (PCA).

Figure 6 shows a PCA plot for the first two principal components using all proteins that were sampled in all three runs. The plot shows a clear separation of the ER, FOXA1 and IgG control pull-downs. The 0, 45 and 90 minute timepoints within each of the ER and FOXA1 groups are not completely separated in the first two principal components. Figure 7 shows the PCA plot for just the ER samples.

4.6 Differential Binding

A statistical analysis of differentially-expressed peptides was carried out using **limma**, a R/Bioconductor package commonly used in the analysis of microarray and RNA-seq data, but applicable to data from any quantitative expression technology.

Limma uses linear models to assess differential expression in the context of multifactor experimental designs. It is able to analyze comparisons between many RNA targets simultaneously (or in this case many proteins) and has features that make these analyses stable even for experiments with small numbers of samples; this is achieved by borrowing information across genes (proteins).

In this analysis, limma was used to estimate $\log_2$ fold changes and standard errors by fitting a linear model for each protein where the model includes variables for the group (the ER and FOXA1 pull-downs at 0, 45 and 90 minute time points and the IgG control at 45 minutes) and the run. This is essentially a two-way ANOVA, a generalization of a paired analysis, in which comparisons between pull-downs at different time points are made within each run.

Limma employs an empirical Bayes method to moderate the standard errors of the estimated $\log_2$ fold changes; this results in more stable inference and improved power, especially for experiments with small numbers of replicates.

Multiple testing correction of p-values was applied using the Benjamini-Hochberg method to control the false discovery rate (FDR). The adjusted p-value (also known as a q-value) can be understood as follows. If all proteins with q-value below a threshold of 0.05 are selected as differentially expressed, then the expected

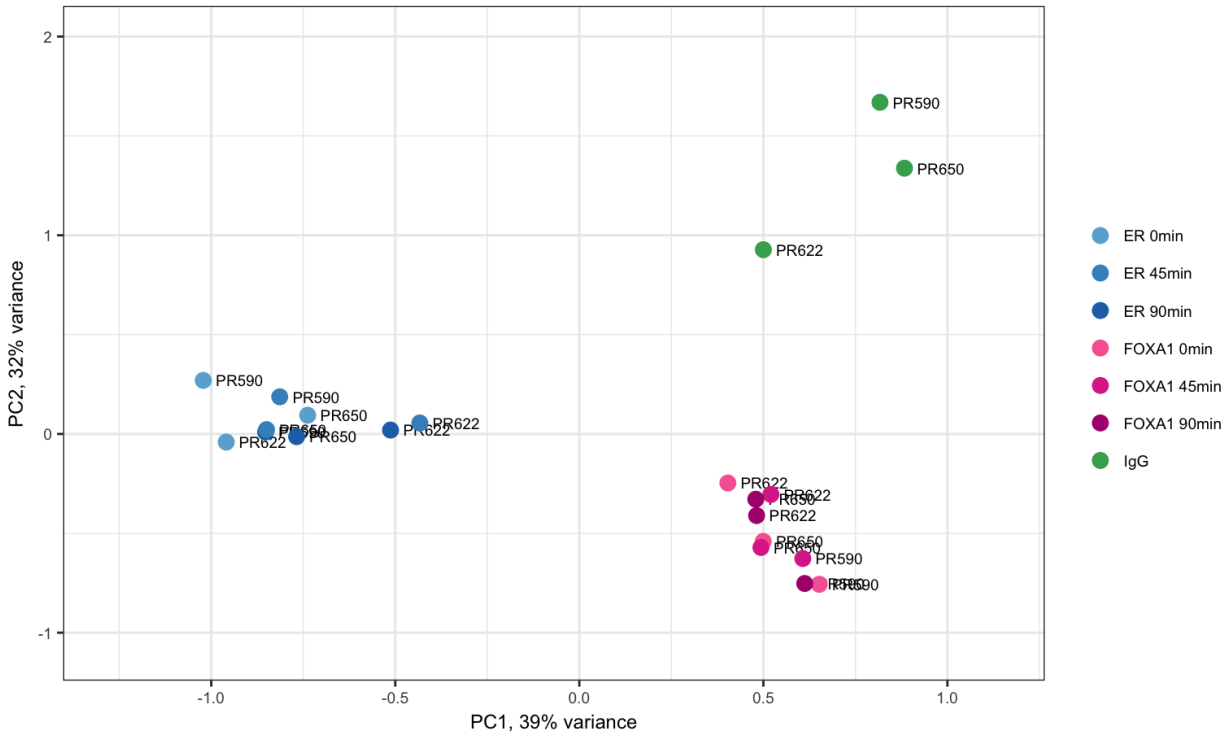


Figure 46: Principal Componenent Analysis for proteins sampled in all 3 runs. The PCA was based on protein-level data resulting from summation of quantile normalized peptide intensities in which missing values were imputed using KNN-based nearest neighbour averaging. The first two principal components are displayed.

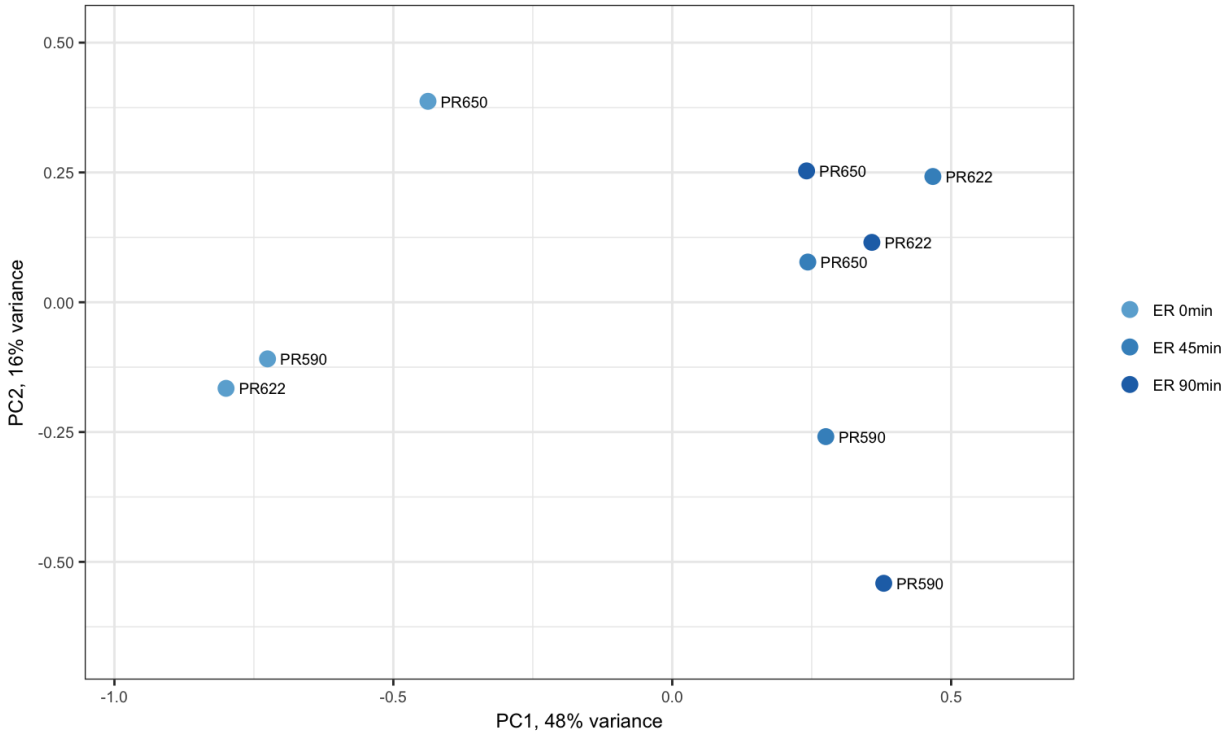


Figure 47: Principal Componenent Analysis for ER-interactng proteins sampled in all 3 runs. The first two principal components are displayed.

proportion of false discoveries in the selected group is controlled to be less than the threshold value, in this case 5%.

The B-statistic (lods or B) is the log odds that the protein is differentially expressed. For example, if B = 1.5, the odds of differential expression is $e^{1.5} = 4.48$, i.e, about four and a half to one. The probability that the protein is differentially expressed is $4.48/(1 + 4.48) = 0.82$. A B-statistic of zero corresponds to a 50-50 chance that the gene is differentially expressed. The B-statistic has been automatically adjusted for multiple testing by assuming that 1% of the proteins are expected to be differentially expressed.

The IgG control was used to assess non-specific binding. The binding of a protein is considered to be non-specific if the $\log_2$ fold change relative to the IgG control is less than 1. In the differential binding analysis, protein intensity values are fitted to a linear model and the fitted values for each condition being compared are also compared to the IgG control. For each comparison of two groups, the maximum $\log_2$ fold change from each of the two groups above the IgG control is used to determine if the binding is specific.

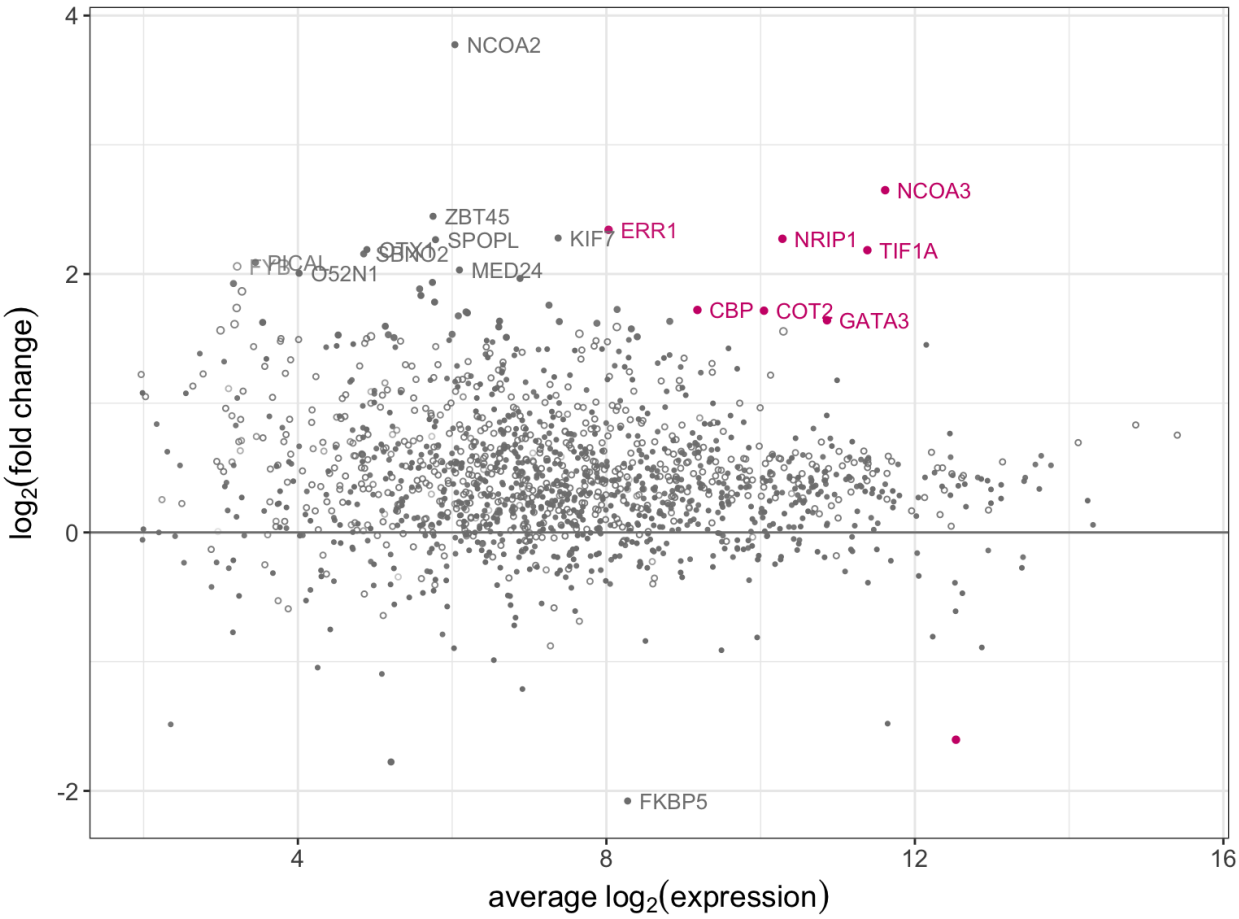


Figure 48: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, no normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.1 ER 45min vs ER 0min, excluding peptides with missing intensities, no normalization

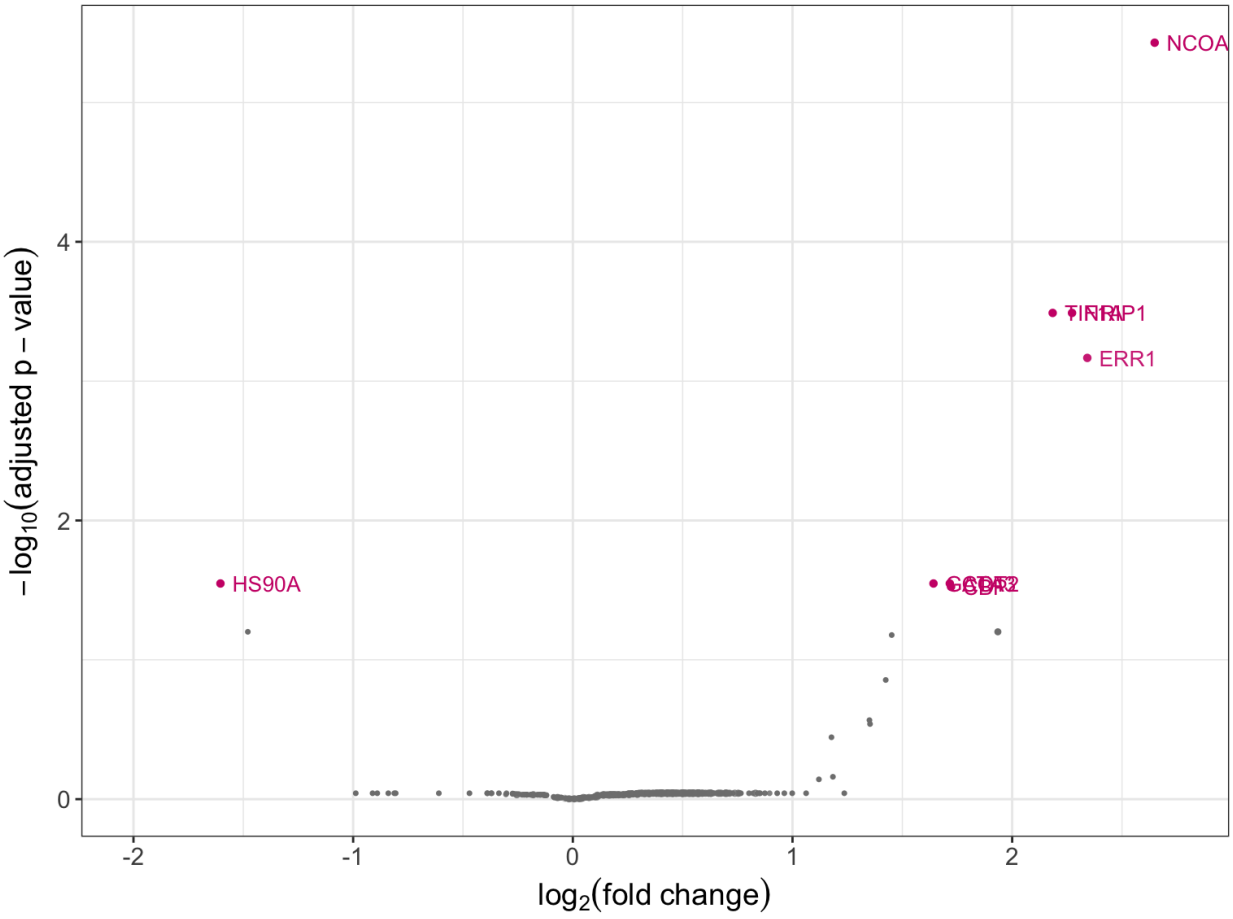


Figure 49: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, no normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

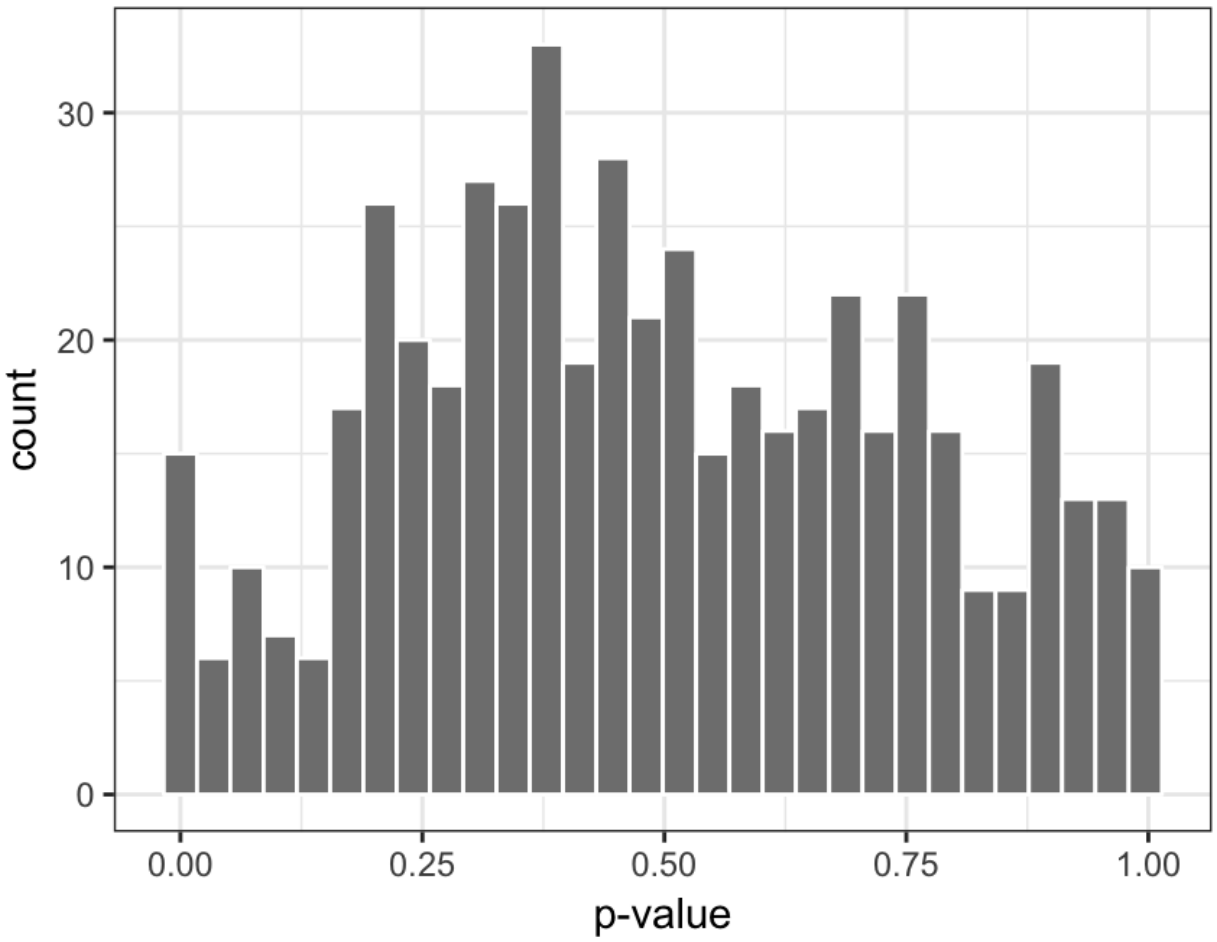


Figure 50: Histogram of p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, no normalization)

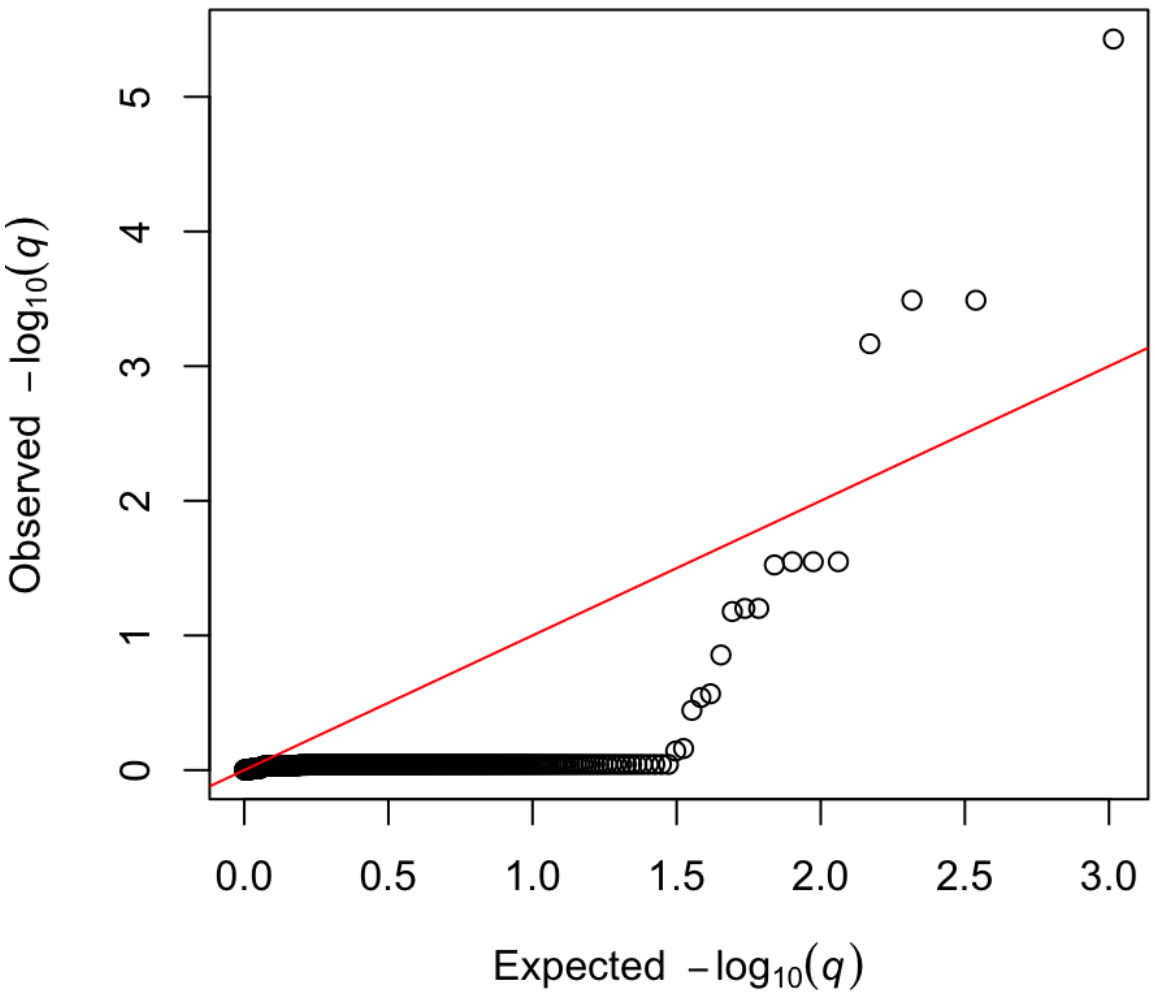


Figure 51: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, no normalization)

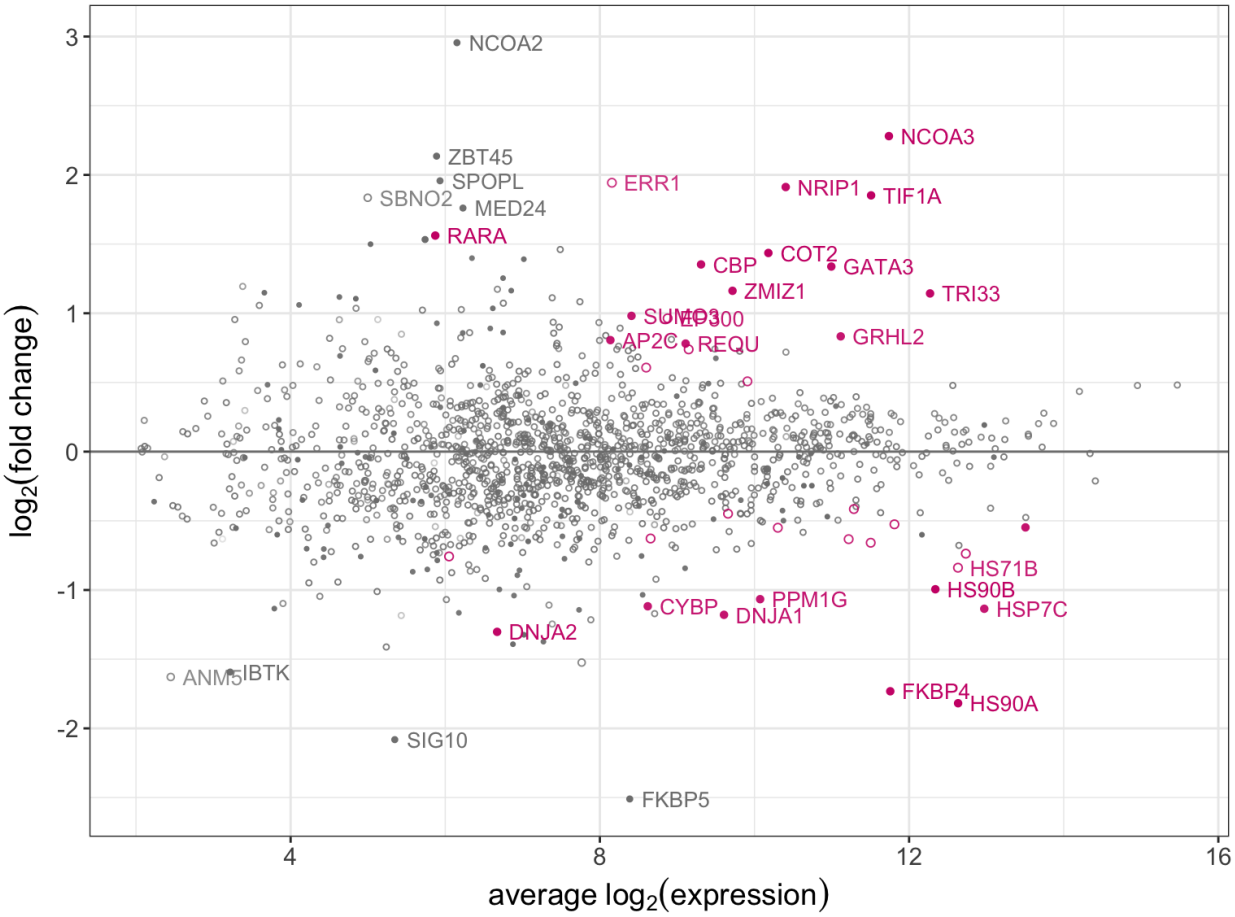


Figure 52: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.2 ER 45min vs ER 0min, excluding peptides with missing intensities, quantile normalization

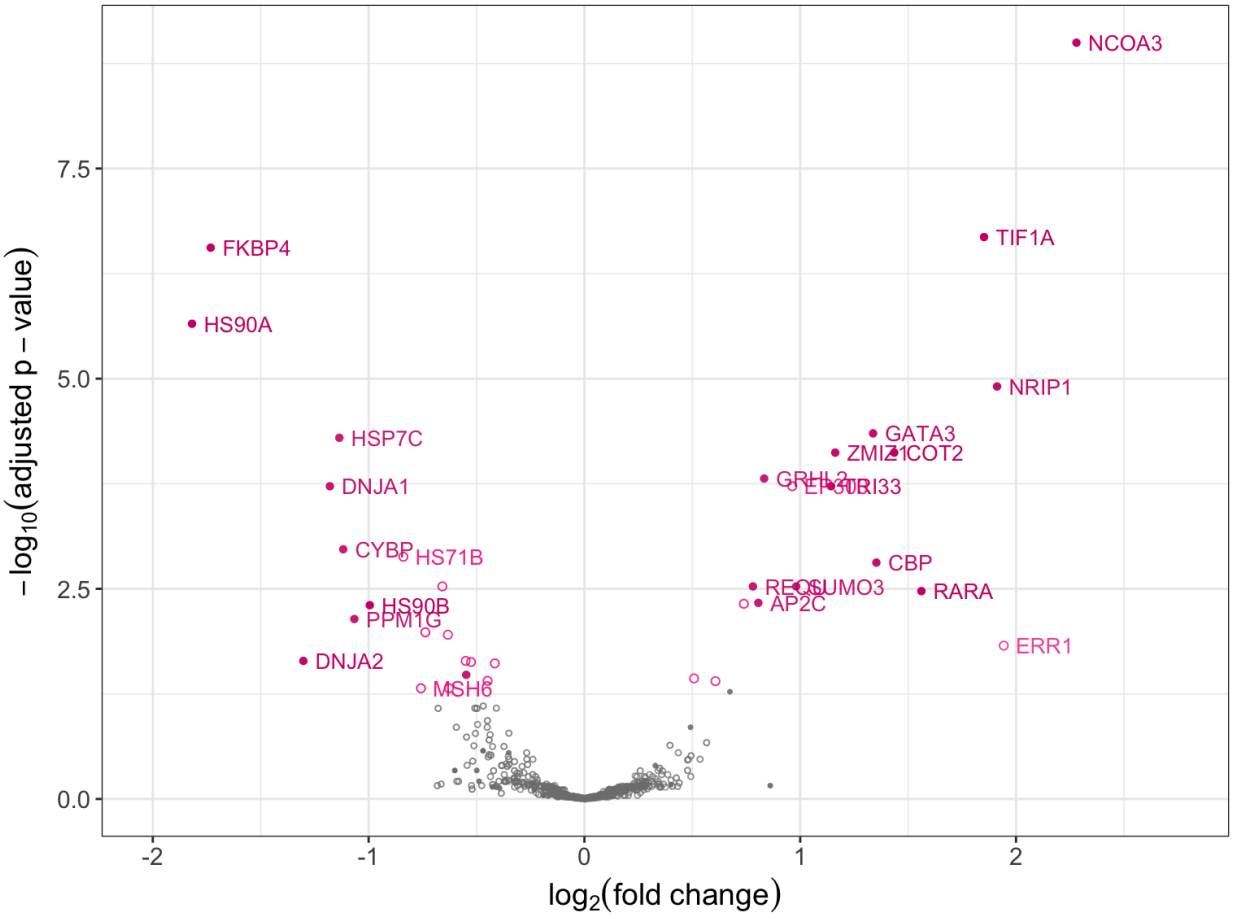


Figure 53: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

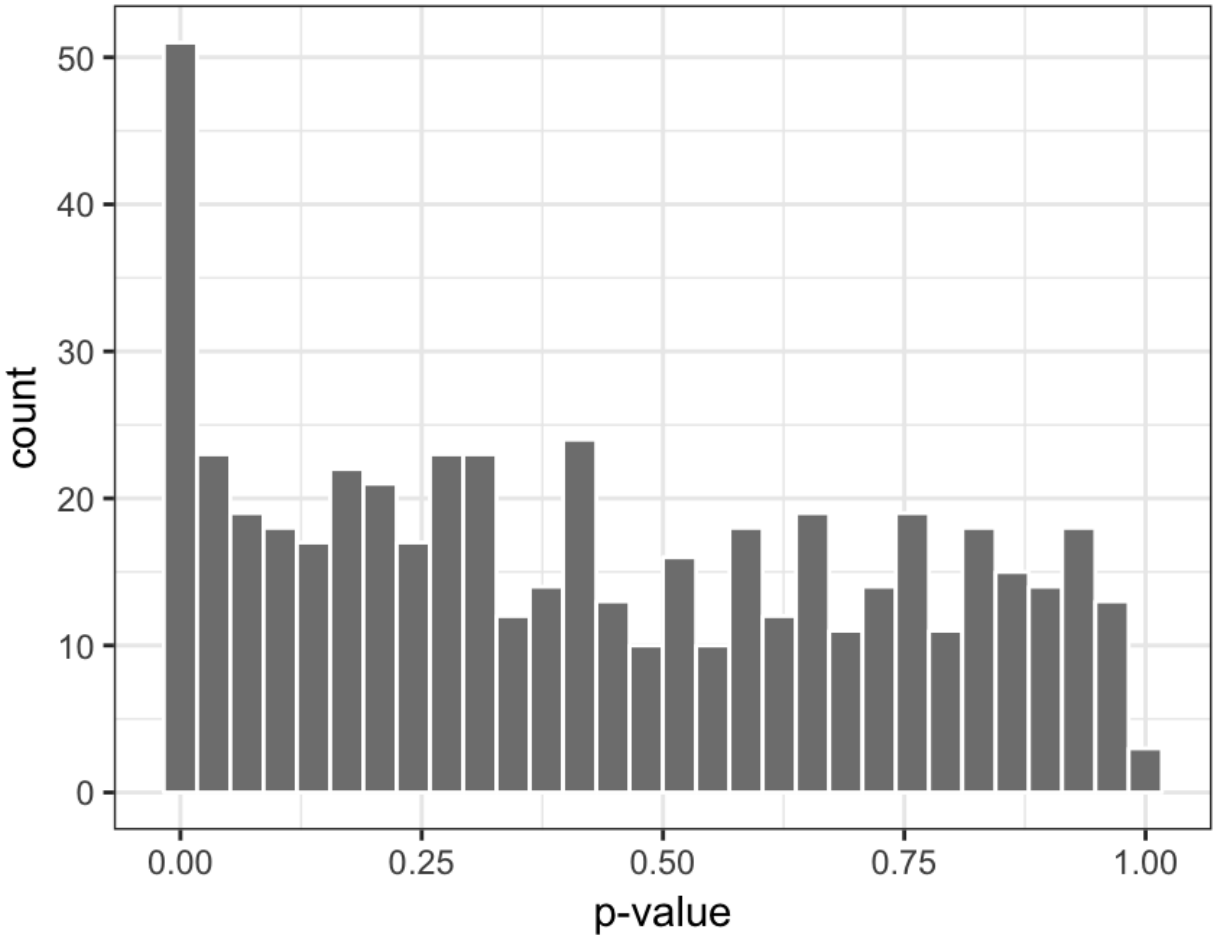


Figure 54: Histogram of p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization)

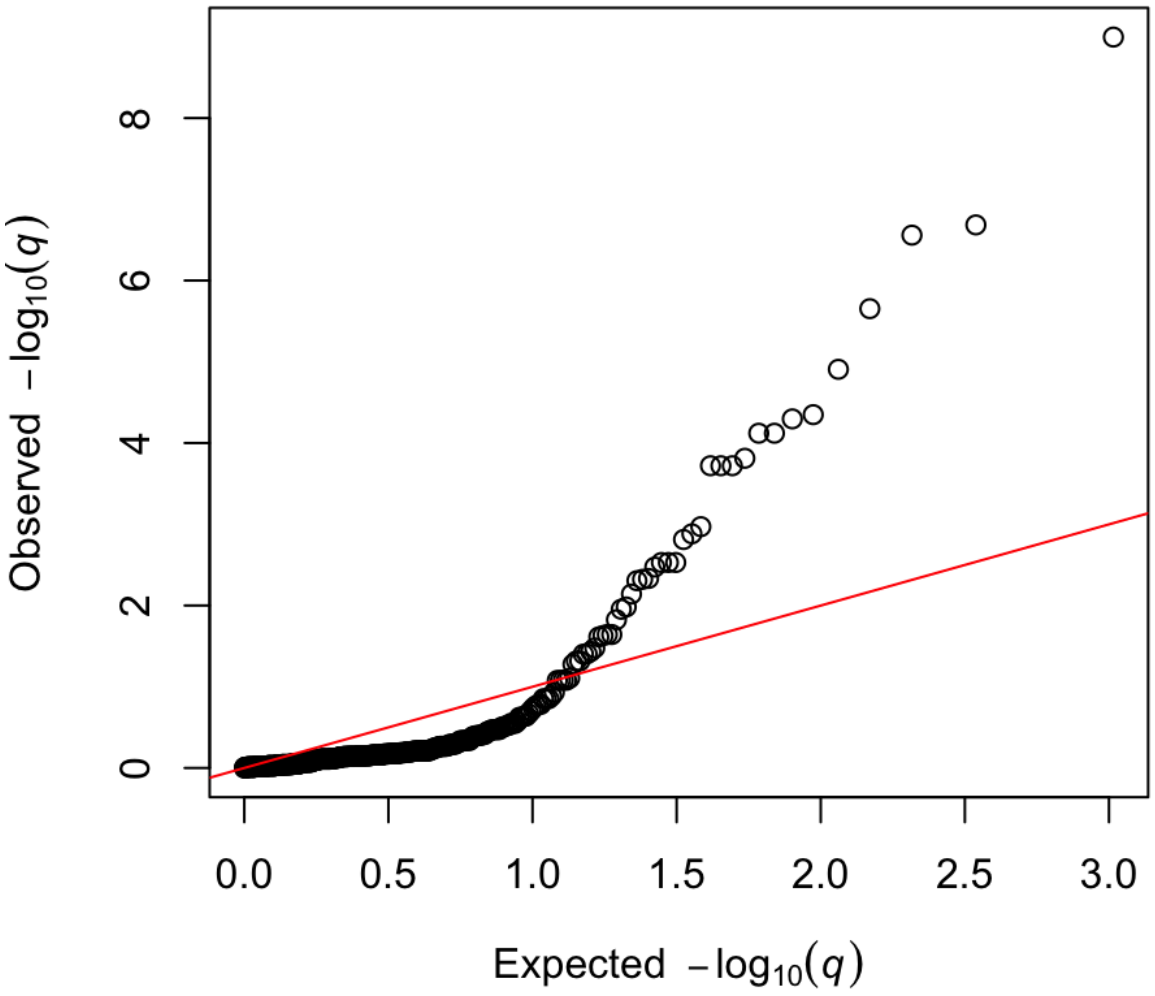


Figure 55: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization)

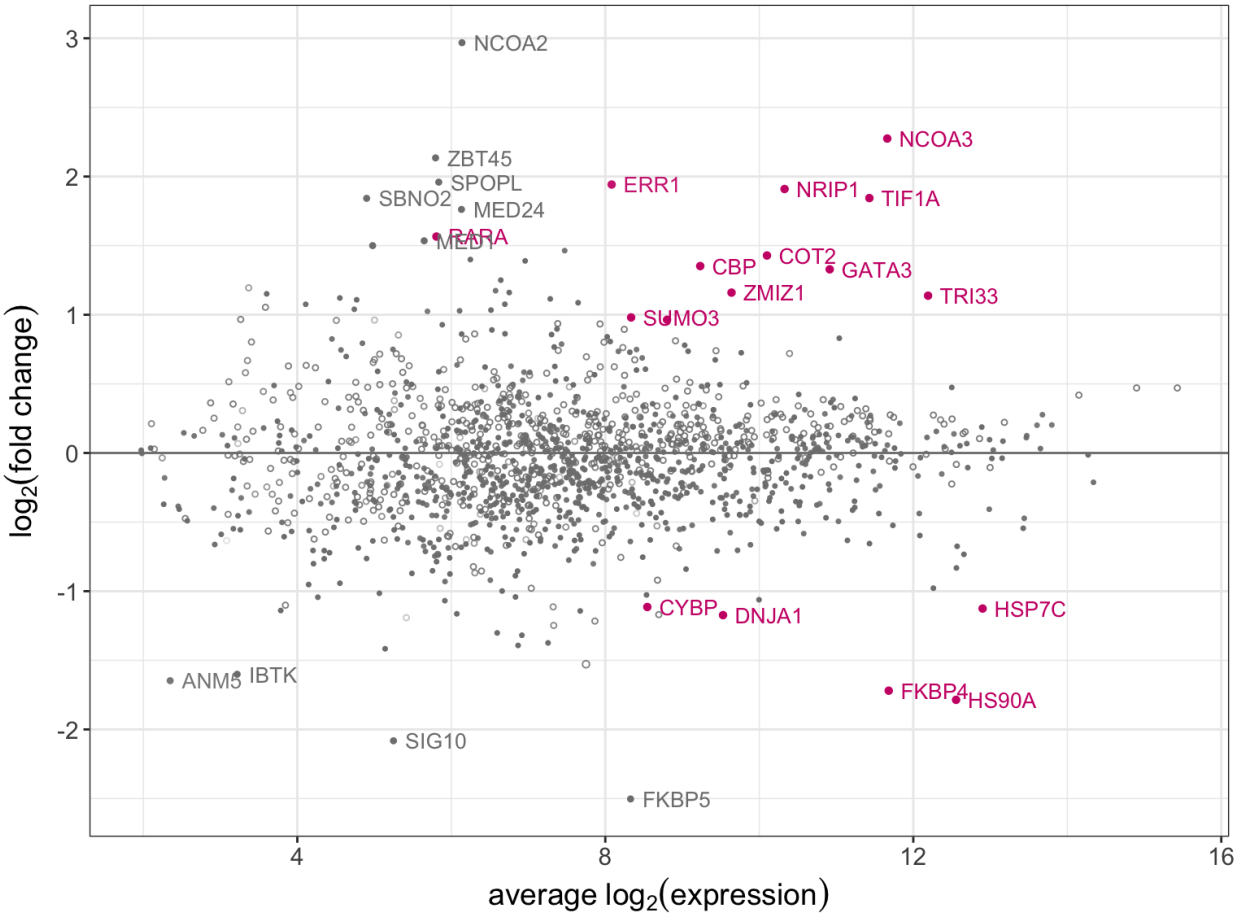


Figure 56: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization excluding IgG control). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.3 ER 45min vs ER 0min, excluding peptides with missing intensities, quantile normalization excluding IgG control

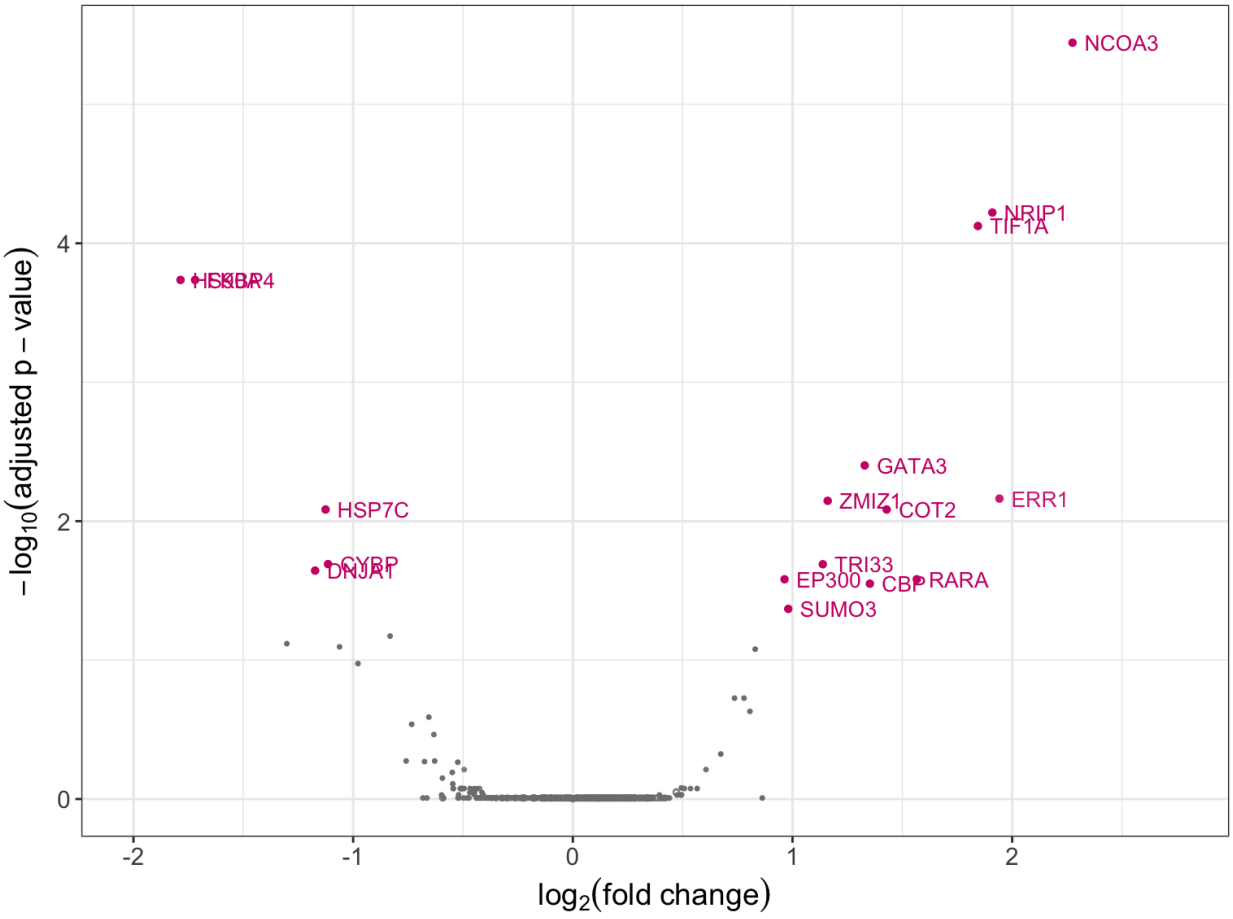


Figure 57: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization excluding IgG control). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

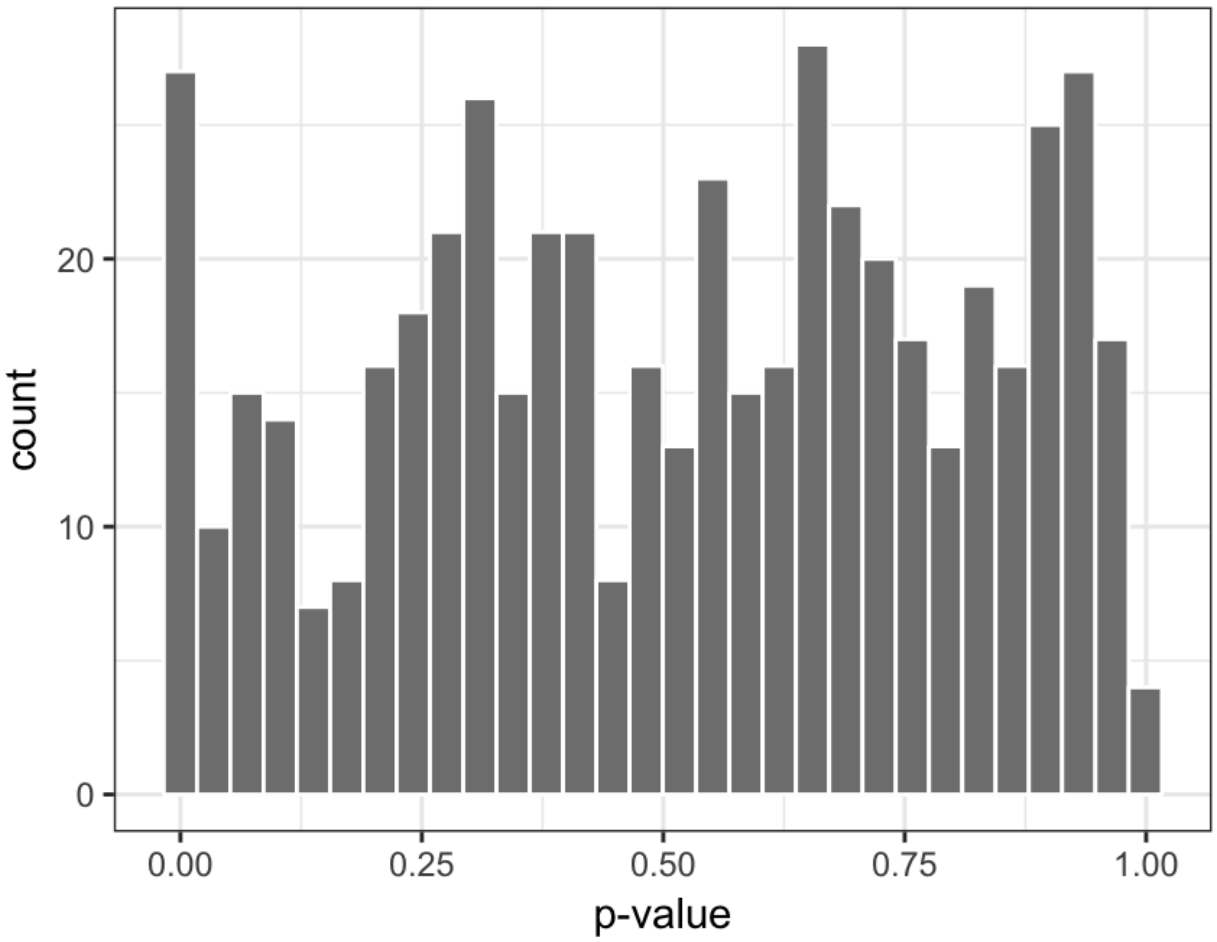


Figure 58: Histogram of p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization excluding IgG control)

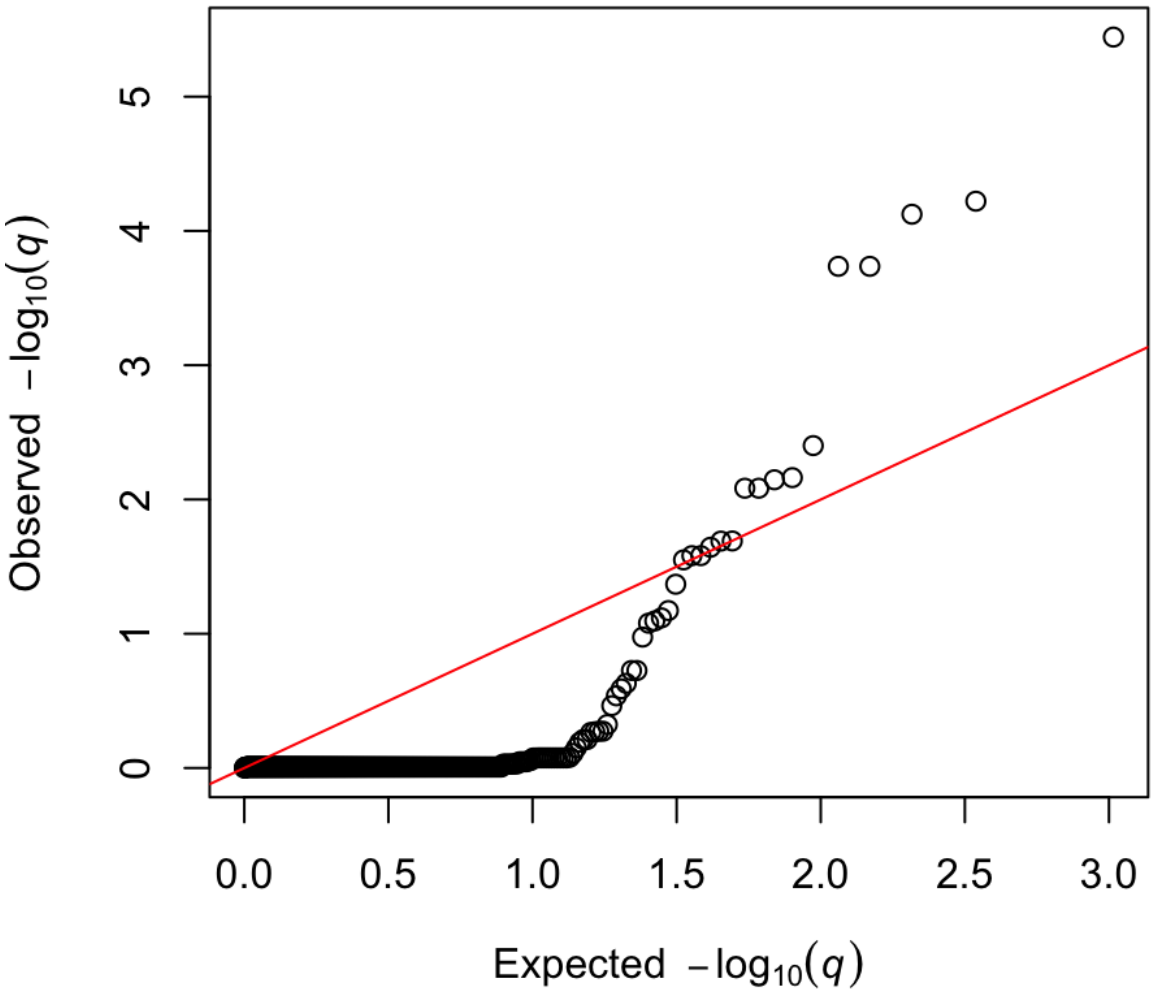


Figure 59: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, quantile normalization excluding IgG control)

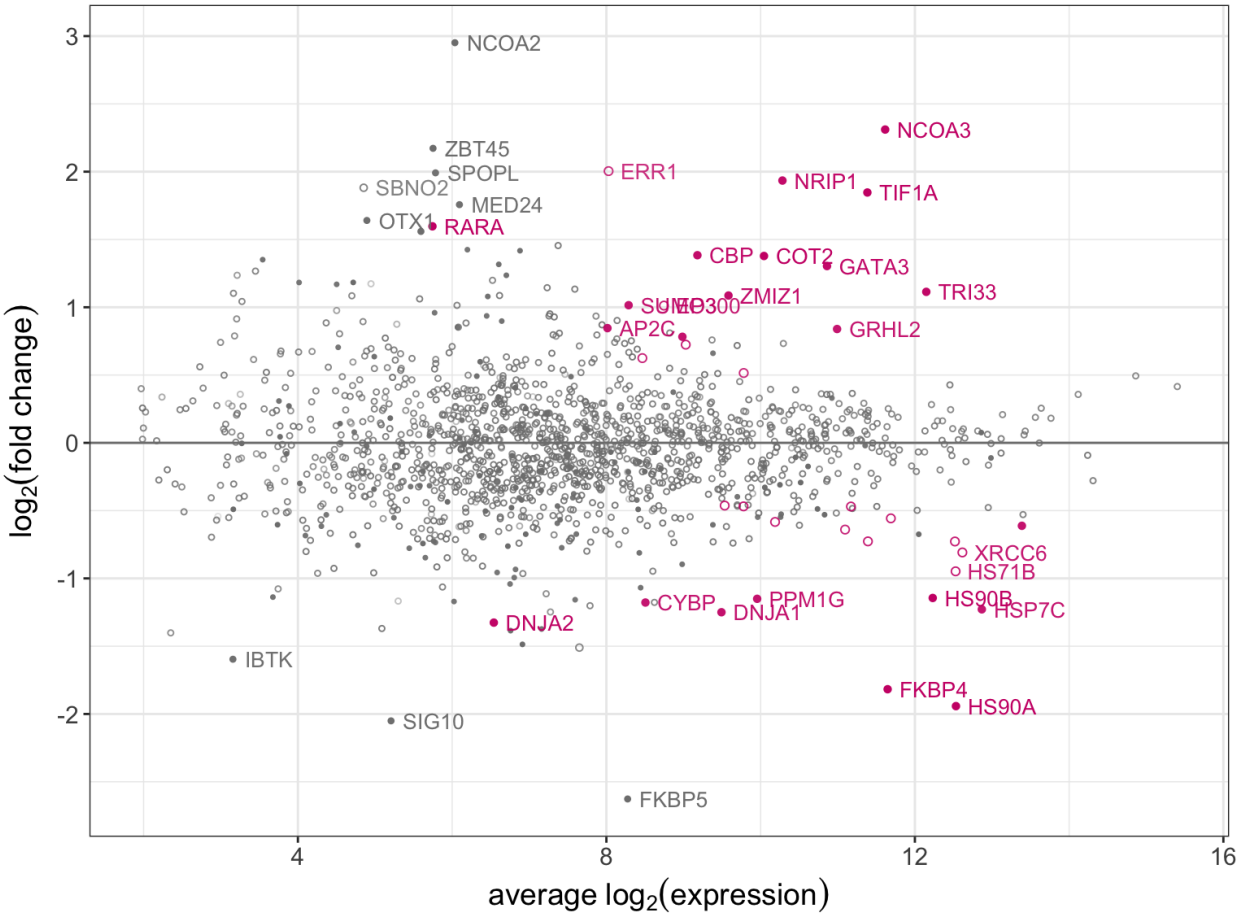


Figure 60: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.4 ER 45min vs ER 0min, excluding peptides with missing intensities, scale normalization

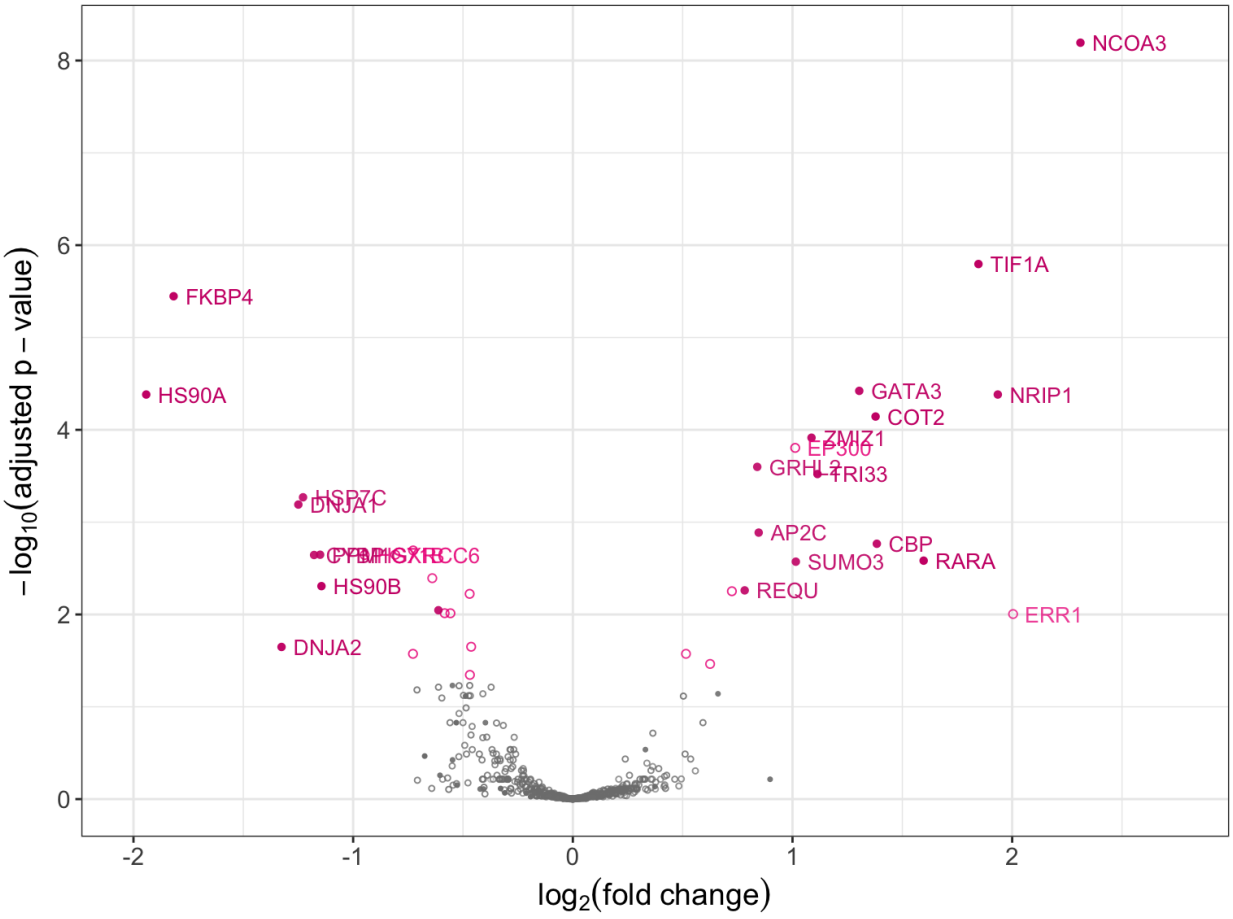


Figure 61: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

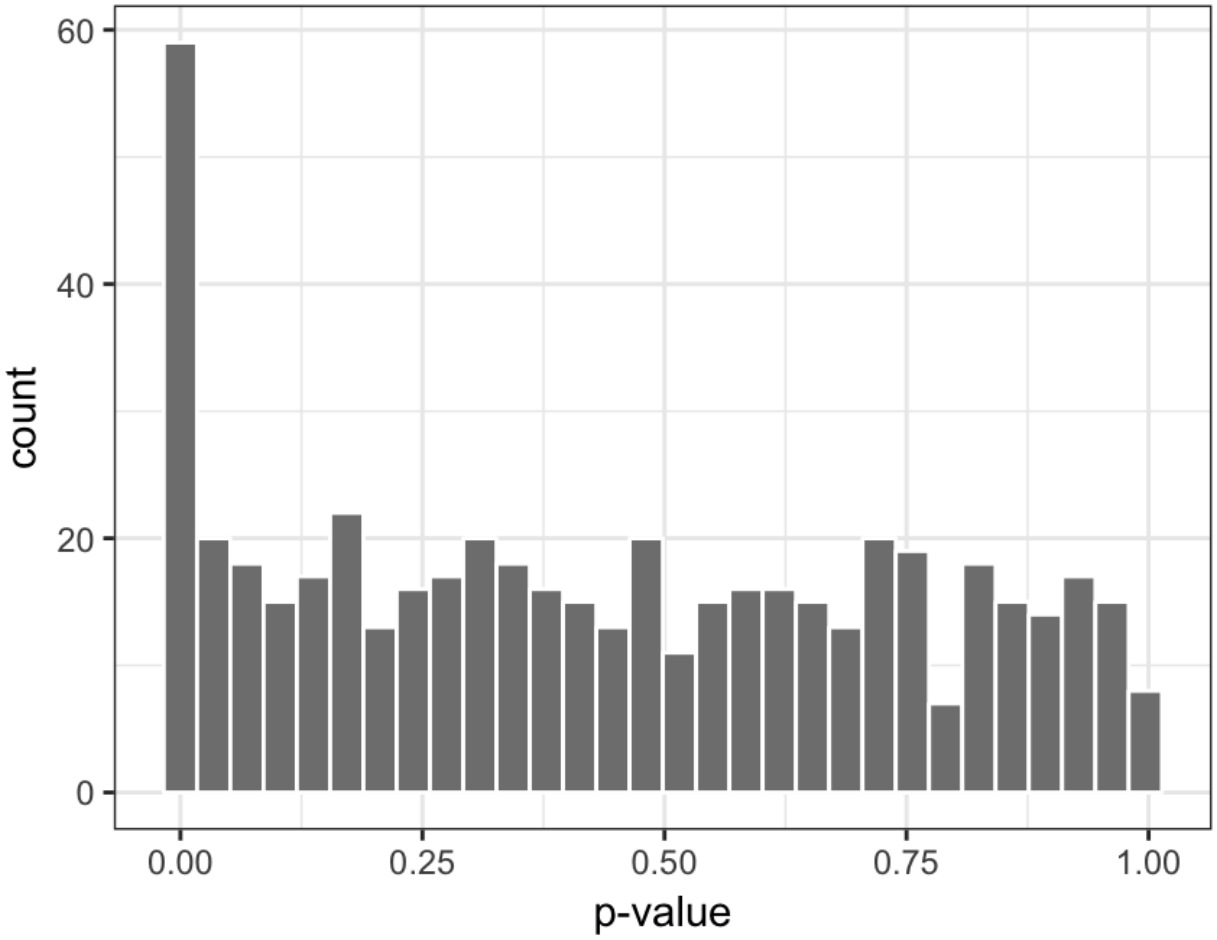


Figure 62: Histogram of p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization)

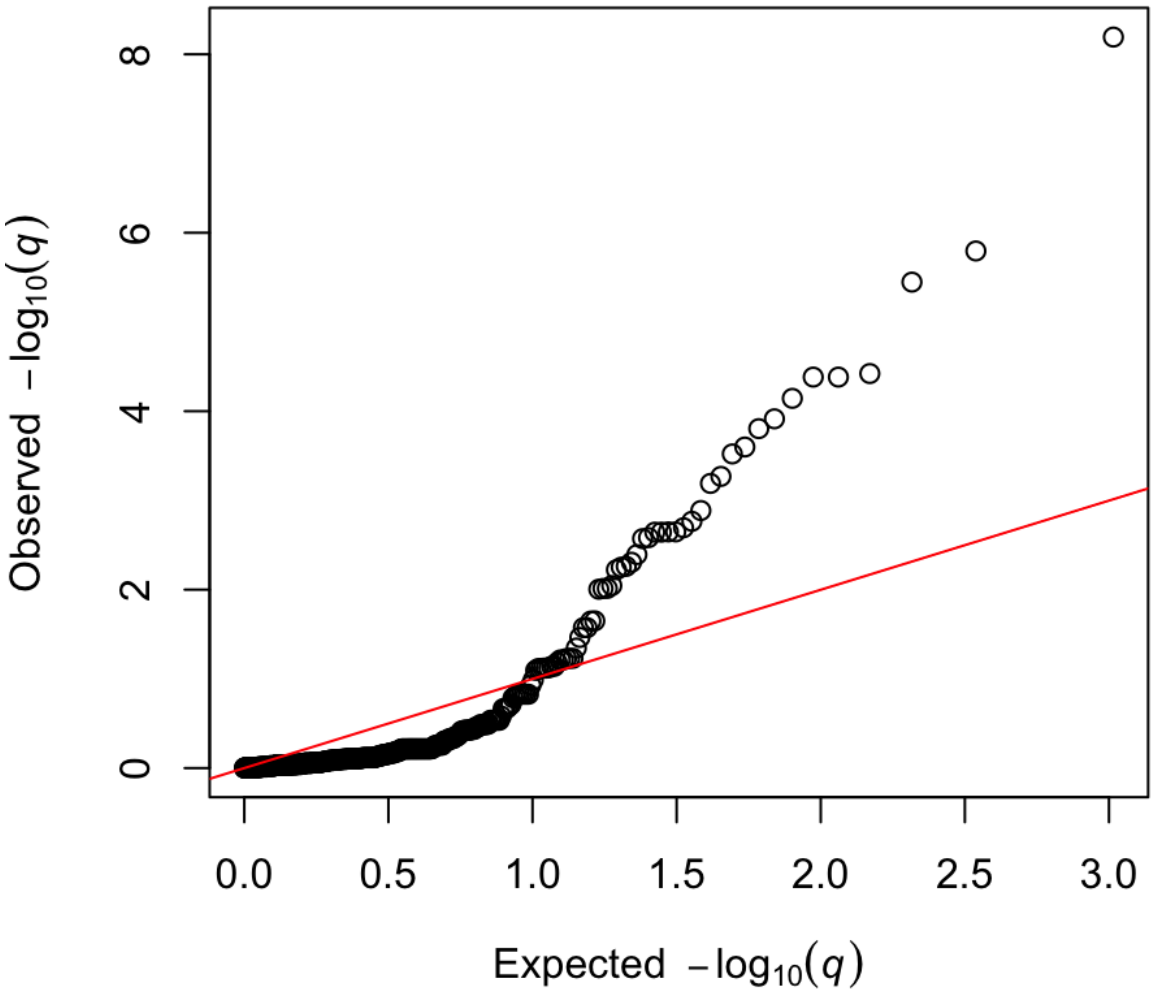


Figure 63: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization)

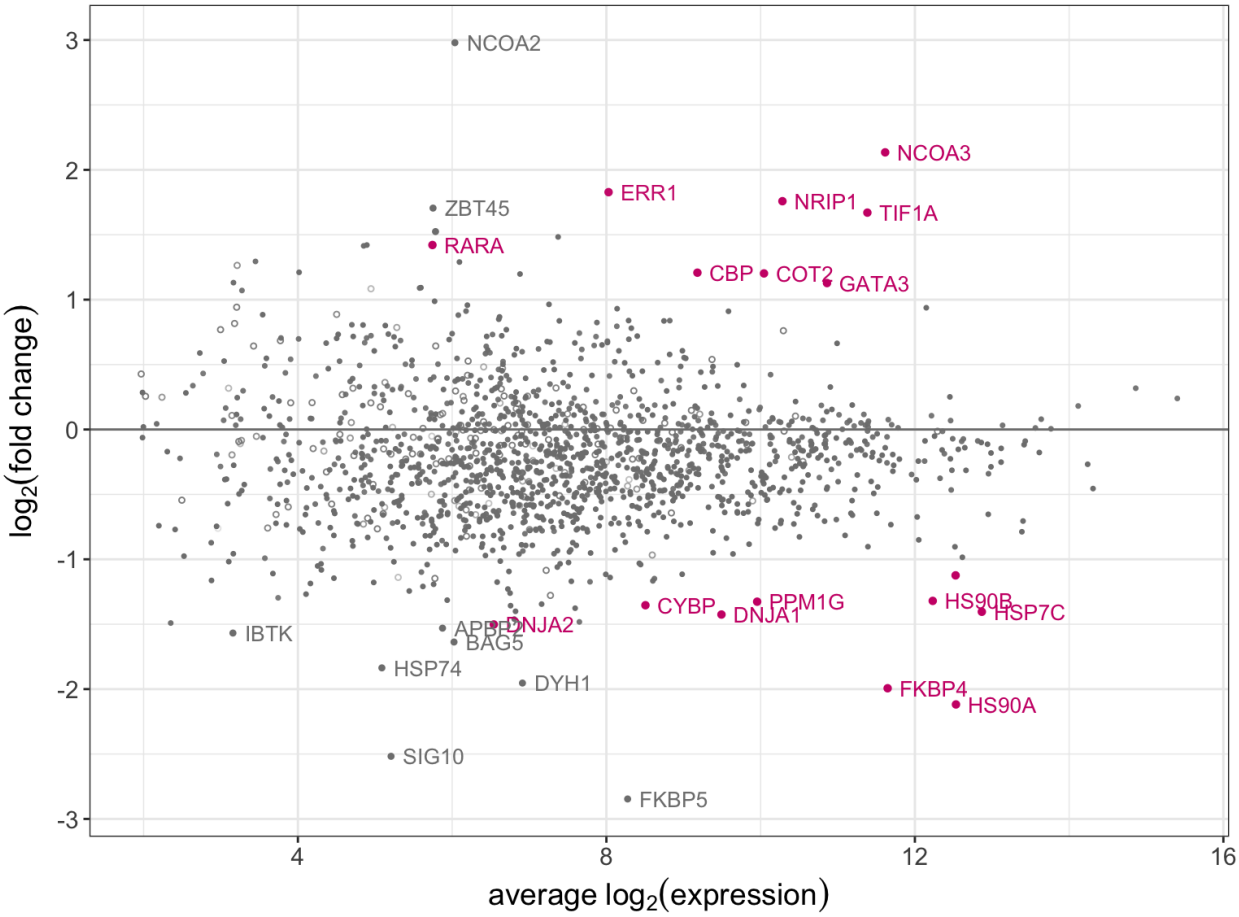


Figure 64: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization using top 10 IgG peptides). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.5 ER 45min vs ER 0min, excluding peptides with missing intensities, scale normalization using top 10 IgG peptides

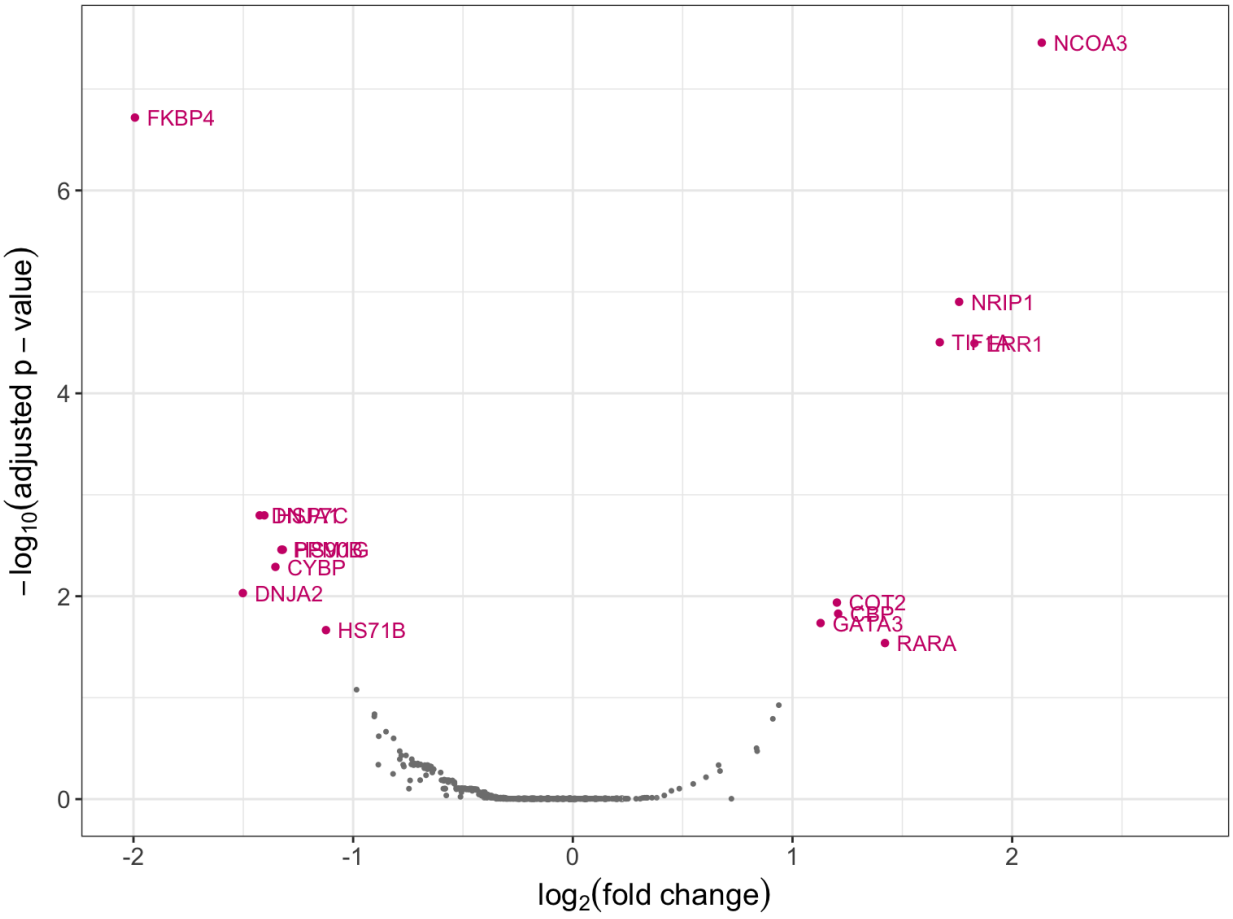


Figure 65: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization using top 10 IgG peptides). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

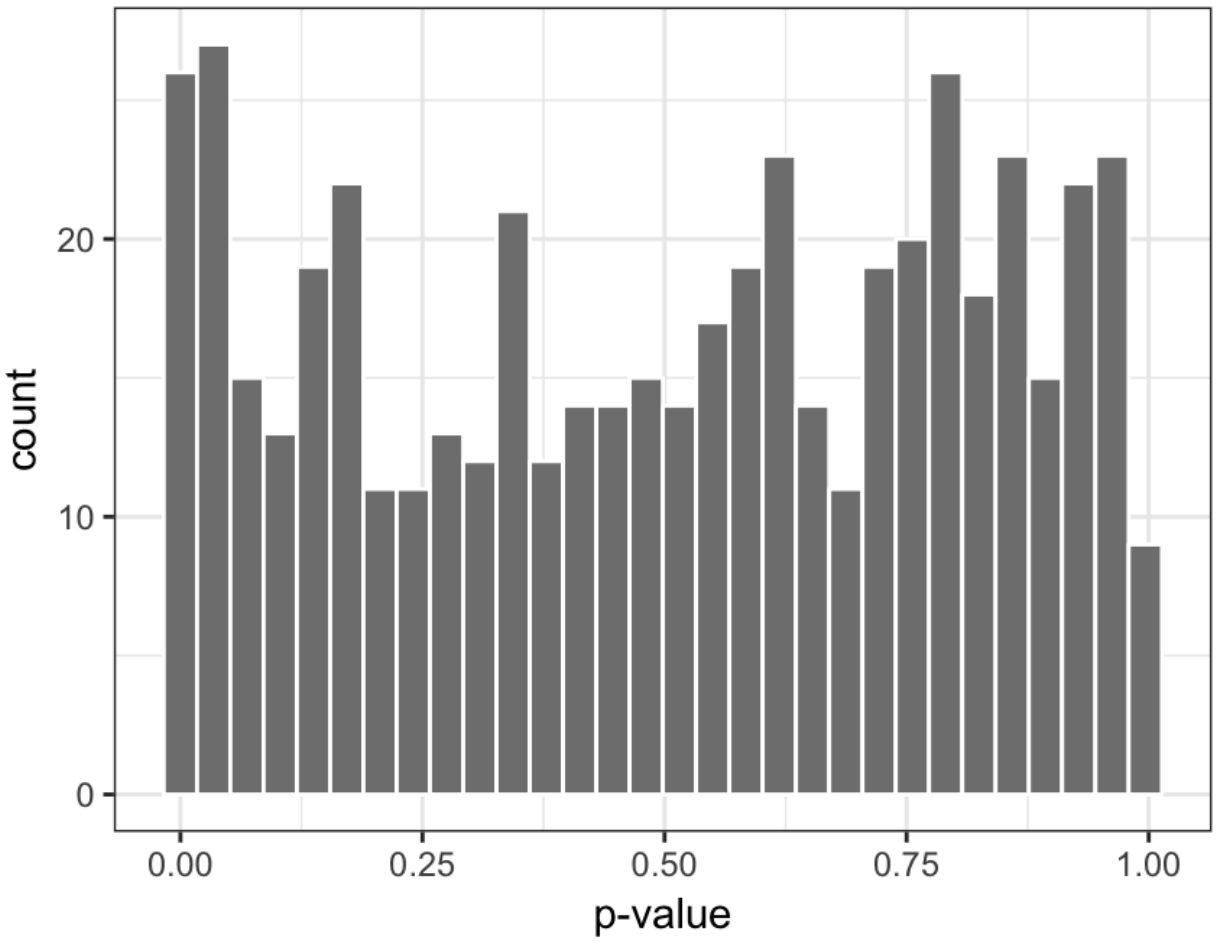


Figure 66: Histogram of p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization using top 10 IgG peptides)

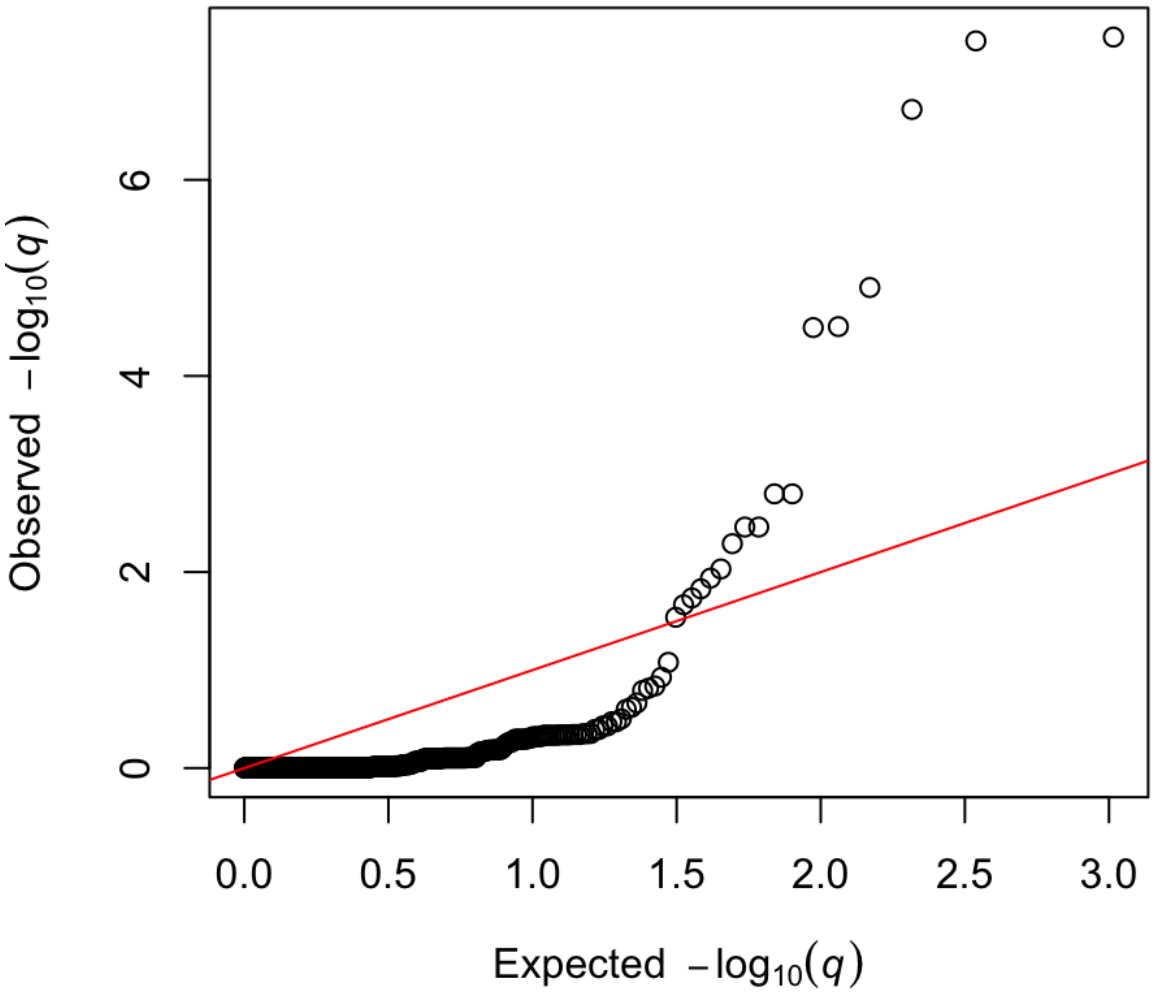


Figure 67: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (excluding peptides with missing intensities, scale normalization using top 10 IgG peptides)

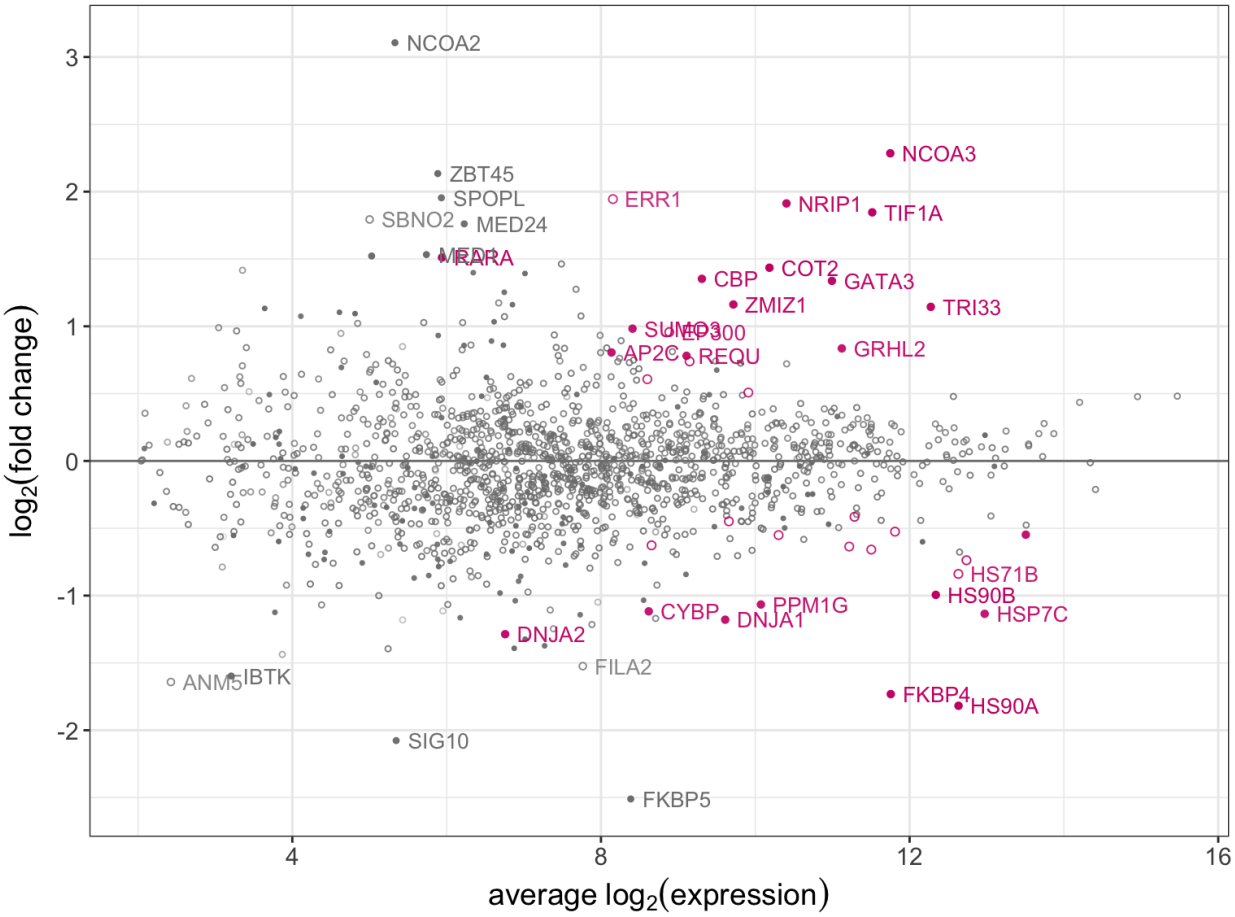


Figure 68: MA plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (KNN nearest neighbour averaging imputation of missing values, quantile normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

4.6.6 ER 45min vs ER 0min, KNN nearest neighbour averaging imputation of missing values, quantile normalization

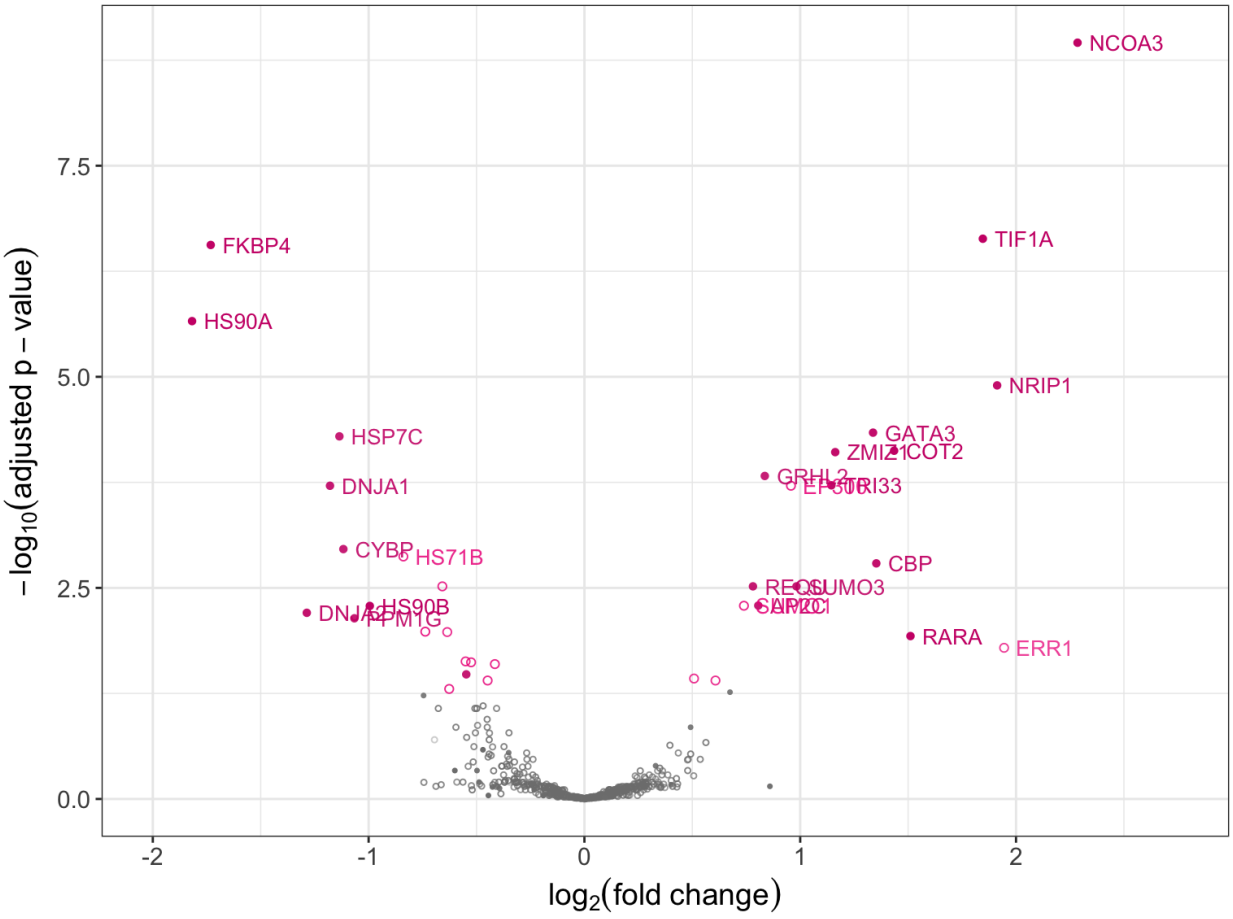


Figure 69: Volcano plot of the average intensity against log 2 fold change for the ER 45min vs ER 0min comparison (KNN nearest neighbour averaging imputation of missing values, quantile normalization). Top ranking differentially-expressed proteins with false discovery rate below 0.05 are highlighted in pink. Open circles indicate that the protein is non-specific from the IgG control comparison.

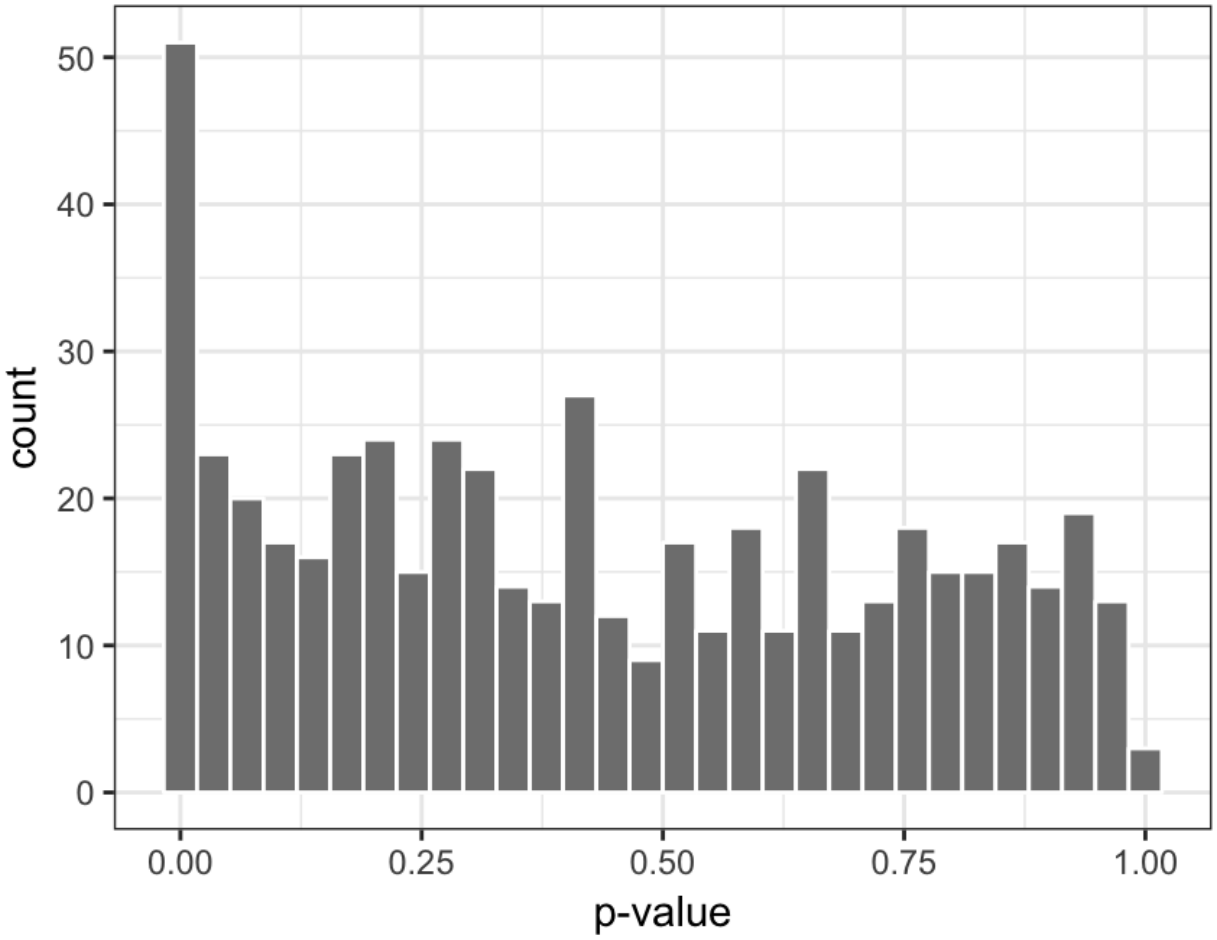


Figure 70: Histogram of p-values for the ER 45min vs ER 0min comparison (KNN nearest neighbour averaging imputation of missing values, quantile normalization)

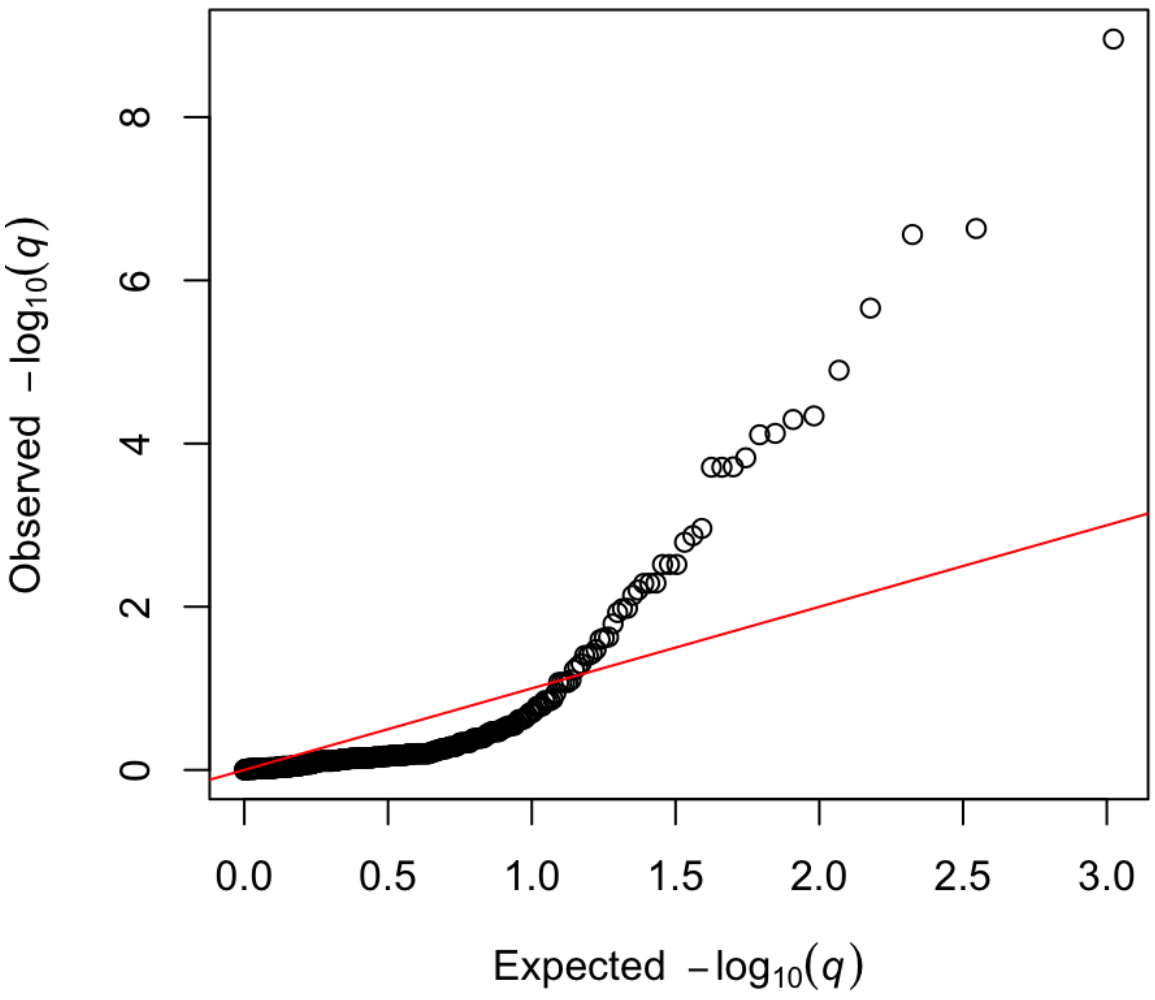


Figure 71: QQ plot of the adjusted p-values for the ER 45min vs ER 0min comparison (KNN nearest neighbour averaging imputation of missing values, quantile normalization)

4.7 Comparison of differential binding analysis approaches

In this section, comparisons of the differential binding results obtained using differing methods for handling missing values and varying normalization techniques are presented. The B-statistic (log odds) is used to rank the proteins from most significantly differentially expressed to least significant. Scatter plots of both the B-statistic generated for the two methods being compared and the rank are presented. The protein with the highest B-statistic, i.e. the most significantly differentially expressed protein, is given a rank of 1.

Differing imputation methods have relatively little effect on the results for most proteins. This is largely because only a small proportion of the proteins have peptide-level measurements containing missing values. There are a few proteins that are only represented in the differential binding results when imputation is carried out, for which all peptide-level measurements contain missing values. These proteins are not represented in the comparions with the analysis in which measurements with missing values are removed.

Different normalization approaches have a more substantial impact on the differential binding analysis.

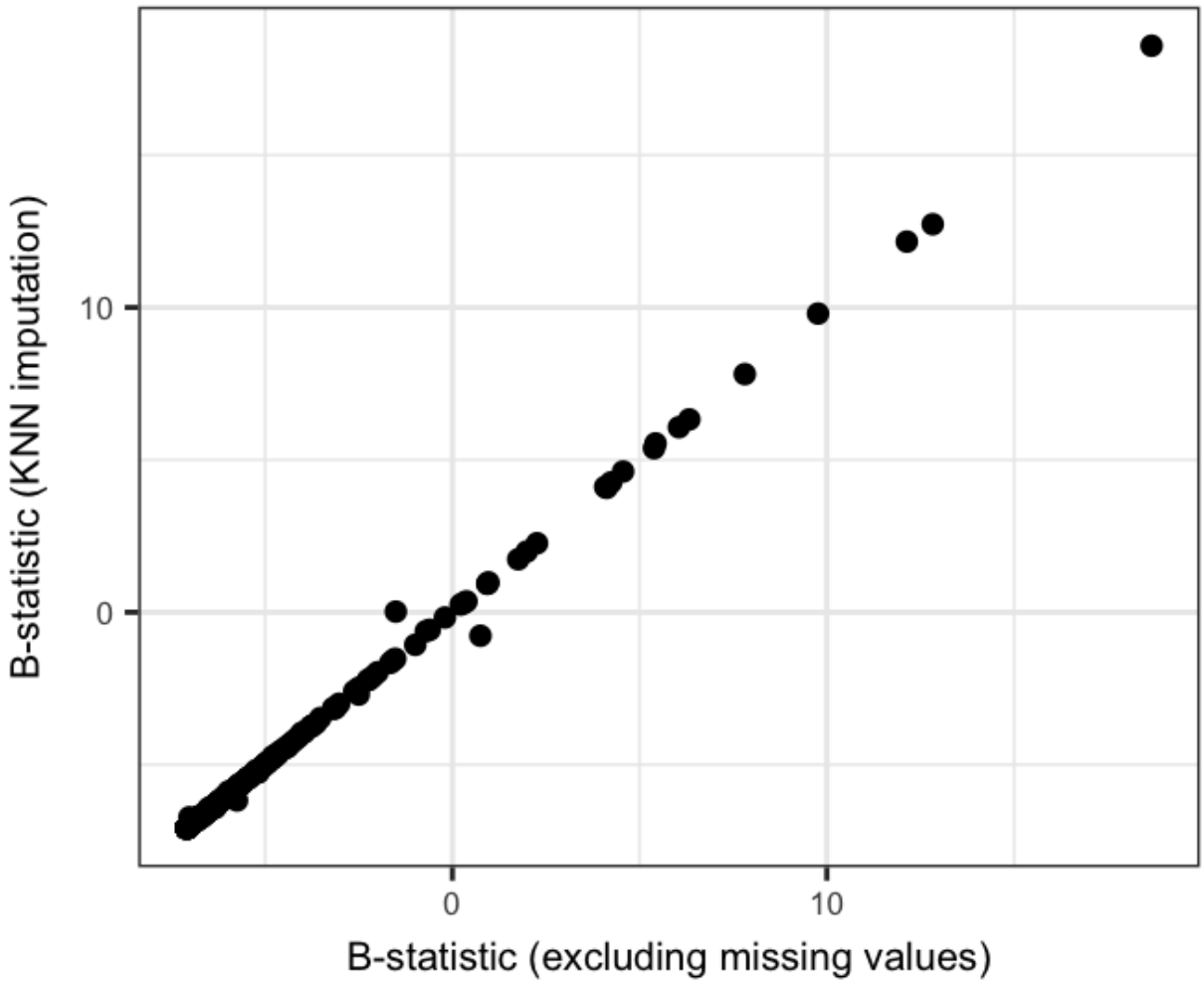


Figure 72: Plot of B-statistic (log odds) for differential expression of proteins for the ER 45min vs ER 0min contrast, comparing results following exclusion of missing values and KNN imputation.

4.7.1 Removal of missing values vs KNN imputation (quantile normalization in both)

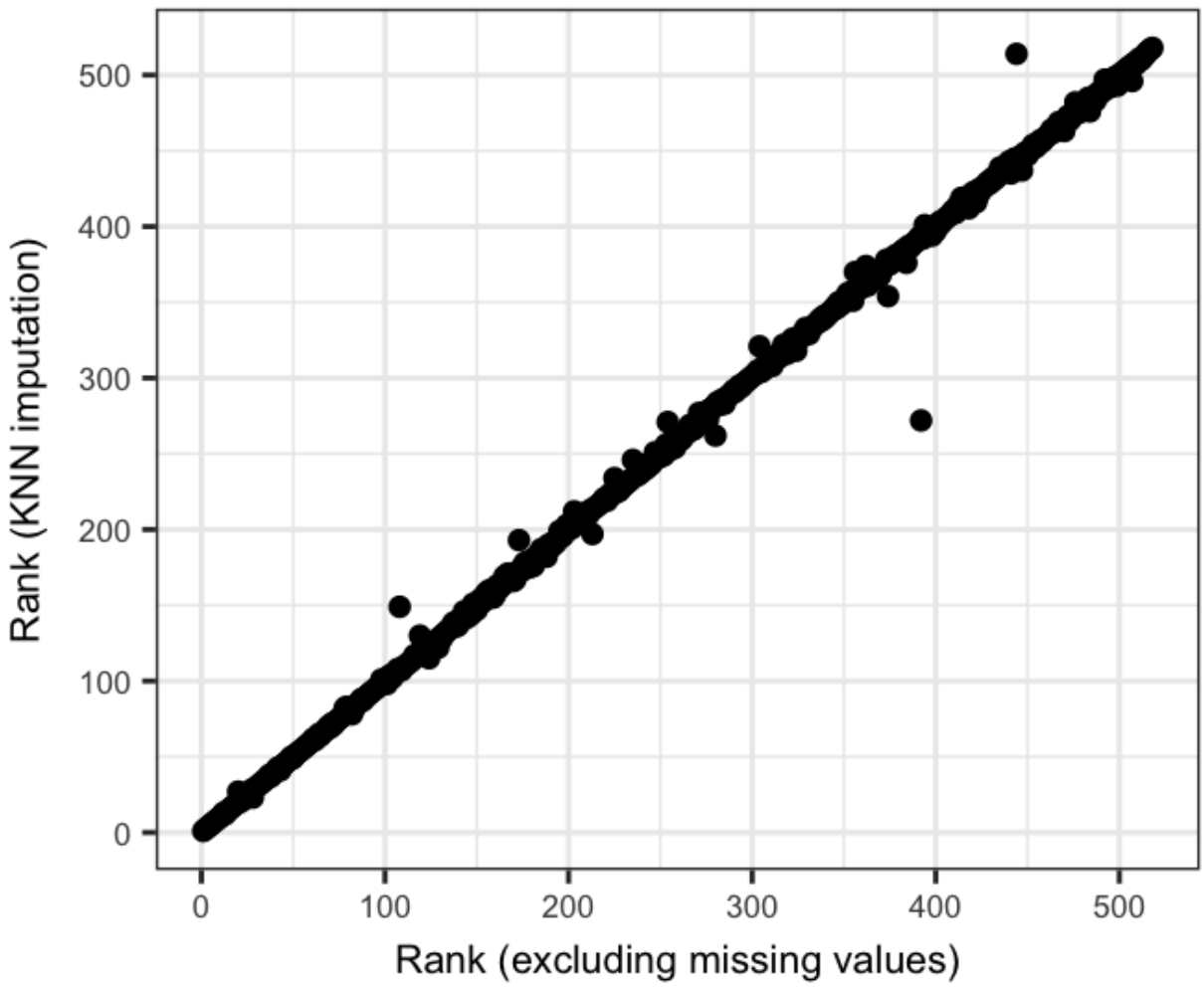


Figure 73: Plot of the rank of differentially-expressed proteins for the ER 45min vs ER 0min contrast, comparing results following exclusion of missing values and KNN imputation.

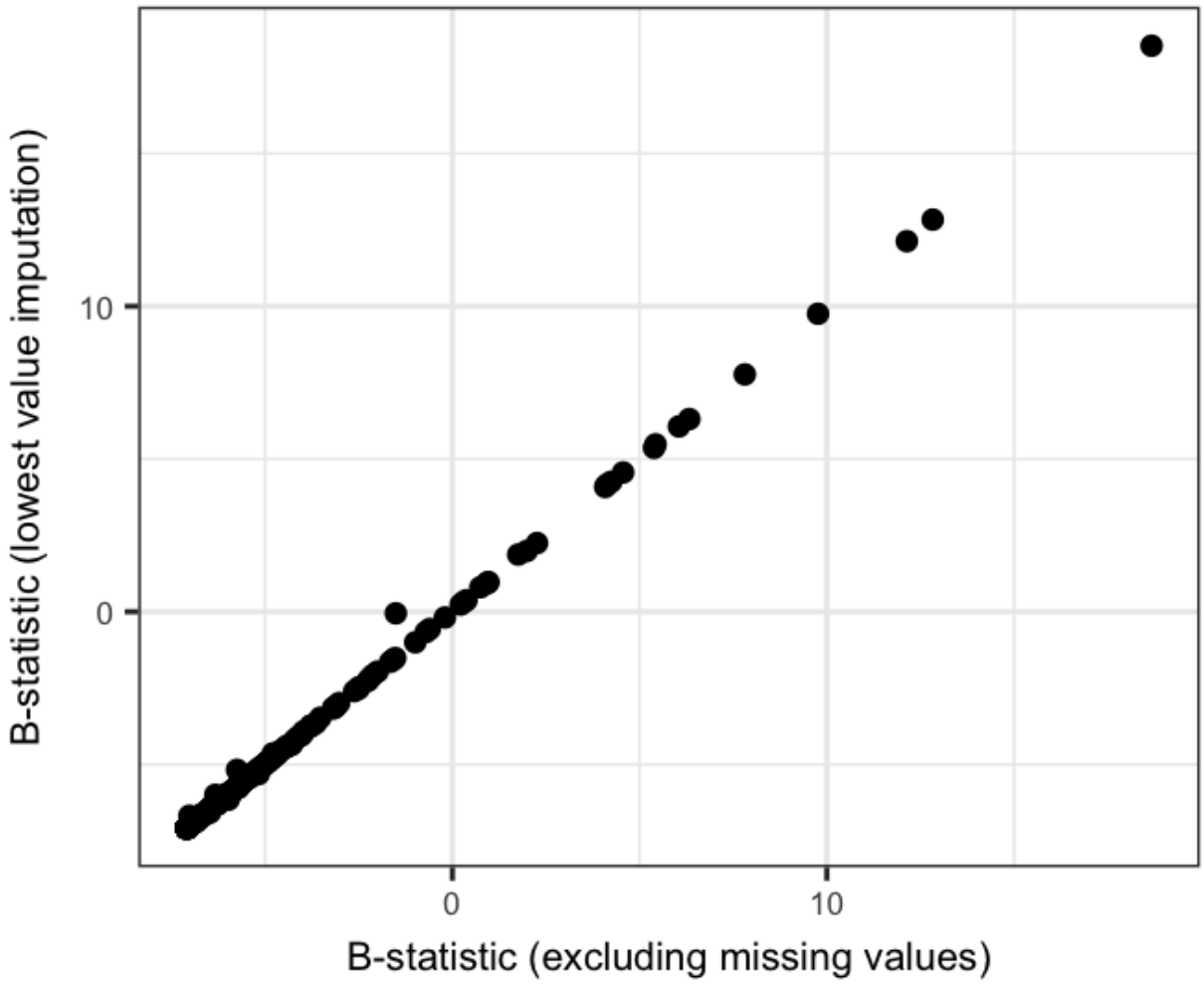


Figure 74: Plot of B-statistic (log odds) for differential expression of proteins for the ER 45min vs ER 0min contrast, comparing results following exclusion of missing values and lowest value imputation.

4.7.2 Removal of missing values vs lowest value imputation (quantile normalization in both)

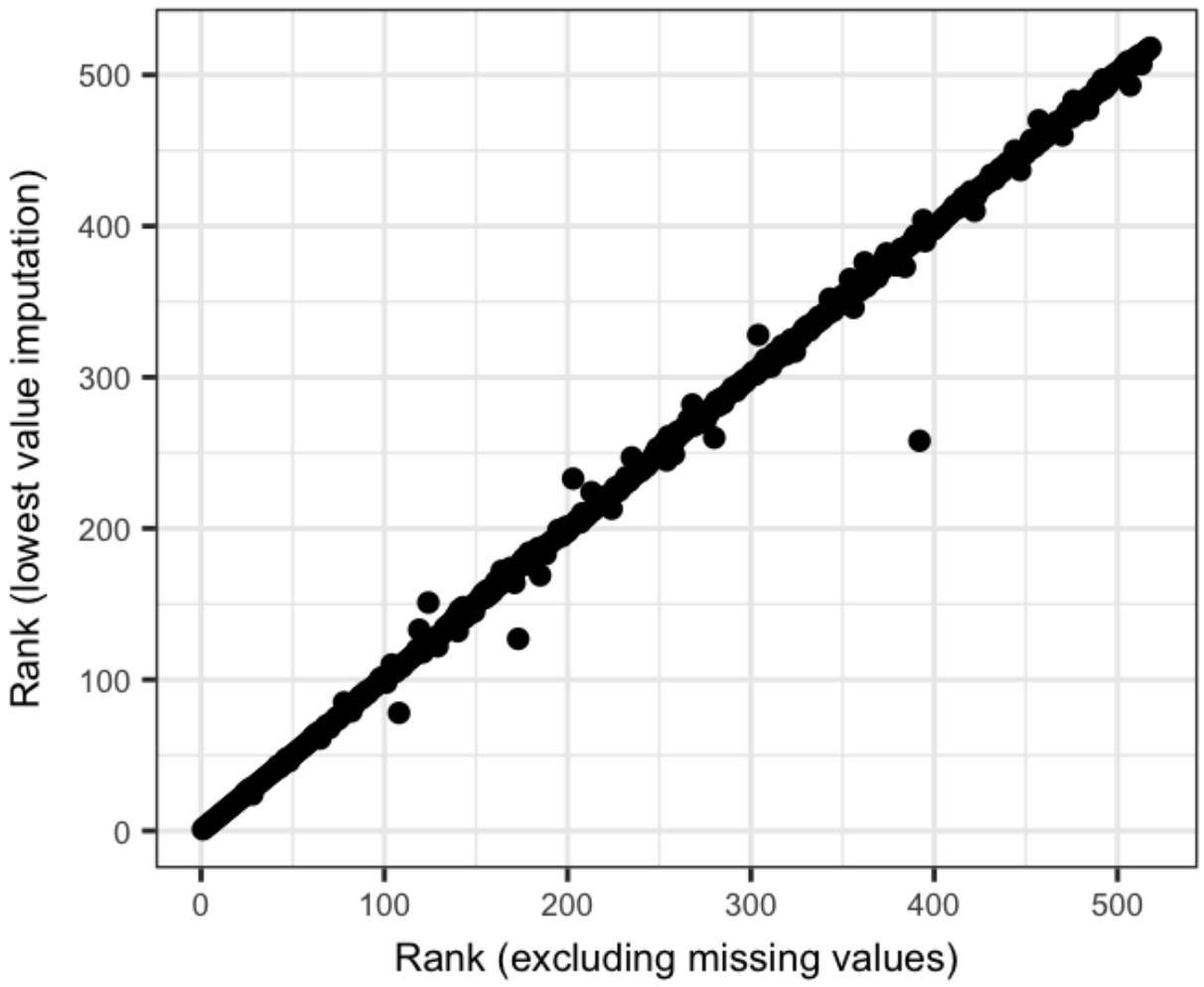


Figure 75: Plot of the rank of differentially-expressed proteins for the ER 45min vs ER 0min contrast, comparing results following exclusion of missing values and lowest value imputation.

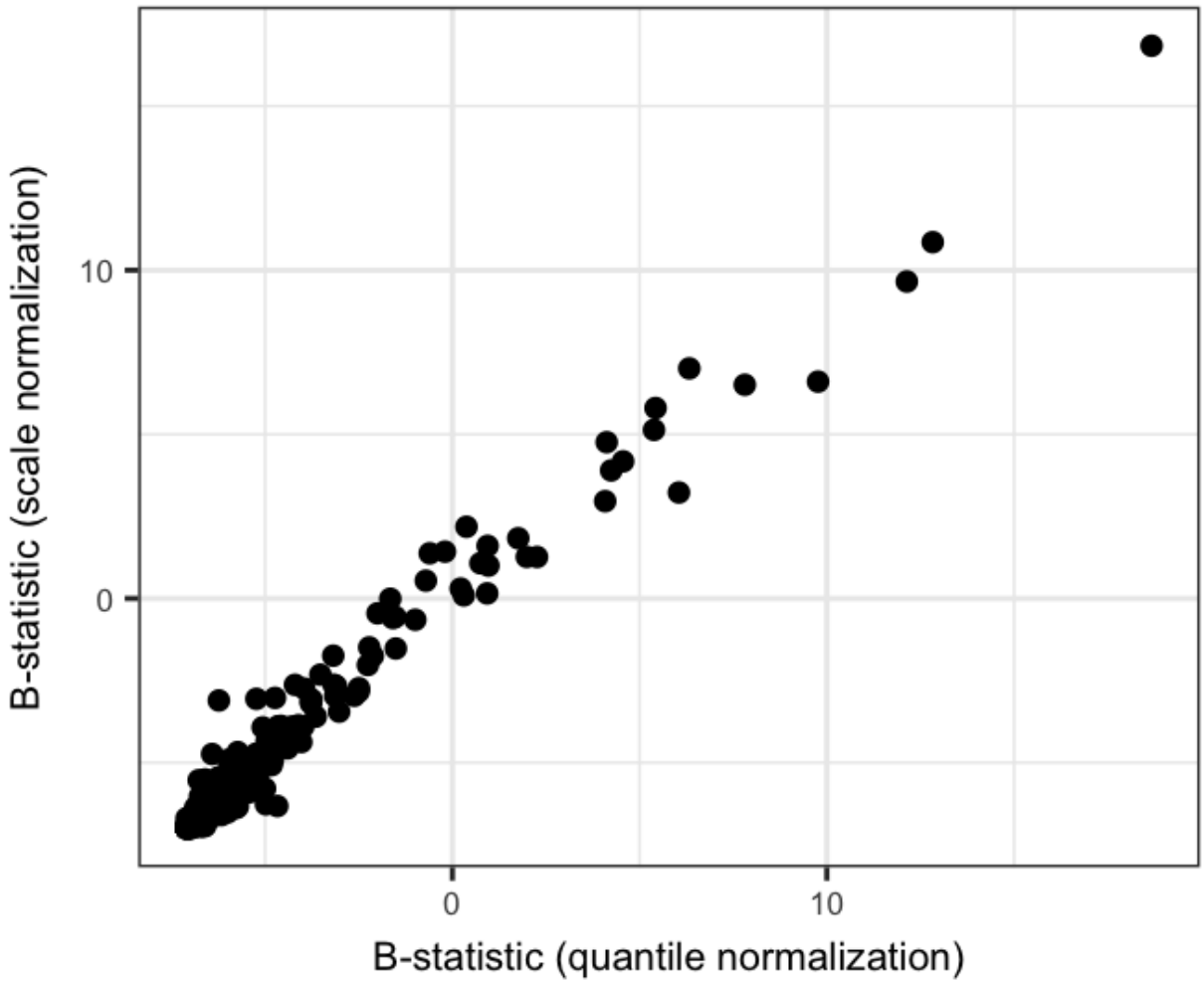


Figure 76: Plot of B-statistic (log odds) for differential expression of proteins for the ER 45min vs ER 0min contrast, comparing results following quantile and scale normalization.

4.7.3 Quantile vs Scale normalization (missing values excluded in both)

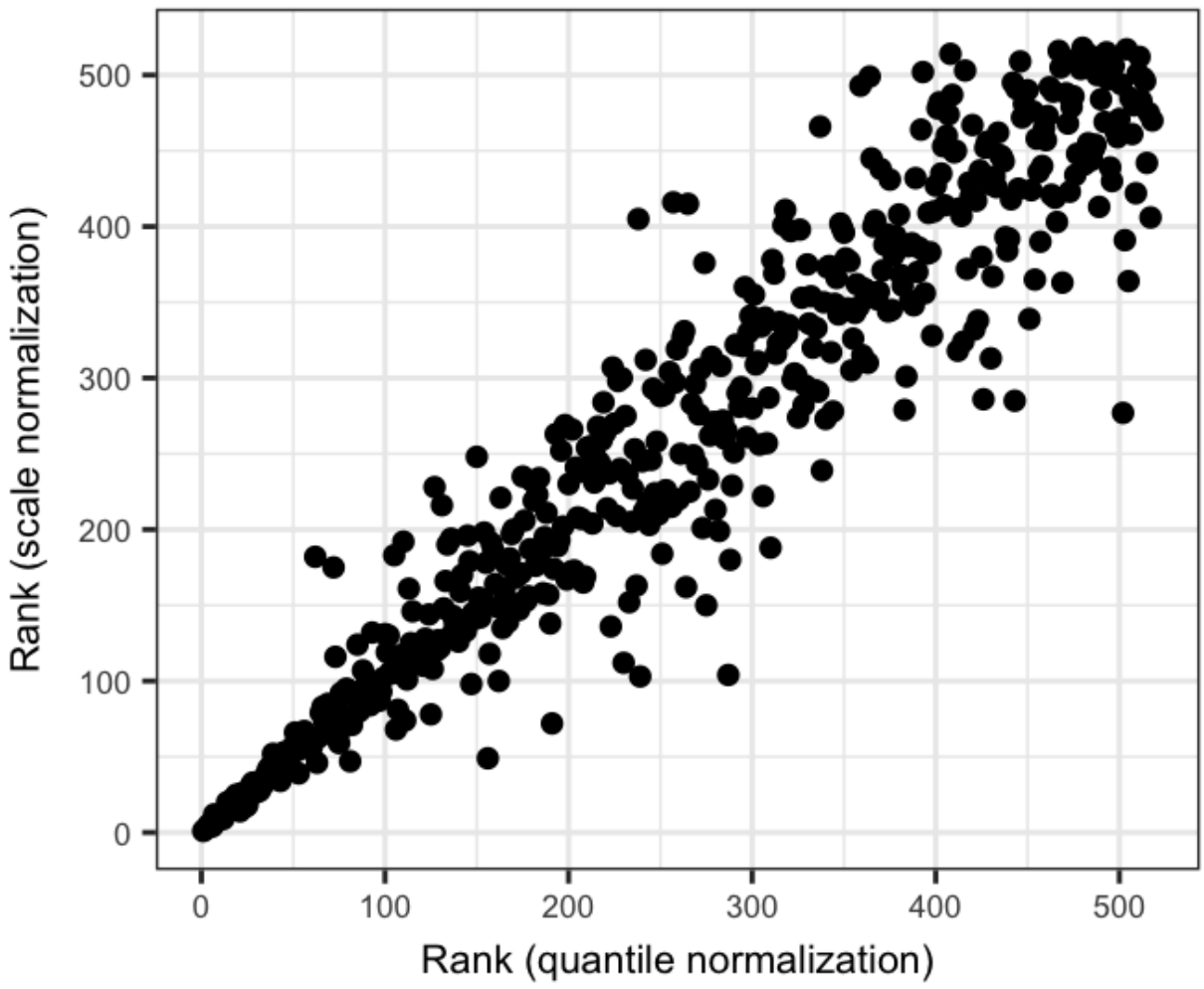


Figure 77: Plot of the rank of differentially-expressed proteins for the ER 45min vs ER 0min contrast, comparing results following quantile and scale normalization.

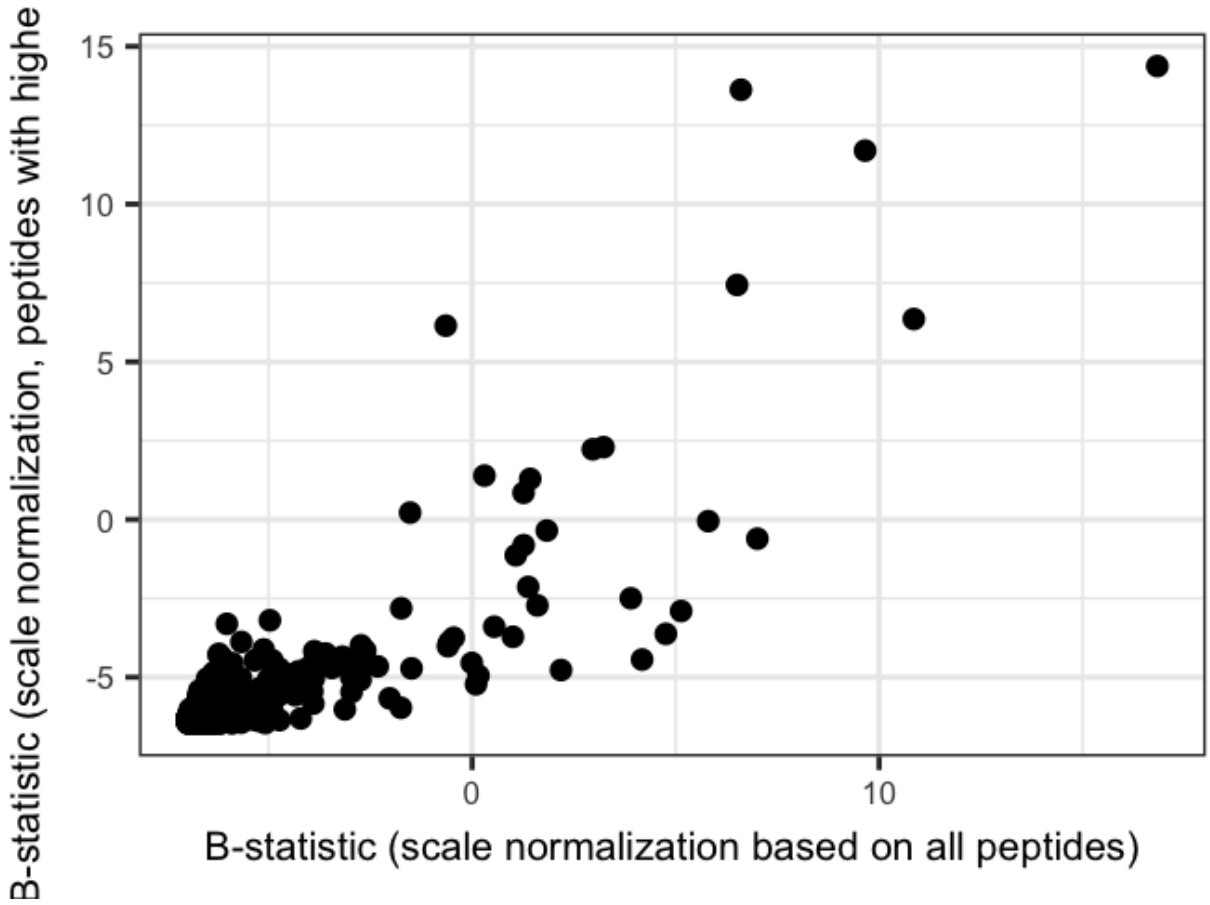


Figure 78: Plot of B-statistic (log odds) for differential expression of proteins for the ER 45min vs ER 0min contrast, comparing results following scale normalization based on all peptides and those with the highest IgG intensities.

4.7.4 Scale normalization based on all peptides vs peptides with highest IgG intensities (missing values excluded in both)

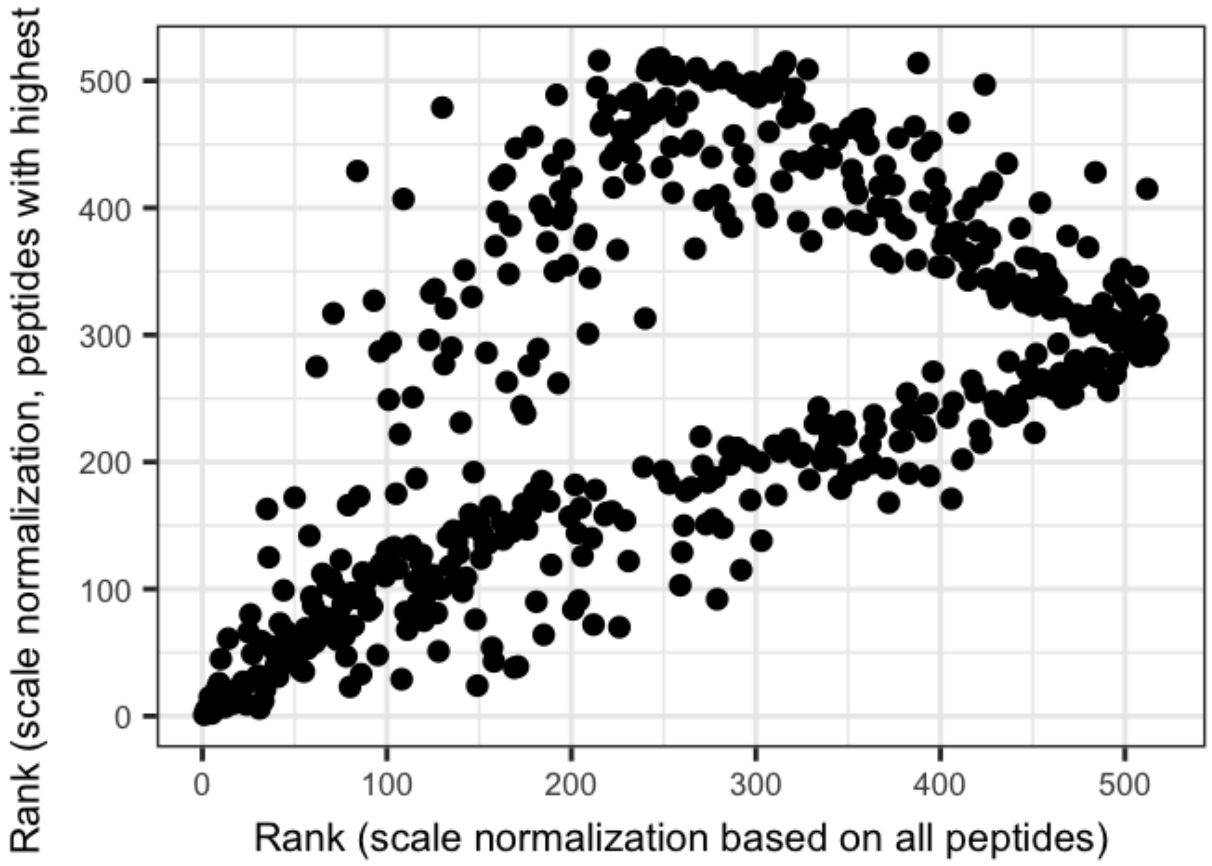


Figure 79: Plot of the rank of differentially-expressed proteins for the ER 45min vs ER 0min contrast, comparing results following scale normalization based on all peptides and those with the highest IgG intensities.

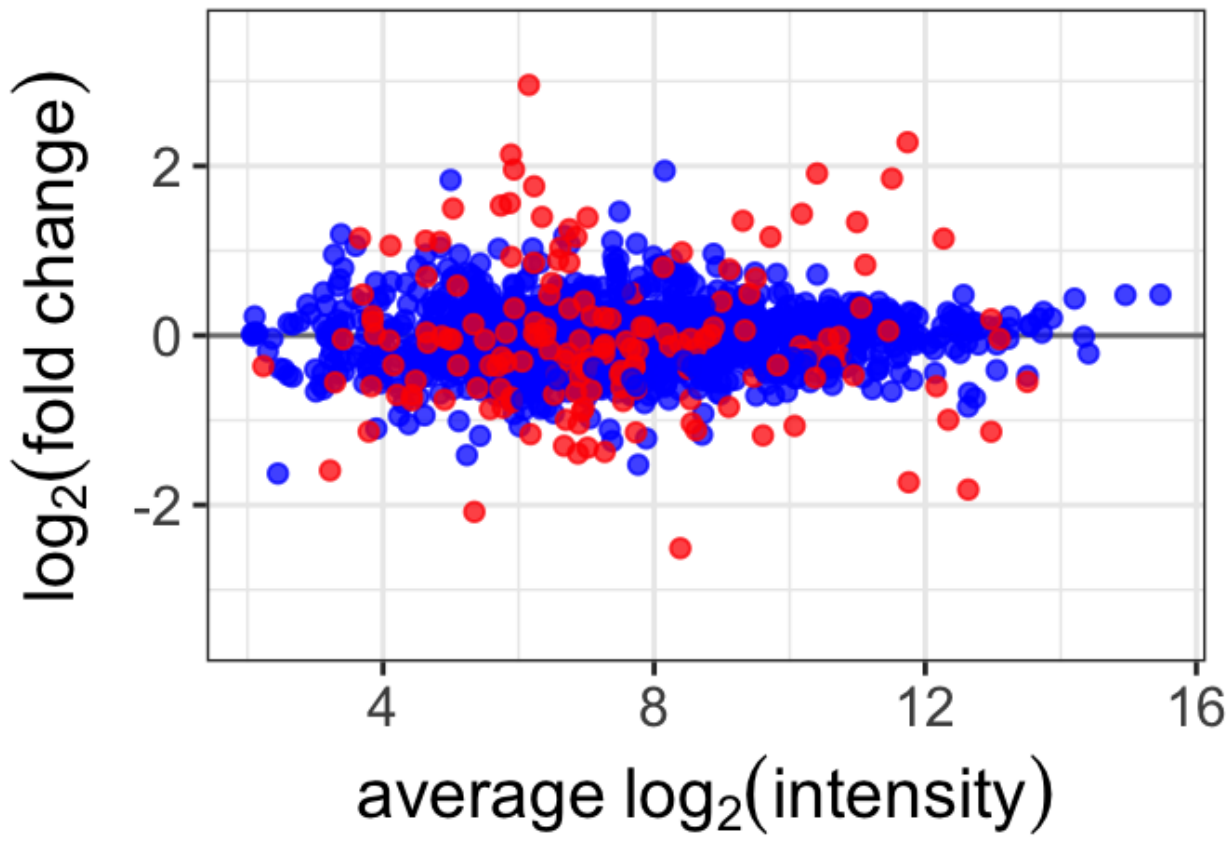


Figure 80: MA plot showing the distribution of non-specific binding. Non-specific binding was identified as proteins that showed less than a log2 enrichment over the IgG channel at either 0 or 45 minutes. The majority of non-specific binding (blue) is found to show less change between time points whereas the specific interactions (red) shows a broader distribution.

4.8 Comparisoin of variance between specfic and non-specific binding

```
##
##  F test to compare two variances
##
## data:  tmp$logFC[tmp$group == 1] and tmp$logFC[tmp$group == 0]
## F = 0.18045, num df = 1373, denom df = 166, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is less than 1
## 95 percent confidence interval:
##  0.0000000 0.2167525
## sample estimates:
## ratio of variances
##          0.1804544
```

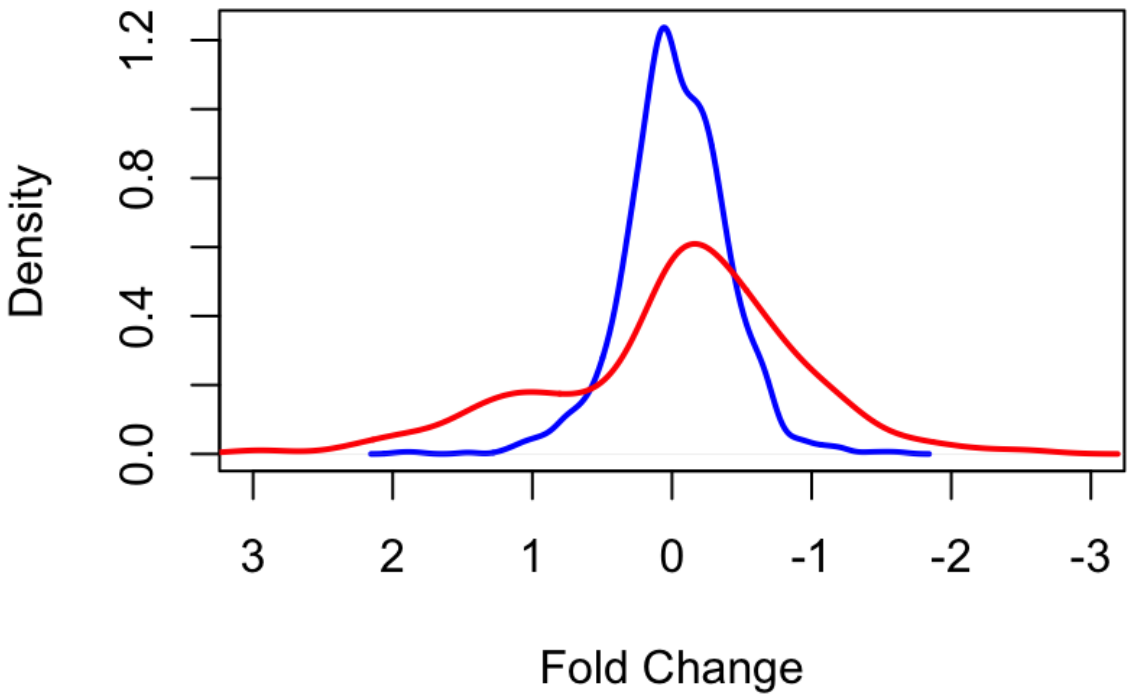


Figure 81: Density plot right showing the distribution of non-specific binding. Non-specific binding was identified as proteins that showed less than a log2 enrichment over the IgG channel at either 0 or 45 minutes. The majority of non-specific binding (blue) is found to show less change between time points whereas the specific interactions (red) shows a broader distribution.

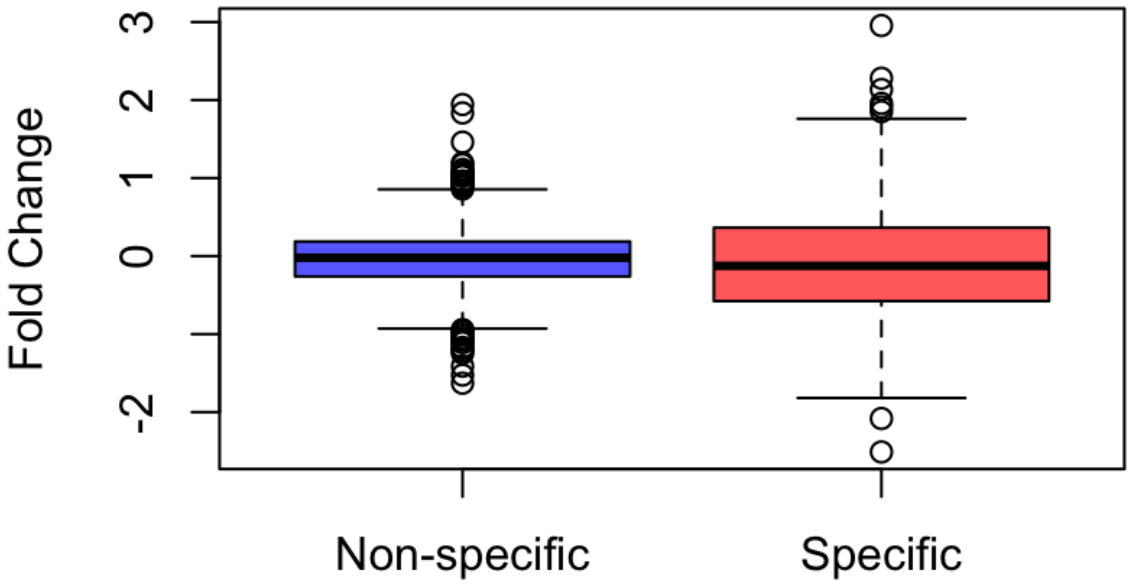


Figure 82: Box plot showing the variance of non-specific vs non-specific binding. Non-specific binding was identified as proteins that showed less than a log2 enrichment over the IgG channel at either 0 or 45 minutes. The the variance of the log fold changes for the non-specific binding interactions was significantly less than that of the specific interactions (F-test, one-sided, p-value $< 2.2e^{*}$-16).

4.9 Differential binding results tables

The following tables contain the top ranking differentially expressed proteins for each comparison. Included are all proteins that reach a statistical signficance of 0.05 in terms of the adjusted p-value and those with an absolute $\log_2$ fold change of 1 or above.

The IgG column gives the larger of the $\log_2$ fold changes for the two groups against the IgG control and an asterisk indicates specific binding where this $\log_2$ fold change is above a threshold of 1. N is the number of replicates in which the protein was observed.

In all cases, peptide intensities were quantile normalized and measurements with missing values were removed prior to summarization at the protein level.

4.9.1 ER 45min vs ER 0min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q15596	NCOA2	1	2.96	6.15			2.12	*
Q13451	FKBP5	2	-2.51	8.39			2.55	*
Q9Y6Q9	NCOA3	3	2.28	11.74	1e-09	18.67	1.90	*
Q96K62	ZBT45	1	2.14	5.89			2.58	*
Q96LC7	SIG10	1	-2.08	5.35			1.79	*
Q6IQ16	SPOPL	1	1.96	5.93			1.59	*
P11474	ERR1	3	1.94	8.16	0.015	-0.99	-0.14	
P48552	NRIP1	3	1.91	10.40	1.2e-05	7.81	1.56	*
O15164	TIF1A	3	1.85	11.51	2.1e-07	12.83	2.15	*
Q9Y2G9	SBNO2	1	1.83	5.00			0.09	
P07900	HS90A	3	-1.82	12.63	2.2e-06	9.77	2.23	*
O75448	MED24	1	1.76	6.23			1.66	*
Q02790	FKBP4	3	-1.73	11.76	2.8e-07	12.14	1.89	*
O14744	ANM5	1	-1.63	2.45			-0.44	
Q9P2D0	IBTK	1	-1.59	3.22			1.84	*
P10276	RARA	3	1.56	5.87	0.0034	0.75	2.75	*
Q15648	MED1	1	1.53	5.74			1.97	*
Q5D862	FILA2	1	-1.52	7.77			-0.29	
P32242	OTX1	2	1.50	5.03			1.88	*
Q2M1P5	KIF7	1	1.46	7.49			0.71	
P24468	COT2	3	1.44	10.18	7.6e-05	5.43	2.07	*
P34932	HSP74	1	-1.41	5.24			0.54	
O14770	MEIS2	1	1.40	6.35			1.08	*
P61960	UFM1	1	-1.39	6.88			1.92	*
Q8N2W9	PIAS4	2	1.39	7.02			1.68	*
Q86UV5	UBP48	1	-1.37	7.27			1.36	*
Q92793	CBP	3	1.35	9.31	0.0015	1.76	1.48	*
P23771	GATA3	3	1.34	10.99	4.5e-05	6.33	1.61	*
Q9P2D7	DYH1	1	-1.32	7.02			2.63	*
O60884	DNJA2	3	-1.30	6.67	0.023	-1.51	1.76	*
Q9UBW7	ZMYM2	1	1.25	6.75			1.46	*
Q01546	K22O	2	-1.25	7.38			-1.21	
O60809	PRA10	1	-1.22	7.88			-0.24	
O15117	FYB	1	1.19	3.38			-1.88	
P53990	IST1	1	-1.18	5.43			-4.76	
P31689	DNJA1	3	-1.18	9.61	0.00019	4.08	1.20	*
O00712	NFIB	1	1.17	6.67			-0.34	
Q8N283	ANR35	1	-1.17	8.71			-0.15	
Q9UL15	BAG5	1	-1.16	6.18			1.11	*
Q6PHW0	IYD1	1	1.16	6.85			1.52	*
Q9ULJ6	ZMIZ1	3	1.16	9.72	7.6e-05	5.38	1.61	*
P15408	FOSL2	1	1.15	3.66			1.69	*
Q9UNE7	CHIP	2	-1.14	7.73			1.05	*
Q9UPN9	TRI33	3	1.14	12.27	0.00019	4.24	1.73	*
P11142	HSP7C	3	-1.14	12.97	5e-05	6.05	1.00	*
Q9NWS6	F118A	1	-1.13	3.79			1.60	*
Q9HB71	CYBP	3	-1.12	8.62	0.0011	2.26	1.02	*
Q8TCU4	ALMS1	1	1.12	4.63			1.44	*
O14686	KMT2D	1	1.11	7.38			0.39	

Continued on next page

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q7Z794	K2C1B	1	-1.11	7.34			-0.84	
Q6KC79	NIPBL	1	1.11	4.84			2.63	*
Q68E01	INT3	2	-1.10	3.90			-0.67	
Q8NFD5	ARI1B	1	1.08	7.74			0.56	
Q9Y6X2	PIAS3	2	1.07	6.75			0.71	
Q92624	APBP2	1	-1.07	6.01			0.05	
O15355	PPM1G	3	-1.07	10.07	0.0072	-0.20	1.09	*
Q8NH53	O52N1	1	1.06	4.11			1.28	*
Q13492	PICAL	1	1.06	3.60			0.12	
Q4L235	ACSF4	1	-1.05	4.38			-0.13	
O00170	AIP	2	-1.04	6.89			1.13	*
P14625	ENPL	2	1.04	4.85			-0.35	
O60244	MED14	1	1.04	6.62			1.56	*
Q15185	TEBP	1	-1.03	8.55			1.29	*
Q6P2C8	MED27	1	1.03	6.21			-0.01	
Q5VTD9	GFI1B	1	1.02	5.70			0.42	
O14929	HAT1	2	-1.01	5.12			0.70	
P08238	HS90B	3	-0.99	12.34	0.005	0.22	2.04	*
P55854	SUMO3	3	0.98	8.41	0.003	0.96	1.13	*
Q09472	EP300	3	0.96	8.88	0.00019	4.12	0.80	
P0DMV9	HS71B	3	-0.84	12.63	0.0013	1.99	0.65	
Q6ISB3	GRHL2	3	0.83	11.12	0.00015	4.56	1.04	*
Q92754	AP2C	3	0.81	8.14	0.0046	0.38	1.15	*
Q92785	REQU	3	0.78	9.11	0.003	0.93	1.16	*
P52701	MSH6	3	-0.76	6.05	0.048	-2.50	0.97	
P63165	SUMO1	3	0.74	9.15	0.0048	0.31	0.72	
P12956	XRCC6	3	-0.74	12.73	0.01	-0.60	0.90	
P31948	STIP1	3	-0.66	11.50	0.003	0.94	0.59	
P78527	PRKDC	3	-0.63	11.22	0.011	-0.71	0.36	
Q9HAV4	XPO5	3	-0.63	8.66	0.048	-2.50	0.72	
O14497	ARI1A	3	0.61	8.60	0.04	-2.26	0.29	
Q9Y383	LC7L2	3	-0.55	10.30	0.023	-1.53	0.51	
Q13263	TIF1B	3	-0.55	13.51	0.033	-2.00	1.17	*
P09874	PARP1	3	-0.53	11.81	0.023	-1.59	0.63	
Q92925	SMRD2	3	0.51	9.91	0.037	-2.12	0.23	
P25685	DNJB1	3	-0.45	9.66	0.039	-2.22	0.60	
Q99873	ANM1	3	-0.42	11.29	0.024	-1.66	0.65	

Table 4: Top ranking differentially expressed proteins from the ER 45min vs ER 0min comparison, sorted by log2 fold change.

4.9.2 ER 90min vs ER 0min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q15596	NCOA2	1	2.82	6.15			1.98	*
Q13451	FKBP5	2	-2.39	8.39			2.55	*
Q96LC7	SIG10	1	-2.25	5.35			1.79	*
Q9Y6Q9	NCOA3	3	2.22	11.74	1.5e-09	18.26	1.84	*
Q96K62	ZBT45	1	2.19	5.89			2.63	*
Q6IQ16	SPOPL	1	2.16	5.93			1.79	*
Q9Y2G9	SBNO2	1	2.15	5.00			0.41	
P48552	NRIP1	3	2.03	10.40	5e-06	8.73	1.68	*
Q6UX73	CP089	1	2.00	8.00			1.27	*
O75448	MED24	1	1.88	6.23			1.77	*
P20393	NR1D1	1	1.86	8.26			1.17	*
Q68CL5	TPGS2	2	-1.85	3.12			-6.76	
O15164	TIF1A	3	1.84	11.51	2.4e-07	12.70	2.14	*
P07900	HS90A	3	-1.82	12.63	2.3e-06	9.75	2.23	*
P11474	ERR1	3	1.81	8.16	0.024	-1.59	-0.28	
O14770	MEIS2	1	1.76	6.35			1.44	*
Q2M1P5	KIF7	1	1.74	7.49			0.99	
Q02790	FKBP4	3	-1.72	11.76	3.2e-07	11.99	1.89	*
Q15648	MED1	1	1.66	5.74			2.09	*
P10276	RARA	3	1.58	5.87	0.0033	0.87	2.77	*
O60229	KALRN	1	1.57	6.32			0.86	
P15408	FOSL2	1	1.57	3.66			2.11	*
Q9UBW7	ZMYM2	1	1.55	6.75			1.76	*
P81605	DCD	1	1.47	4.63			-1.79	
P24468	COT2	3	1.45	10.18	7.4e-05	5.53	2.08	*
Q86UV5	UBP48	1	-1.44	7.27			1.36	*
P32242	OTX1	2	1.43	5.03			1.81	*
Q92624	APBP2	1	-1.42	6.01			0.05	
Q6KC79	NIPBL	1	1.42	4.84			2.94	*
Q92793	CBP	3	1.41	9.31	0.0011	2.27	1.54	*
P23771	GATA3	3	1.38	10.99	2.8e-05	6.79	1.65	*
Q9P2D7	DYH1	1	-1.37	7.02			2.63	*
O15117	FYB	1	1.36	3.38			-1.71	
P34932	HSP74	1	-1.29	5.24			0.54	
O60884	DNJA2	3	-1.28	6.67	0.024	-1.61	1.76	*
O00712	NFIB	1	1.28	6.67			-0.24	
O76094	SRP72	1	1.25	3.41			0.35	
Q9NWS6	F118A	1	-1.25	3.79			1.60	*
O60934	NBN	1	-1.24	4.24			0.84	
Q13492	PICAL	1	1.24	3.60			0.30	
Q8WX92	NELFB	2	-1.23	6.70			1.26	*
Q8N2W9	PIAS4	2	1.21	7.02			1.50	*
Q8N283	ANR35	1	-1.20	8.71			-0.15	
Q6KB66	K2C80	2	1.19	11.02			0.73	
Q9UNE7	CHIP	2	-1.17	7.73			1.05	*
O60809	PRA10	1	-1.17	7.88			-0.24	
P31689	DNJA1	3	-1.16	9.61	0.00026	3.87	1.20	*
Q9Y6X2	PIAS3	2	1.15	6.75			0.79	
Q4L235	ACSF4	1	-1.14	4.38			-0.13	

Continued on next page

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q15126	PMVK	1	-1.14	5.45			0.49	
Q9Y4C1	KDM3A	1	-1.11	5.93			0.91	
Q13158	FADD	1	-1.11	2.11			-0.90	
Q9ULL5	PRR12	1	1.11	3.33			-0.10	
P11142	HSP7C	3	-1.11	12.97	7.3e-05	5.69	1.00	*
Q9HB71	CYBP	3	-1.10	8.62	0.0013	2.04	1.02	*
Q6P2C8	MED27	1	1.10	6.21			0.06	
Q9UL15	BAG5	1	-1.09	6.18			1.11	*
Q6PHW0	IYD1	1	1.09	6.85			1.45	*
P07205	PGK2	1	-1.08	6.73			1.03	*
Q96QK1	VPS35	1	1.08	4.72			0.59	
O15355	PPM1G	3	-1.07	10.07	0.0065	-0.12	1.09	*
Q9UPN9	TRI33	3	1.05	12.27	0.00045	3.22	1.64	*
P61960	UFM1	1	-1.05	6.88			1.92	*
Q9ULJ6	ZMIZ1	3	1.04	9.72	0.00025	3.98	1.49	*
O43866	CD5L	2	-1.04	6.17			-2.51	
O14686	KMT2D	1	1.04	7.38			0.32	
Q5D862	FILA2	1	-1.03	7.77			-0.29	
Q09472	EP300	3	1.03	8.88	0.00011	5.00	0.86	
Q8NFD5	ARI1B	1	1.02	7.74			0.50	
P08238	HS90B	3	-1.00	12.34	0.0047	0.32	2.04	*
P55854	SUMO3	3	0.96	8.41	0.0034	0.71	1.11	*
P52701	MSH6	3	-0.93	6.05	0.012	-0.78	0.97	
Q6ISB3	GRHL2	3	0.85	11.12	0.00012	4.82	1.05	*
Q92754	AP2C	3	0.84	8.14	0.0033	0.88	1.19	*
P63165	SUMO1	3	0.79	9.15	0.0033	0.98	0.77	
Q92785	REQU	3	0.77	9.11	0.0034	0.81	1.15	*
P0DMV9	HS71B	3	-0.75	12.63	0.0034	0.76	0.65	
P61956	SUMO2	3	0.73	9.50	0.033	-2.05	1.10	*
P12956	XRCC6	3	-0.72	12.73	0.012	-0.75	0.90	
O14497	ARI1A	3	0.72	8.60	0.012	-0.81	0.41	
P78527	PRKDC	3	-0.68	11.22	0.0061	-0.03	0.36	
Q9HAV4	XPO5	3	-0.65	8.66	0.037	-2.19	0.72	
P31948	STIP1	3	-0.62	11.50	0.0047	0.29	0.59	
Q04724	TLE1	3	0.61	9.40	0.041	-2.32	1.12	*
Q92925	SMRD2	3	0.58	9.91	0.014	-1.00	0.31	
P55060	XPO2	3	-0.56	10.10	0.044	-2.43	0.86	
Q9Y383	LC7L2	3	-0.51	10.30	0.037	-2.18	0.51	
P09874	PARP1	3	-0.50	11.81	0.033	-2.01	0.63	
Q99873	ANM1	3	-0.40	11.29	0.033	-1.99	0.65	

Table 5: Top ranking differentially expressed proteins from the ER 90min vs ER 0min comparison, sorted by log2 fold change.

4.9.3 ER 90min vs ER 45min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q13158	FADD	1	-1.34	2.11			-0.67	
Q68CL5	TPGS2	2	-1.21	3.12			-7.40	
Q6UX73	CP089	1	1.20	8.00			1.27	*
P33176	KINH	1	1.09	4.64			0.78	
O14744	ANM5	1	1.06	2.45			-1.01	
Q8WXG9	GPR98	1	1.02	6.85			2.30	*

Table 6: Top ranking differentially expressed proteins from the ER 90min vs ER 45min comparison, sorted by log2 fold change.

4.9.4 FOXA1 45min vs FOXA1 0min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q8NH53	O52N1	1	4.29	4.11			4.90	*
Q96QK1	VPS35	1	3.72	4.72			4.08	*
A6NNA2	SRRM3	1	3.69	7.45			4.27	*
P57773	CXA9	1	3.12	8.22			2.52	*
A5PLK6	RGSL	1	-3.03	6.46			0.16	
P42356	PI4KA	1	-2.64	4.49			-1.42	
Q9NVG8	TBC13	1	-2.39	6.80			0.41	
O94844	RHBT1	1	-2.35	5.71			0.20	
P57678	GEMI4	1	-2.32	7.40			-0.16	
Q86UP2	KTN1	1	2.23	6.69			2.02	*
P08729	K2C7	1	-2.19	3.89			0.04	
Q9NW13	RBM28	1	2.06	6.68			2.13	*
O94992	HEXI1	2	-1.70	4.22			0.53	
Q86WI1	PKHL1	1	-1.69	4.09			-0.70	
Q9H2Y7	ZN106	1	1.66	7.50			-1.73	
Q01813	PFKAP	1	-1.62	3.27			-0.69	
Q8IV04	TB10C	1	-1.61	6.35			0.15	
Q9H3P2	NELFA	1	-1.54	3.92			0.55	
Q9NWS6	F118A	1	-1.49	3.79			0.73	
Q9Y3E5	PTH2	1	1.49	3.11			0.89	
Q8TDN6	BRX1	1	1.46	7.29			1.55	*
Q8NF37	PCAT1	1	-1.41	7.34			-0.42	
Q9H0A0	NAT10	1	1.38	2.30			1.61	*
Q96KQ4	ASPP1	1	-1.38	3.41			-0.79	
P33176	KINH	1	-1.36	4.64			-0.26	
P20393	NR1D1	1	-1.32	8.26			0.77	
O60610	DIAP1	1	-1.32	4.23			0.08	
Q9BQG0	MBB1A	1	1.30	8.49			1.14	*
Q9HA92	RSAD1	1	-1.30	7.23			0.48	
Q9UHB6	LIMA1	1	1.30	3.12			-0.08	
Q9BXD5	NPL	1	-1.29	4.27			0.16	
Q5VTD9	GFI1B	1	-1.29	5.70			0.24	
Q13492	PICAL	1	1.26	3.60			1.43	*
Q15413	RYR3	1	-1.26	6.19			-5.07	
Q96HY6	DDRGK	1	-1.23	3.95			-0.17	
O00487	PSDE	1	1.23	2.15			0.67	
O15269	SPTC1	1	-1.18	3.96			0.19	
Q9HDC9	APMAP	1	-1.14	7.62			-0.44	
O75179	ANR17	1	1.14	3.36			0.61	
Q8ND56	LS14A	1	1.14	3.28			0.07	
O60229	KALRN	1	-1.13	6.32			0.23	
O95433	AHSA1	1	1.13	3.19			1.67	*
Q5T0W9	FA83B	1	1.13	6.88			0.78	
P00450	CERU	1	1.12	6.76			0.04	
Q96CT7	CC124	1	-1.12	6.21			-0.32	
Q14677	EPN4	1	-1.09	6.32			-1.86	
Q08J23	NSUN2	2	-1.09	4.80			0.50	
Q14966	ZN638	1	1.08	7.79			1.09	*
Q9H9B1	EHMT1	1	1.07	3.76			1.84	*

Continued on next page

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG
A5YKK6	CNOT1	1	-1.06	5.89			-0.60
Q9HCH5	SYTL2	1	-1.04	6.63			0.77
P61966	AP1S1	1	1.01	3.79			0.46
Q06787	FMR1	1	-1.01	7.26			-0.15
O15117	FYB	1	1.01	3.38			0.64
P29083	T2EA	1	1.01	5.86			-3.44
Q8IVT2	MISP	1	1.01	5.52			0.40

Table 7: Top ranking differentially expressed proteins from the FOXA1 45min vs FOXA1 0min comparison, sorted by log2 fold change.

4.9.5 FOXA1 90min vs FOXA1 0min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG
A5PLK6	RGSL	1	-1.53	6.46			0.16
Q5T0F9	C2D1B	1	-1.28	3.34			0.00
P04433	KV309	1	-1.24	8.54			0.36
P55036	PSMD4	1	-1.22	4.17			0.15
O95433	AHSA1	1	-1.13	3.19			0.54
O00487	PSDE	1	-1.08	2.15			-0.56

Table 8: Top ranking differentially expressed proteins from the FOXA1 90min vs FOXA1 0min comparison, sorted by log2 fold change.

4.9.6 FOXA1 90min vs FOXA1 45min

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
Q8NH53	O52N1	1	-4.27	4.11			4.90	*
Q96QK1	VPS35	1	-4.20	4.72			4.08	*
P42356	PI4KA	1	3.49	4.49			-0.57	
P57773	CXA9	1	-2.97	8.22			2.52	*
A6NNA2	SRRM3	1	-2.88	7.45			4.27	*
Q86UP2	KTN1	1	-2.69	6.69			2.02	*
Q9NVG8	TBC13	1	2.67	6.80			0.69	
O94844	RHBT1	1	2.51	5.71			0.36	
O00487	PSDE	1	-2.31	2.15			0.67	
O95433	AHSA1	1	-2.26	3.19			1.67	*
Q15413	RYR3	1	2.13	6.19			-4.20	
P08729	K2C7	1	2.04	3.89			-0.11	
Q9H0A0	NAT10	1	-1.98	2.30			1.61	*
O75179	ANR17	1	-1.91	3.36			0.61	
Q9HDC9	APMAP	1	1.89	7.62			0.31	
P57678	GEMI4	1	1.88	7.40			-0.60	
Q8ND56	LS14A	1	-1.81	3.28			0.07	
Q9H3P2	NELFA	1	1.74	3.92			0.76	
O94992	HEXI1	2	1.69	4.22			0.51	
Q8IV04	TB10C	1	1.67	6.35			0.20	
P33176	KINH	1	1.65	4.64			0.03	
P61966	AP1S1	1	-1.62	3.79			0.46	
Q9NWS6	F118A	1	1.51	3.79			0.75	
A5PLK6	RGSL	1	1.50	6.46			-1.37	
Q13492	PICAL	1	-1.46	3.60			1.43	*
Q8TE85	GRHL3	1	1.43	4.85			-0.03	
Q9HCH5	SYTL2	1	1.43	6.63			1.16	*
Q9H2Y7	ZN106	1	-1.40	7.50			-1.73	
Q9Y6E0	STK24	1	-1.39	3.66			0.52	
Q14677	EPN4	1	1.37	6.32			-1.58	
P23677	IP3KA	1	1.35	7.05			-0.42	
Q9NW13	RBM28	1	-1.33	6.68			2.13	*
Q9Y3E5	PTH2	1	-1.32	3.11			0.89	
Q5T0F9	C2D1B	1	-1.32	3.34			0.04	
Q9HA92	RSAD1	1	1.31	7.23			0.49	
P04004	VTNC	1	1.29	6.39			-0.30	
O60229	KALRN	1	1.29	6.32			0.38	
Q8NF37	PCAT1	1	1.28	7.34			-0.55	
Q86WI1	PKHL1	1	1.26	4.09			-1.13	
Q5T1M5	FKB15	1	-1.25	6.17			-0.05	
Q8IUE6	H2A2B	2	1.23	7.40			0.00	
Q9H0S4	DDX47	1	1.18	7.09			-0.14	
Q96KQ4	ASPP1	1	1.16	3.41			-1.01	
Q8IVT2	MISP	1	-1.15	5.52			0.40	
Q9H9B1	EHMT1	1	-1.13	3.76			1.84	*
Q13158	FADD	1	-1.10	2.11			0.37	
P07384	CAN1	1	-1.09	5.85			0.49	
P32242	OTX1	2	-1.09	5.03			2.36	*
P02656	APOC3	3	1.08	6.97	1	-4.11	-0.76	

Continued on next page

Protein	Gene	N	log2FC	Avg Expr	p-value	B	IgG	
P35580	MYH10	1	1.07	7.89			-0.23	
O00442	RTCA	1	-1.06	2.48			1.26	*
P39019	RS19	2	-1.06	8.37			-0.10	
P14770	GPIX	2	1.06	6.28			-0.01	
Q9UHB6	LIMA1	1	-1.03	3.12			-0.08	
O14745	NHRF1	1	-1.02	5.90			0.31	

Table 9: Top ranking differentially expressed proteins from the FOXA1 90min vs FOXA1 45min comparison, sorted by log2 fold change.

5 GRHL2 ChIP-Seq Analysis

5.1 Pre-processing

5.1.1 Download files

```
#!/bin/sh
java -jar ../java/clarity-tools.jar -l SLX-14333
```

5.1.2 Removal of reads in blacklisted sites

```
#downloaded from http://mitra.stanford.edu/kundaje/akundaje/release/blacklists/hg38-human/hg38.blacklis

bl=../blacklists/hg38.blacklist.bed


mkdir ./blacklist_filtered
cd SLX-14333
for f in *.bam
do
    echo $f
    bedtools intersect -v -abam $f -b $bl  > ../blacklist_filtered/$f
done

### Re-index

cd ../blacklist_filtered
for f in *.bam
do
samtools index $f
done
```

5.1.3 Peak Calling

```
### MACS peak caller

### Run macs on the blacklisted data
```

```
mkdir ./peaks
cd peaks
control=../GRHL2_filtered/SLX-14333.D701_D503.bam
for bam in ../GRHL2_filtered/*.bam
do
root=`basename $bam .bam`
macs2 callpeak -t $bam -c $control -f BAM -n $root -g hs &
done
```

5.2 Differntial binding analysis

5.2.1 MA plot

```
suppressMessages(library(DiffBind))

if(!file.exists("rdata/003_diffbind.rda")) {
    GRHL2 <- dba(sampleSheet="samplesheet/samplesheet.csv")
    GRHL2 <- dba.count(GRHL2, summits=250)
    GRHL2 <- dba.contrast(GRHL2)
    GRHL2 <- dba.analyze(GRHL2)
    save(GRHL2,file="rdata/003_diffbind.rda")
} else {
    load("rdata/003_diffbind.rda")
}


dba.plotMA(GRHL2, bFlip=1)
```

5.2.2 PCA

```
dba.plotPCA(GRHL2,components = 2:3)
```

5.2.3 Table of differntiall bound sites

```
library(knitr)
kable(head(as.data.frame(dba.report(GRHL2))))
```

	seqnames	start	end	width	strand	Conc	Conc_none	Conc_ER	Fold	p.value	FDR
12123	14	93959394	93959894	501	*	7.63	6.04	8.37	-2.33	0	0e+00
9047	12	75706270	75706770	501	*	6.74	4.93	7.52	-2.60	0	0e+00
24493	21	31529399	31529899	501	*	6.50	4.86	7.25	-2.39	0	0e+00
129	1	7447769	7448269	501	*	4.98	2.44	5.85	-3.41	0	0e+00
40054	9	75101020	75101520	501	*	6.94	4.80	7.76	-2.96	0	0e+00
28564	3	176953741	176954241	501	*	5.53	3.80	6.30	-2.50	0	1e-07

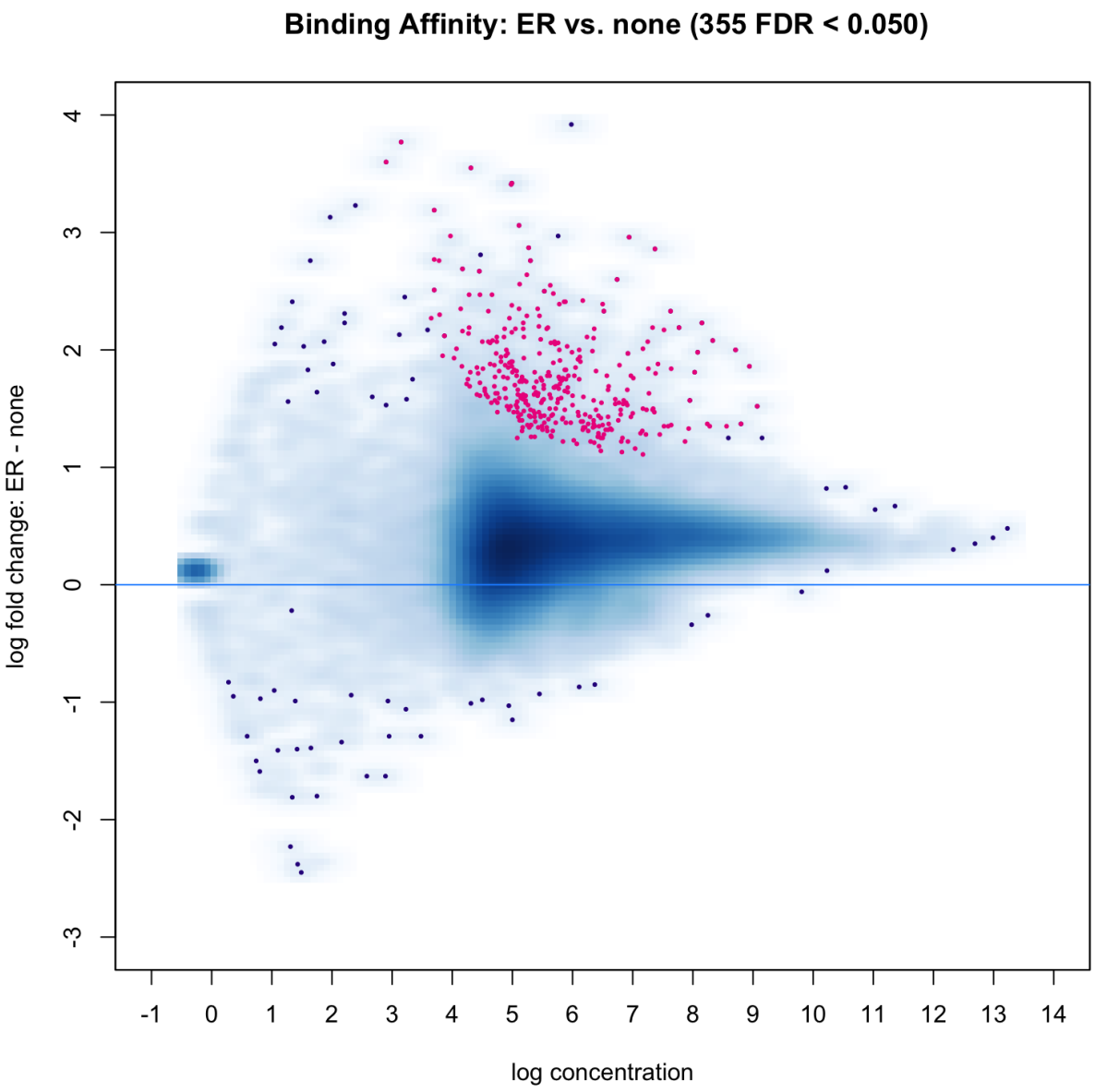


Figure 83: MA plot showing changes in GRHL2 binding before and after treatment with 100nM E2.

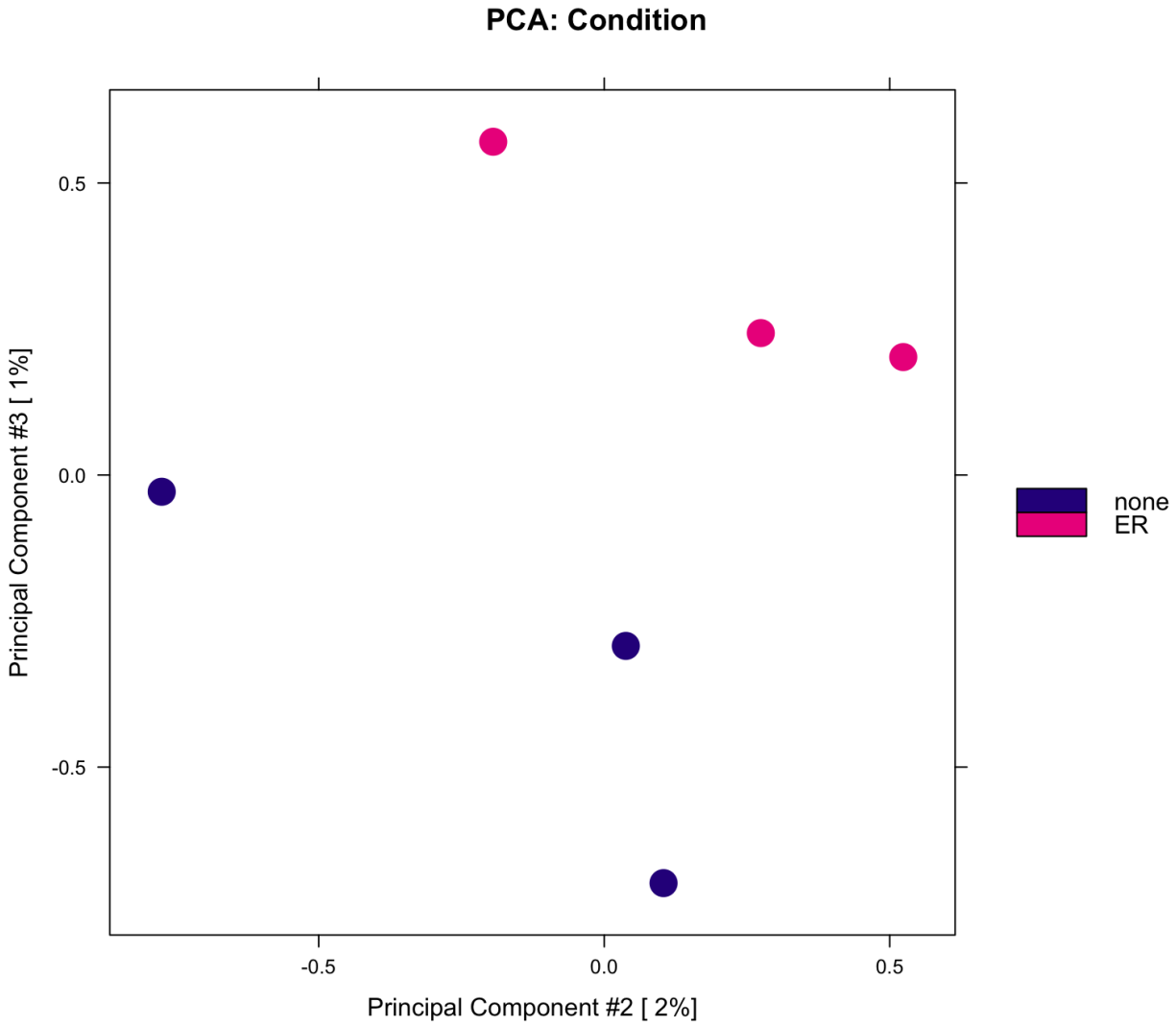


Figure 84: PCA Plot showing clustering of samples by condition

5.2.4 Number of sites with increased or decreased binding

```
r<-dba.report(GRHL2,th=1)
length(r[r$Fold>0])
```

```
## [1] 4973
```

```
length(r[r$Fold<0])
```

```
## [1] 37496
```

5.3 Quality Control

5.3.1 Reproducability of peaks

```
dba.plotVenn(GRHL2,GRHL2$masks$ER, label1="Rep1", label2="Rep2", main="GRHL2 +E2")
```

```
dba.plotVenn(GRHL2,GRHL2$masks$none, label1="Rep1", label2="Rep2", main="GRHL2 -E2")
```

```
called_none<-rowSums(GRHL2$called[,c(1:3)])
called_ER<-rowSums(GRHL2$called[,c(4:6)])
print(paste("Peaks called in -E2 samples:", length(called_none[called_none>0])))
```

```
## [1] "Peaks called in -E2 samples: 38763"
```

```
print(paste("Peaks called in +E2 samples:", length(called_ER[called_ER>0])))
```

```
## [1] "Peaks called in +E2 samples: 42565"
```

```
called_both<-called_none*called_none
print(paste("Peaks called in both settings:", length(called_both[called_both>0])))
```

```
## [1] "Peaks called in both settings: 38763"
```

5.4 VULCAN Analysis

```
suppressMessages(library(vulcan))

dist_calc<-function(method,dfanno,genematrix,genesmore,allsamples){
    # This function structure was strongly suggested
    # by the Bioconductor reviewer
    supportedMethods<-c(
        "closest",
        "strongest",
        "sum",
        "topvar",
        "farthest",
        "lowvar"
    )
    if(!method%in%supportedMethods){
        stop("unsupported method ", method)
    }

    for (gene in genesmore) {
```

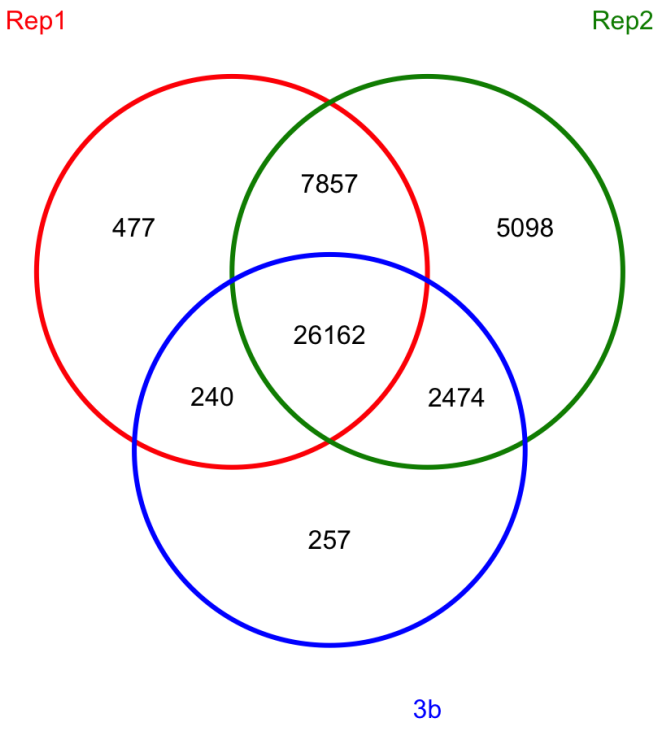


Figure 85: Peak overlap +E2

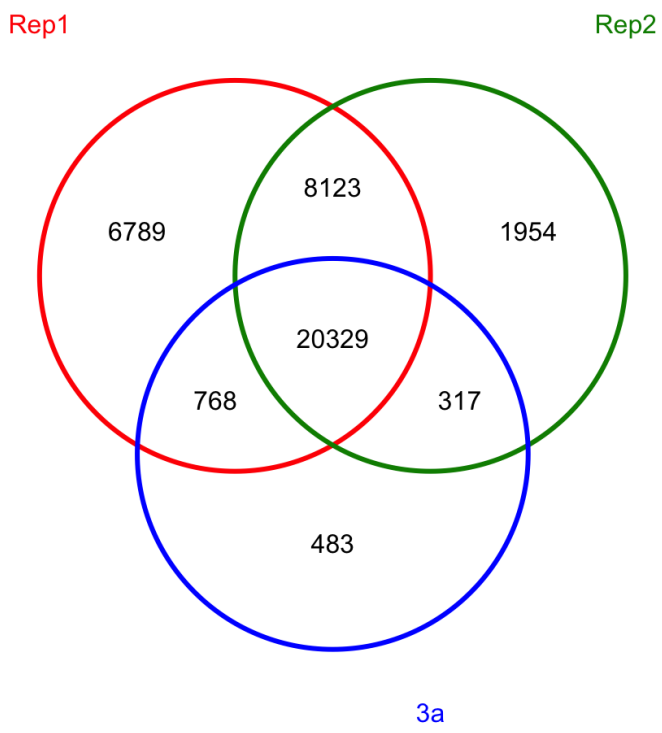


Figure 86: Peak overlap -E2

```
        subanno <- dfanno[dfanno$feature == gene, ]

        if (method == "closest") {
            closest <- which.min(subanno$distanceToStart)
            genematrix[gene, allsamples] <- as.numeric(subanno[closest,
                                                               allsamples])
        }

        if (method == "farthest") {
            farthest <- which.max(subanno$distanceToStart)
            genematrix[gene, allsamples] <- as.numeric(subanno[farthest,
                                                               allsamples])
        }

        if (method == "sum") {
            sums <- apply(subanno[, allsamples], 2, sum)
            genematrix[gene, allsamples] <- as.numeric(sums)
        }

        if (method == "strongest") {
            sums <- apply(subanno[, allsamples], 1, sum)
            top <- which.max(sums)
            genematrix[gene, allsamples] <- as.numeric(subanno[top,
                                                               allsamples])
        }

        if (method == "topvar") {
            vars <- apply(subanno[, allsamples], 1, var)
            top <- which.max(vars)
            genematrix[gene, allsamples] <- as.numeric(subanno[top,
                                                               allsamples])
        }

        if (method == "lowvar") {
            vars <- apply(subanno[, allsamples], 1, var)
            top <- which.min(vars)
            genematrix[gene, allsamples] <- as.numeric(subanno[top,
                                                               allsamples])
        }

    }
    return(genematrix)
}


vulcan.import.dba<- function (dbaobj, samples,intervals = NULL)
{
    dbcounts <- dbaobj
    listcounts <- dbcounts$peaks
    names(listcounts) <- dbcounts$samples[, 1]
    first <- listcounts[[1]]
    rawmat <- matrix(NA, nrow = nrow(first), ncol = length(listcounts) +
                         3)
```

```
    colnames(rawmat) <- c("Chr", "Start", "End", names(listcounts))
    rownames(rawmat) <- 1:nrow(rawmat)
    rawmat <- as.data.frame(rawmat)
    rawmat[, 1] <- as.character(first[, 1])
    rawmat[, 2] <- as.integer(first[, 2])
    rawmat[, 3] <- as.integer(first[, 3])
    for (i in 1:length(listcounts)) {
        rawmat[, names(listcounts)[i]] <- as.numeric(listcounts[[i]]$RPKM)
    }
    peakrpkms <- rawmat
    rm(rawmat)
    first <- listcounts[[1]]
    rawmat <- matrix(NA, nrow = nrow(first), ncol = length(listcounts) +
                         3)
    colnames(rawmat) <- c("Chr", "Start", "End", names(listcounts))
    rownames(rawmat) <- 1:nrow(rawmat)
    rawmat <- as.data.frame(rawmat)
    rawmat[, 1] <- as.character(first[, 1])
    rawmat[, 2] <- as.integer(first[, 2])
    rawmat[, 3] <- as.integer(first[, 3])
    for (i in 1:length(listcounts)) {
        rawmat[, names(listcounts)[i]] <- as.integer(listcounts[[i]]$Reads)
    }
    peakcounts <- rawmat
    rm(rawmat)
    vobj <- list(peakcounts = peakcounts, samples = samples,
                 peakrpkms = peakrpkms)
    return(vobj)
}

prependSampleNames <- function(vobj,prependString){
    colnames(vobj$peakcounts)[0:-3]<-paste0(prependString,colnames(vobj$peakcounts)[0:-3])
    vobj$samples[[2]]<-paste0(prependString,vobj$samples[[2]])
    vobj$samples[[1]]<-paste0(prependString,vobj$samples[[1]])
    colnames(vobj$peakrpkms)[0:-3]<-paste0(prependString,colnames(vobj$peakrpkms)[0:-3])
    return(vobj)
}

loadVulcanNetworks<-function(){
    regulons<-list()
    load("networks/laml-tf-regulon.rda")
    regulons$laml<-regul
    rm(regul)
    load("networks/brca-tf-regulon.rda")
    regulons$tcga<-regul
    rm(regul)
    load("networks/metabric-regulon-tfs.rda")
    regulons$metabric<-regulon
    rm(regulon)
    return(regulons)
}
```

```
#Slow so just load the file below
#vobj<-vulcan.import("samplesheet/samplesheet.csv")
#load(file="003_vobj.Rda")
samples <- list()
samples[['ER']]<-c('1a','2a','3a')
samples[['none']]<-c('1b','2b','3b')

vobj <-vulcan.import.dba(GRHL2,samples)

#vobj<-vulcan.annotate(vobj,lborder=-10000,rborder=10000,method='sum')

vobj <-prependSampleNames(vobj,"X")

lborder=-10000
rborder=10000
method='sum'
#DEBUG
#source("https://bioconductor.org/biocLite.R")
#biocLite("TxDb.Hsapiens.UCSC.hg38.knownGene")
suppressMessages(library("TxDb.Hsapiens.UCSC.hg38.knownGene"))

annotation <- toGRanges(TxDb.Hsapiens.UCSC.hg38.knownGene,
                        feature = "gene")
gr <- GRanges(vobj$peakcounts)
seqlevels(annotation)<- sub('chr','',seqlevels(annotation))
anno <- annotatePeakInBatch(gr, AnnotationData = annotation,
                            output = "overlapping", FeatureLocForDistance = "TSS",
                            bindingRegion = c(lborder, rborder))
```

```
## Annotate peaks by annoPeaks, see ?annoPeaks for details.
```

```
## maxgap will be ignored.
```

```
dfanno <- anno
names(dfanno) <- seq_len(length(dfanno))
dfanno <- as.data.frame(dfanno)
allsamples <- unique(unlist(vobj$samples))
genes <- unique(dfanno$feature)
peakspergene <- table(dfanno$feature)
rawcounts <- matrix(NA, nrow = length(genes), ncol = length(allsamples))
colnames(rawcounts) <- allsamples
rownames(rawcounts) <- genes
genesone <- names(peakspergene)[peakspergene == 1]
for (gene in genesone) {
    rawcounts[gene, allsamples] <- as.numeric(dfanno[dfanno$feature ==
                                                         gene, allsamples])
}

genesmore <- names(peakspergene)[peakspergene > 1]
rawcounts <- dist_calc(method, dfanno, rawcounts, genesmore,
                       allsamples)

gr <- GRanges(vobj$peakrpkms)
anno <- annotatePeakInBatch(gr, AnnotationData = annotation,
                            output = "overlapping", FeatureLocForDistance = "TSS",
```

```
                                bindingRegion = c(lborder, rborder))
```

```
## Annotate peaks by annoPeaks, see ?annoPeaks for details.
## maxgap will be ignored.
```

```
dfanno <- anno
names(dfanno) <- seq_len(length(dfanno))
dfanno <- as.data.frame(dfanno)
allsamples <- unique(unlist(vobj$samples))
genes <- unique(dfanno$feature)
peakspergene <- table(dfanno$feature)
rpkms <- matrix(NA, nrow = length(genes), ncol = length(allsamples))
rownames(rpkms) <- genes
genesone <- names(peakspergene)[peakspergene == 1]
colnames(rpkms)<-allsamples
for (gene in genesone) {
    rpkms[gene, allsamples] <- as.numeric(dfanno[dfanno$feature ==
                                                     gene, allsamples])
}
genesmore <- names(peakspergene)[peakspergene > 1]

rpkms <- dist_calc(method, dfanno, rpkms, genesmore, allsamples)
rawcounts <- matrix(as.numeric(rawcounts), nrow = nrow(rawcounts),
                    dimnames = dimnames(rawcounts))
rpkms <- matrix(as.numeric(rpkms), nrow = nrow(rpkms), dimnames = dimnames(rpkms))
vobj$rawcounts <- rawcounts
colnames(rpkms)<-allsamples
vobj$rpkms <- rpkms

#DEBUG ENDS

vobj<-vulcan.normalize(vobj)

load(file="networks/metabric-regulon-tfs.rda")

regulons <- loadVulcanNetworks()
suppressMessages(library("org.Hs.eg.db"))
list_eg2symbol<-as.list(org.Hs.egSYMBOL[mappedkeys(org.Hs.egSYMBOL)])

vobj_results<-list()

#test<-vulcan(vobj,       network=regulon, contrast=c("none","ER")    )

networks<-c("tcga","metabric") #,"laml")

for (network in networks) {
    vobj_results[[network]]<-vulcan(vobj,
                                    network=regulons[[network]],
                                    contrast=c("none","ER"),
                                    annotation=list_eg2symbol)
}
```

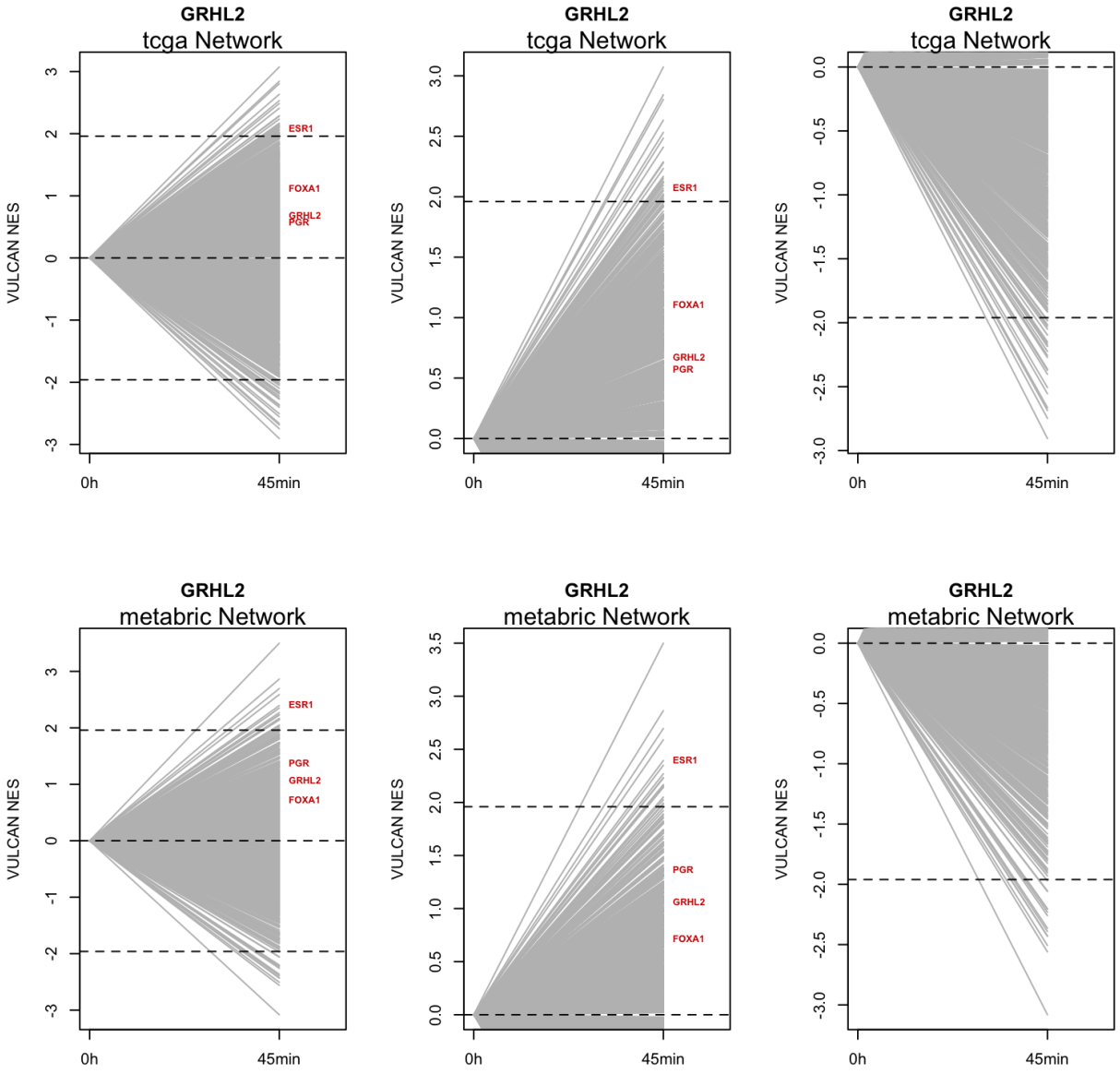
```
## Wed Jan 31 21:48:23 2018
## Computing the null model distribution by 1000 permutations.
```

```
## ------------------------------------------------------------
## Wed Jan 31 21:48:35 2018
## Computing regulon enrichment with aREA algorithm
##
## Estimating the normalized enrichment scores
## Wed Jan 31 21:49:06 2018
## Computing the null model distribution by 1000 permutations.
## ------------------------------------------------------------
## Wed Jan 31 21:49:18 2018
## Computing regulon enrichment with aREA algorithm
##
## Estimating the normalized enrichment scores
```

```
vobj_objects<-list(vobj_results)
names(vobj_objects)<-c("GRHL2")
networks<-c("tcga","metabric")
```

5.4.1 Transcription factor activity

```
plotVulcan <-function(vobj,threshold,network_title,title,plotTF,xlim,ylim) {
    threshold<-sign(threshold)*p2z(abs(threshold))
    network=vobj$mrs[,"NES"]
    tfs<-names(network)
    networkmat<-cbind(rep(0,length(network)),network[tfs])
    colnames(networkmat)<-c("0h","45min")
    matplot(t(networkmat),type="l",col="grey",ylab="VULCAN NES",xaxt="n",lty=1,main=title,xlim=xlim,yli
    axis(1,at=c(1:2),labels=colnames(networkmat))
    abline(h=c(0,threshold,-threshold),lty=2)
    text(2,networkmat[plotTF,2],label=names(networkmat[,2][plotTF]),pos=4,cex=0.6,font=2,col="red3")
    mtext(network_title)
}

par(mfrow=c(2,3))
for(network in networks){
    for (vobj_name in names(vobj_objects))
    {

        vobj<-vobj_objects[[vobj_name]]
        vobj<-vobj[[network]]
        #TFs<-getTFs(vobj)
        TFs=c("ESR1","PGR","FOXA1","GRHL2") #Overide TFS
        plotVulcan(vobj,0.05, paste0(network," Network"),vobj_name,TFs,xlim=c(1,2.3), ylim=c(min(vobj$m

        #TFs<-getTFs(vobj,0.05)
        plotVulcan(vobj,0.05, paste0(network," Network"),vobj_name,TFs,xlim=c(1,2.3), ylim=c(0,max(vobj

        #TFs<-getTFs(vobj,-0.05)
        plotVulcan(vobj,-0.05, paste0(network," Network"),vobj_name,TFs,xlim=c(1,2.3), ylim=c(min(vobj$
    }
}
```



5.4.2 METABRIC TGCA comparison

```
interect<-intersect(rownames(vobj_results$metabric$mrs),
                    rownames(vobj_results$tcga$mrs))
plot(-log10(vobj_results$metabric$mrs[interect,"pvalue"]),
     -log10(vobj_results$tcga$mrs[interect,"pvalue"]),
     pch=20, xlab="Metabric Network Enrichment Score", ylab="TCGA Network Enrichment Score",main="VULCA

filtered<-(log10(vobj_results$metabric$mrs[interect,"pvalue"])^2+log10(vobj_results$tcga$mrs[interect,"

text(-log10(vobj_results$metabric$mrs[interect,"pvalue"][filtered]),
     -log10(vobj_results$tcga$mrs[interect,"pvalue"][filtered])-0.1,
```

```
     labels=interect[filtered])

points(-log10(vobj_results$metabric$mrs[interect,"pvalue"][c("GATA3","PGR","FOXA1","ESR1","GRHL2")]),
       -log10(vobj_results$tcga$mrs[interect,"pvalue"][c("GATA3","PGR","FOXA1","ESR1","GRHL2")]),pch=20
text(-log10(vobj_results$metabric$mrs[interect,"pvalue"][c("GATA3","PGR","FOXA1","ESR1","GRHL2")]),
     -log10(vobj_results$tcga$mrs[interect,"pvalue"][c("GATA3","PGR","FOXA1","ESR1","GRHL2")])-0.1,col=
     labels=c("GATA3","PGR","FOXA1","ESR1","GRHL2"))
```

5.5 Motif Analysis

5.5.1 Export bed files

```
#Note homer needs this as Hg format so you need to change from "1" to "chr1" etc.
df<-as.data.frame(dba.report(GRHL2))
df$seqnames<-paste0("chr",df$seqnames)

kable(head(df))
```

	seqnames	start	end	width	strand	Conc	Conc_none	Conc_ER	Fold	p.value	FDR
12123	chr14	93959394	93959894	501	*	7.63	6.04	8.37	-2.33	0	0e+00
9047	chr12	75706270	75706770	501	*	6.74	4.93	7.52	-2.60	0	0e+00
24493	chr21	31529399	31529899	501	*	6.50	4.86	7.25	-2.39	0	0e+00
129	chr1	7447769	7448269	501	*	4.98	2.44	5.85	-3.41	0	0e+00
40054	chr9	75101020	75101520	501	*	6.94	4.80	7.76	-2.96	0	0e+00
28564	chr3	176953741	176954241	501	*	5.53	3.80	6.30	-2.50	0	1e-07

```
#write.bed(df,"bed/up.bed")
```

```
df_all<-as.data.frame(dba.report(GRHL2, th=1))
df_all$seqnames<-paste0("chr",df_all$seqnames)

kable(head(df_all))
```

	seqnames	start	end	width	strand	Conc	Conc_none	Conc_ER	Fold	p.value	FDR
12123	chr14	93959394	93959894	501	*	7.63	6.04	8.37	-2.33	0	0e+00
9047	chr12	75706270	75706770	501	*	6.74	4.93	7.52	-2.60	0	0e+00
24493	chr21	31529399	31529899	501	*	6.50	4.86	7.25	-2.39	0	0e+00
129	chr1	7447769	7448269	501	*	4.98	2.44	5.85	-3.41	0	0e+00
40054	chr9	75101020	75101520	501	*	6.94	4.80	7.76	-2.96	0	0e+00
28564	chr3	176953741	176954241	501	*	5.53	3.80	6.30	-2.50	0	1e-07

```
#write.bed(df_all,"bed/all.bed")
```

5.5.2 Homer

```


#Homer
```

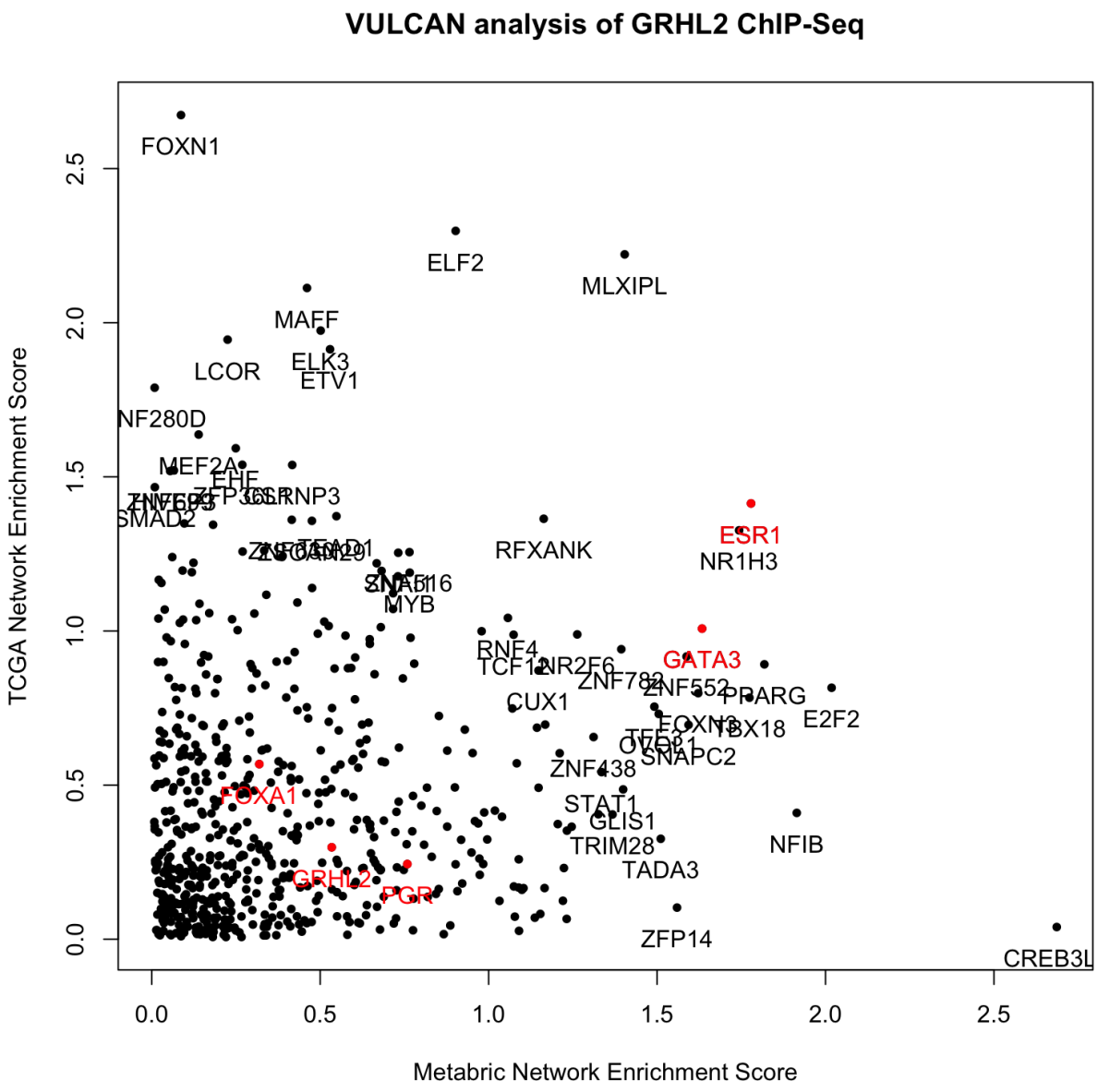


Figure 87: Comparision of results of VULCAN analysis for METABRIC and TGCA

```
mkdir motifAnalysis
cd motifAnalysis
findMotifsGenome.pl ../bed/up.bed hg38 grhl2UpSites
findMotifsGenome.pl ../bed/all.bed hg38 grhl2AllSites
```

5.5.3 Generate promoter locations

Generate Bed file of Promoters

```
require(TxDb.Hsapiens.UCSC.hg38.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene
p<-promoters(genes(txdb), upstream = 1500, downstream = 500)

write.bed<-function(df, filename){
    write.table(as.data.frame(df)[,1:3],
                filename, quote=FALSE, sep="\t",
                row.names=FALSE, col.names=FALSE)
}

write.bed(p,"bed/promoter.bed")
```

5.6 Ovlerap of GRHL2 binding sites with published data

5.6.1 Extract ChIA-PET Data

```
#ChIA-pet all 3 https://www.encodeproject.org/experiments/ENCSR000BZZ/
cd bed
multiIntersectBed -i *bed6_sorted.bed | awk '{if ($4 > 1) {print} }' > hglft_ChIA_combined.bed
```

5.6.2 Find overlapping sites

```
#p300 from Zwart, EMBO, 2011
#ER from Hurtado 2011
#gro-seq GSE43835
#Rest from Caroll MLL3 paper

cd bed
mkdir overlaps

#generate gro-seq interect
#file1=hglft_E2.40m.rep1.bed
#file2=hglft_E2.40m.rep2.bed
#output=gro-seq.bed
#bedtools intersect -sorted -a $file1 -b $file2 > overlaps/$output

file1=hglft_foxa1.bed
file2=up.bed
output=foxa1-up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output
```

```
file1=hglft_ER_Hurtado_2011.bed
output=er-up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

#file1=gro-seq.bed
#output=gro-up.bed
#bedtools intersect -a $file1 -b $file2 > overlaps/$output


file1=hglft_ChIA_combined.bed
output=ChIAPet_up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_h3k4me1.bed
output=h3k4me1-up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_h3k4me3.bed
output=h3k4me3-up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_p300_ctrl.bed
output=p300_ctrl_up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_p300_e2.bed
output=p300_e2_up.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output


#all peaks

file1=hglft_foxa1.bed
file2=all.bed
output=foxa1-all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_ER_Hurtado_2011.bed
output=er-all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

#file1=gro-seq.bed
#output=gro-all.bed
#bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_ChIA_combined.bed
output=ChIAPet_all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_h3k4me1.bed
output=h3k4me1-all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output
```

```
file1=hglft_h3k4me3.bed
output=h3k4me3-all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_p300_ctrl.bed
output=p300_ctrl_all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=hglft_p300_e2.bed
output=p300_e2_all.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

##Overlap with promoters

file1=overlaps/ChIAPet_up.bed
file2=promoter.bed
output=ChIAPet_up_prom.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output

file1=overlaps/ChIAPet_all.bed
file2=promoter.bed
output=ChIAPet_all_prom.bed
bedtools intersect -a $file1 -b $file2 > overlaps/$output
```

5.6.3 Count number of overlaping sites

```
cd bed/overlaps
wc -l *.bed > overlaps.txt
```

5.6.4 Table of number of overlaping sites for each factor

```
overlaps<-read.table("bed/overlaps/overlaps.txt")
kable(overlaps)
```

V1	V2
1764	ChIAPet_all.bed
109	ChIAPet_all_prom.bed
214	ChIAPet_up.bed
8	ChIAPet_up_prom.bed
5539	er-all.bed
318	er-up.bed
13422	foxa1-all.bed
255	foxa1-up.bed
33831	h3k4me1-all.bed
343	h3k4me1-up.bed
10234	h3k4me3-all.bed
55	h3k4me3-up.bed
7072	p300_ctrl_all.bed
155	p300_ctrl_up.bed
13588	p300_e2_all.bed

V1	V2
337	p300_e2_up.bed
87244	total

5.6.5 GRHL2 binding overlap with known factors

```
overlapsDF<-cbind(overlaps[1:16,],c('all','all','up','up','all','up','all','up','all','up','all','up','
overlapsDF<-cbind(overlapsDF,c('all','promoter','all','promoter',rep("all",12)))
overlapsDF<-cbind(overlapsDF,c(rep("None",14),"E2","E2"))
colnames(overlapsDF)<-c("Number","Factor","GRHL2","Feature","Treatment")
overlapsDF$Factor<-c(rep("ChIAPet",4),rep("ER",2),rep("FOXA1",2),rep("H3K4Me1",2),rep("H3K4Me3",2),rep(
overlapsDF$Number<-as.numeric(overlapsDF$Number)
ChIAPet<-overlapsDF[overlapsDF$Factor=='ChIAPet' & overlapsDF$Feature=='all',]
ChIAPet$Number<-overlapsDF[overlapsDF$Factor=='ChIAPet' & overlapsDF$Feature=='all',]$Number - overlapsD
ChIAPet$Feature<-c('enchancer','enchancer')
overlapsDF<-rbind(overlapsDF,ChIAPet)
overlapsDF<-data.frame(overlapsDF)
overlapsDF[overlapsDF$GRHL2=='up',]$Number<-(overlapsDF[overlapsDF$GRHL2=='up',]$Number/355)*100 #355 i
overlapsDF[overlapsDF$GRHL2=='all',]$Number<-(overlapsDF[overlapsDF$GRHL2=='all',]$Number/42721)*100 #T

library(lattice)

barchart(Number ~ Factor ,data=overlapsDF[overlapsDF$Factor != 'ChIAPet',],
         par.settings = simpleTheme(col=c("lightblue","pink")),
         group=GRHL2,auto.key=list(space="top", columns=2, text=c('All GRHL2 Sites','GRHL2 Sites Respons
         main="GRHL2 Binding Overlap with Known Factors", ylab="Percentage of GRHL2 sites that overlap"
         xlab="Factor")
```

5.6.6 GRHL2 binding overlap with P300 binding sites

```
barchart(Number ~ Treatment ,data=overlapsDF[overlapsDF$Factor=='P300',],
         group=GRHL2,
         auto.key=list(space="top",columns=2, text=c('All GRHL2 Sites','GRHL2 Responsive to E2')),
         par.settings = simpleTheme(col=c("lightblue","pink")),
         main="GRHL2 overlap with P300 binding sites",
         xlab="P300 ChIP-Seq Condition",
         ylab="Percentage of GRHL2 sites that overlap")
```

5.6.7 GRHL2 overlap with ChIA-PET sites

```
barchart(Number ~ Feature ,data=overlapsDF[overlapsDF$Factor=='ChIAPet',],
         group=GRHL2,
         auto.key=list(space="top",columns=2, text=c('All GRHL2 Sites','GRHL2 Sites Responsive to E2'))
         par.settings = simpleTheme(col=c("lightblue","pink")),
         main="GRHL2 overlap with ChIA-PET sites",
         xlab="Genomic Feature ChIP-Seq Condition",
         ylab="Percentage of GRHL2 sites that overlap")
```

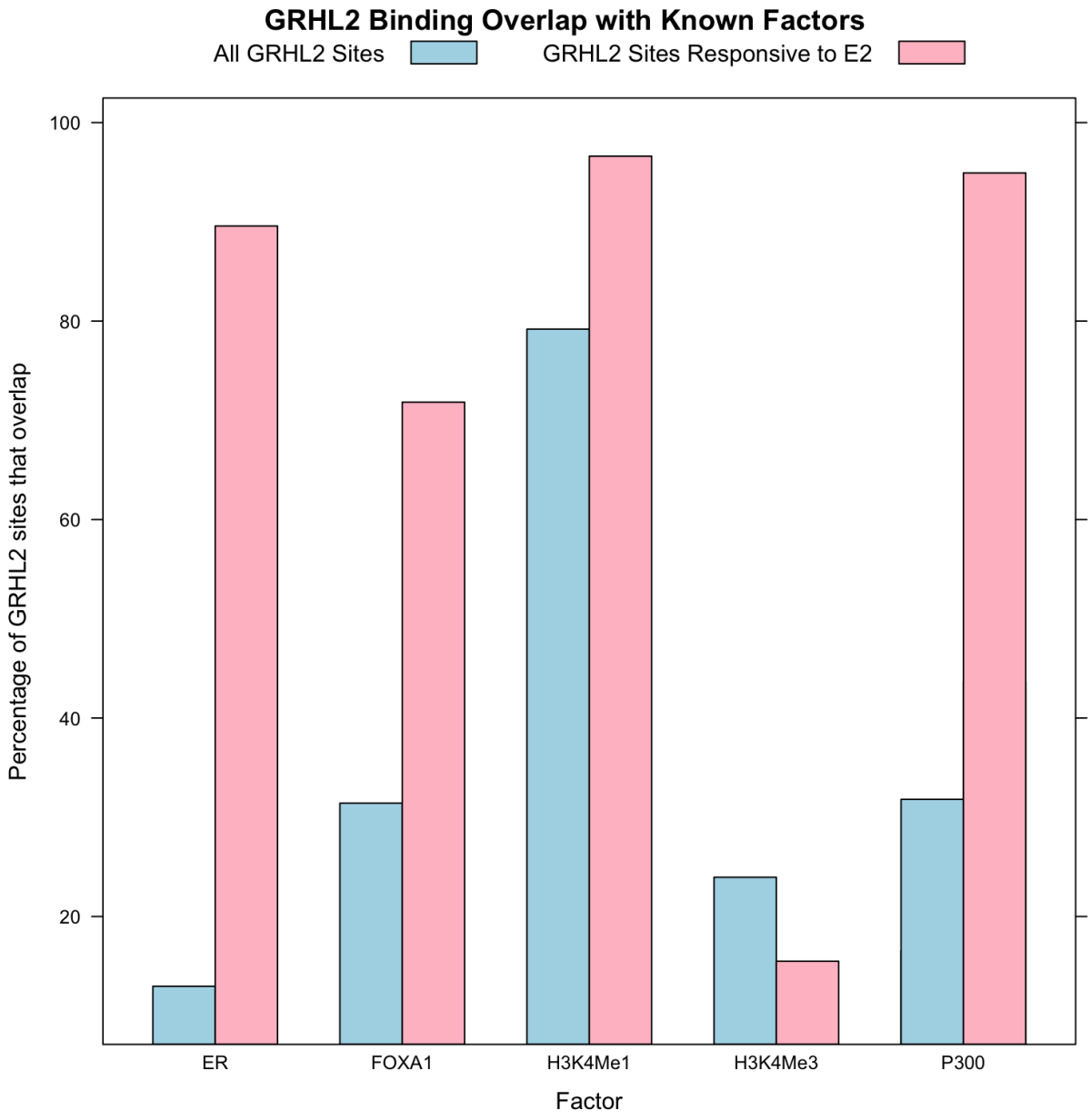


Figure 88: GRHL2 binding overlap with known factors

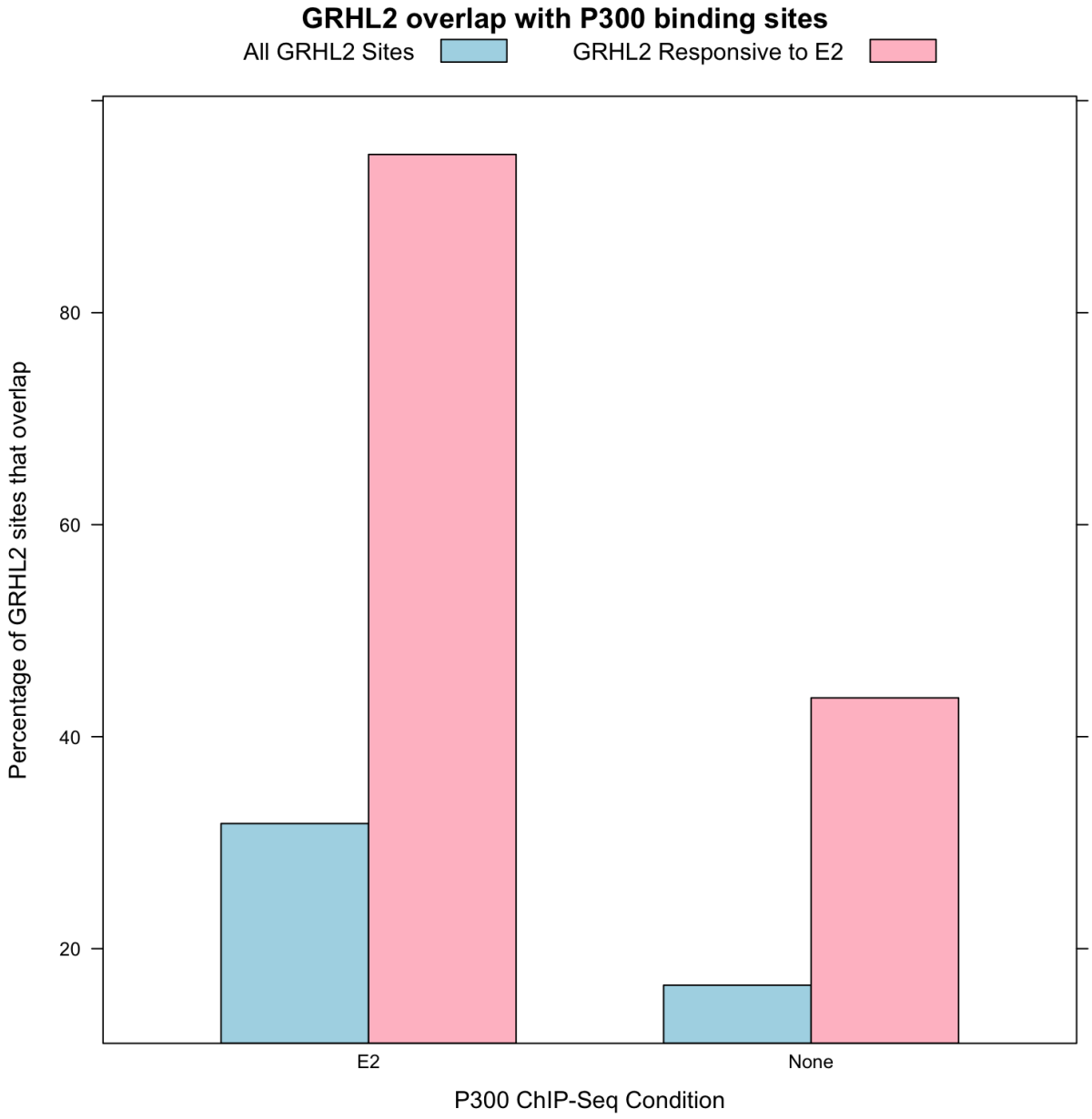


Figure 89: GRHL2 binding overlap with P300 binding sites

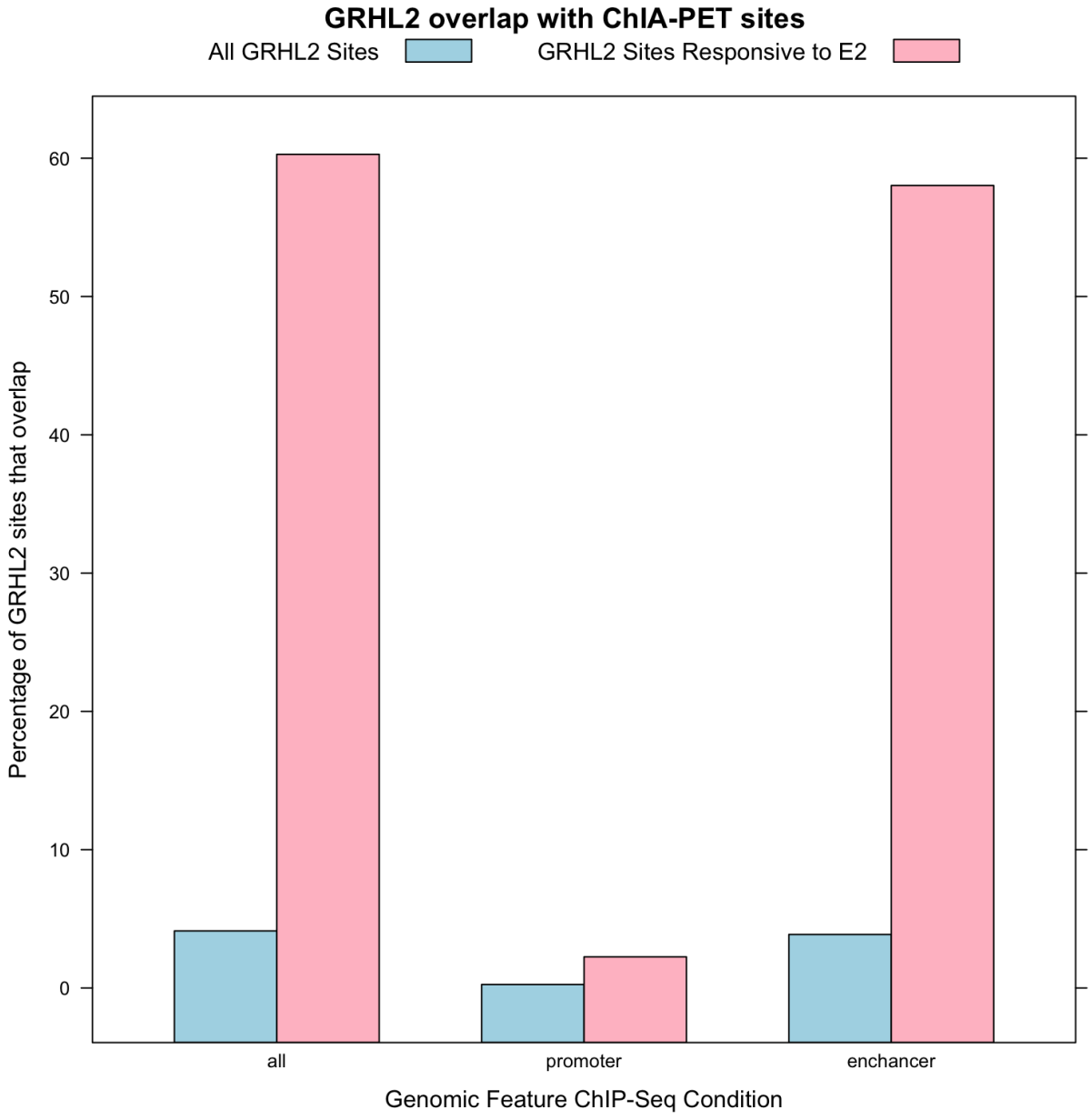


Figure 90: GRHL2 overlap with ChIA-PET sites

5.6.8 GRHL2 overlap with Gro-SEQ data

```
allPeaks<-read.csv("txt/All_GRHL2_peaks_GSE43836.csv")
upPeaks<-read.csv("txt/Up_GRHL2_peaks_GSE43836.csv")


par(mfrow=c(1,2))

plot(allPeaks[1:2],ylim=c(0,6),type="n", lwd=2, ylab="Read Depth", main="All GRHL2 Sites")
lines(allPeaks[c(1,2)], lwd=2, col="lightblue")
lines(allPeaks[c(1,3)], lwd=2,  col="lightblue")
lines(allPeaks[c(1,4)], lwd=2,  col="pink")
lines(allPeaks[c(1,5)],  lwd=2, col="pink")

plot(upPeaks[c(1,5)],ylim=c(0,6),type="n", ylab="Read Depth", main="E2 responsive GRHL2 Sites")
lines(upPeaks[c(1,2)], lwd=2,col="lightblue")
lines(upPeaks[c(1,3)], lwd=2, col="lightblue")
lines(upPeaks[c(1,4)], lwd=2, col="pink")
lines(upPeaks[c(1,5)],  lwd=2,col="pink")
```

6 GRHL2 qPLEX-RIME Analysis

6.1 Introduction

This report provides a summary of the results of your proteomics experiment.

Your experiment has **11** samples. **9032** peptides from **1514** unique proteins were captured in the experiment. Each protein is represented by **1** to **255** peptides, but median number of peptides per protein was **2**.

6.2 Raw data QC

6.2.1 Coverage plot

Figure 1 shows the coverage of the bait protein, GRHL2, in terms of peptides detected.

6.2.2 Intensity Plot

Figure 2 shows the distribution of raw peptide intensities for each sample.

6.2.3 Peptide intensities for GHRL2

Figure 3 shows the raw intensities for each peptide detected for the bait protein GRHL2 in each sample.

6.2.4 Correlation Plot

Figure 4 shows a correlation matrix to visualize the level of linear association of samples within and between groups based on the raw peptide intensities.

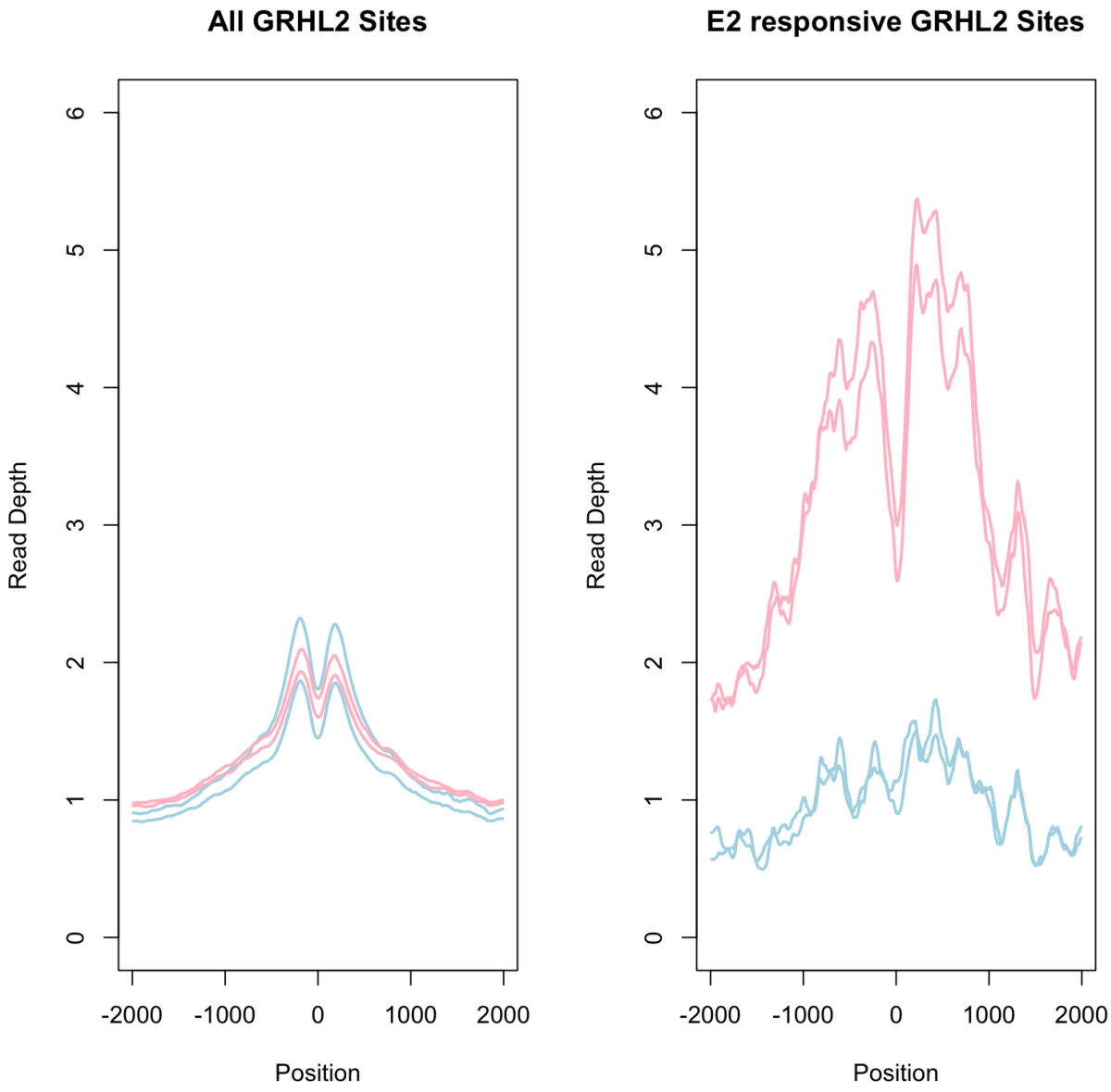


Figure 91: Gro-Seq data at GRHL2 sites. Blue is control samples, pink is E2 treated samples. Data from GSE43836.

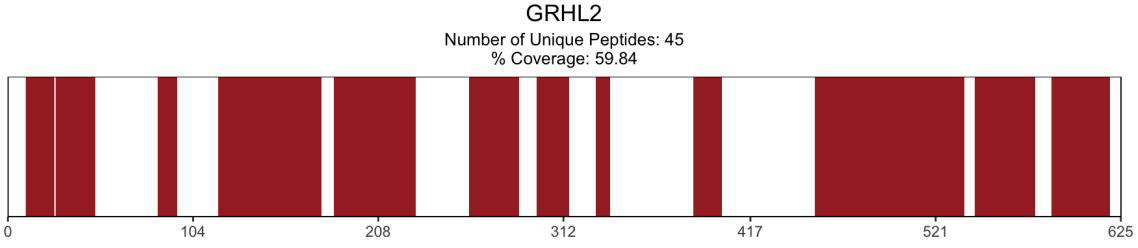


Figure 92: **Figure 1. Plot of peptide coverage for GRHL2 (UniprotID: Q6ISB3) **

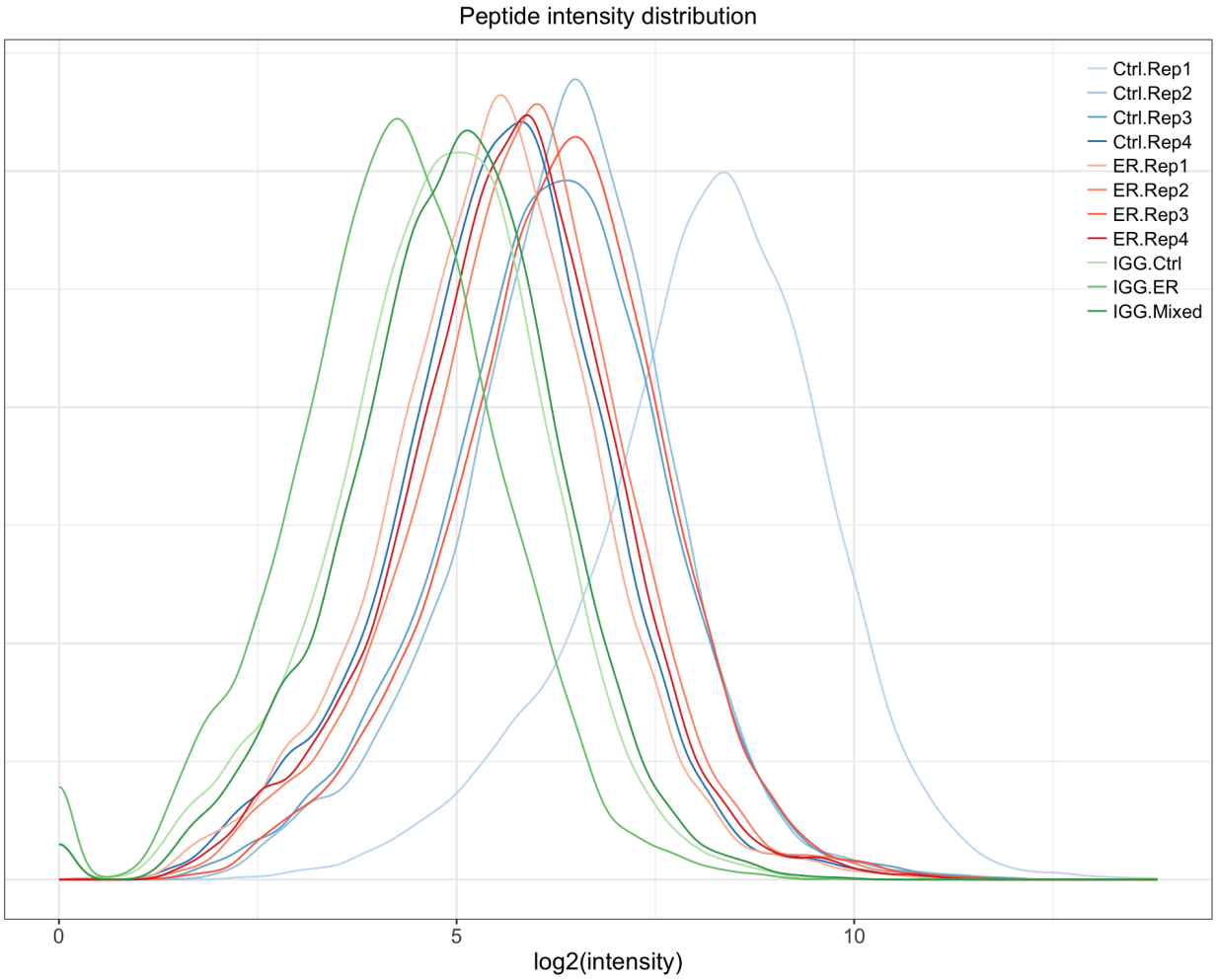


Figure 93: **Figure 2. Raw intensities plot. **

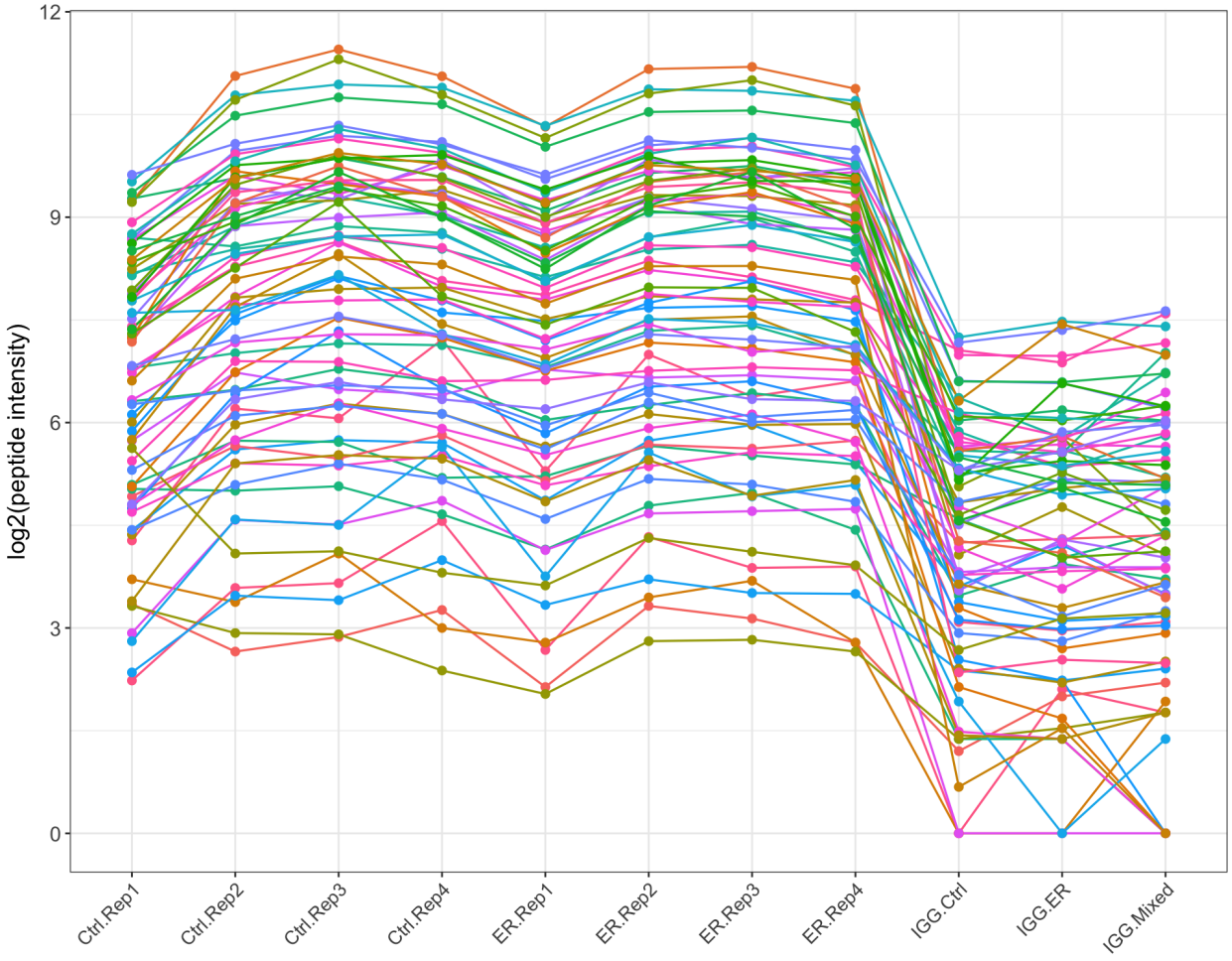


Figure 94: **Figure 3. Peptide intensities for GRHL2. **

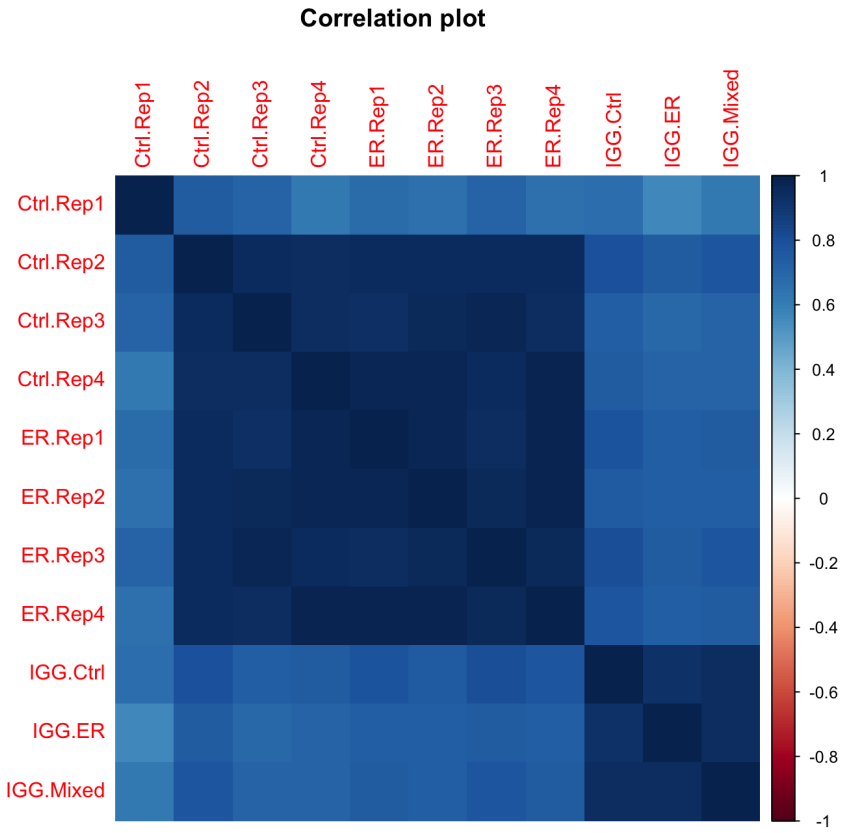


Figure 95: **Figure 4. Correlation plot based on raw intensities. **

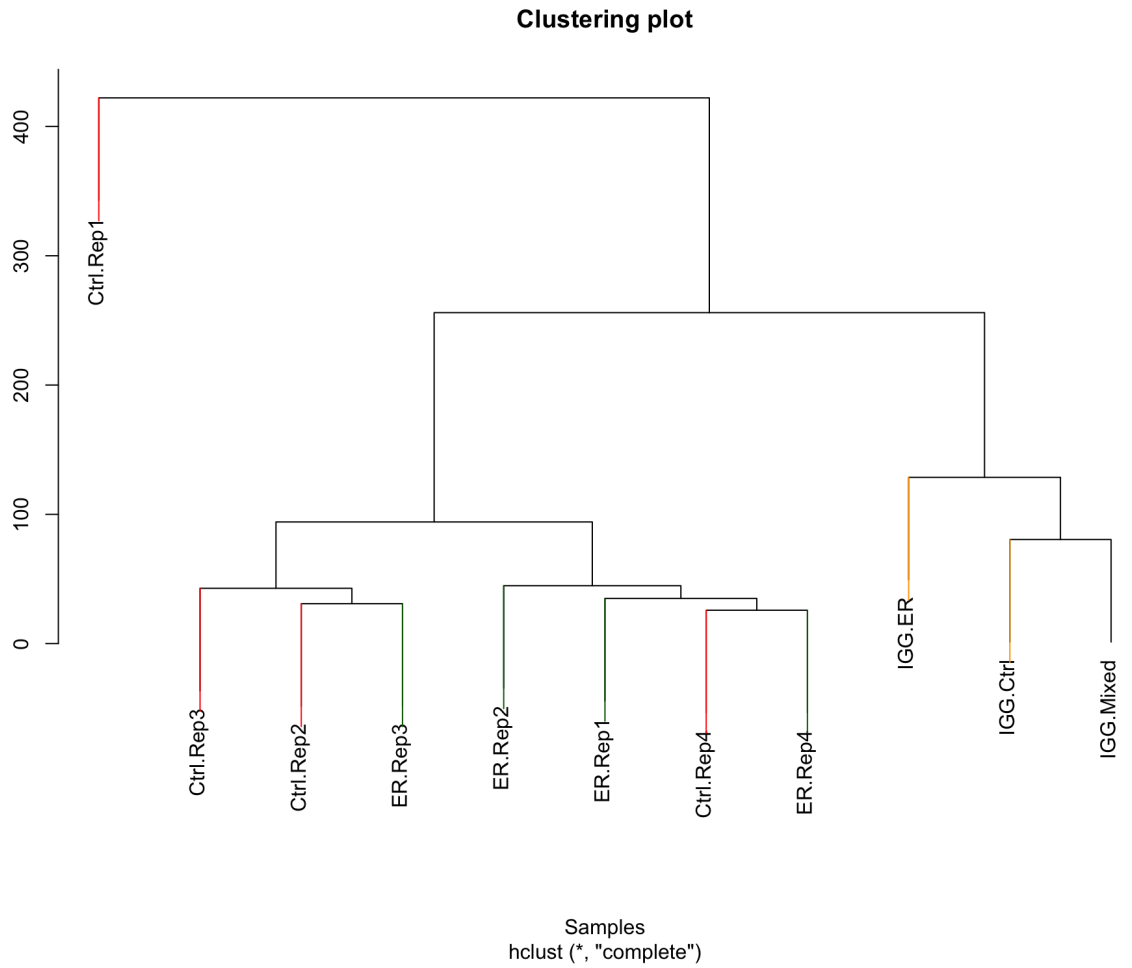


Figure 96: **Figure 5. Hierachical clustering based on raw intensities. **

6.2.5 Hierichical clustering dendrogram

Figure 5 shows a dendrogram displaying the hierarchical relationship among samples. The vertical axis shows the dissimilarity (measured by means of the Euclidean distance) between samples: similar samples appear on the same branches. Colors correspond to sample groups.

6.2.6 PCA Plot

Figure 6 shows a visual representation of the scaled loading of the first two dimensions of a principle component analysis of the raw peptide intensities.

List of 1 $ aspect.ratio: num 1 - attr(, *"class")= chr [1:2] "theme" "gg" - attr(*, "complete")= logi FALSE - attr(*, "validate")= logi TRUE

6.2.7 QC Conclusion

The sample Ctrl rep 1 appears to be an outlier. The differential anlaysis was carried out twice, first with this sample included and then again with the sample excluded.

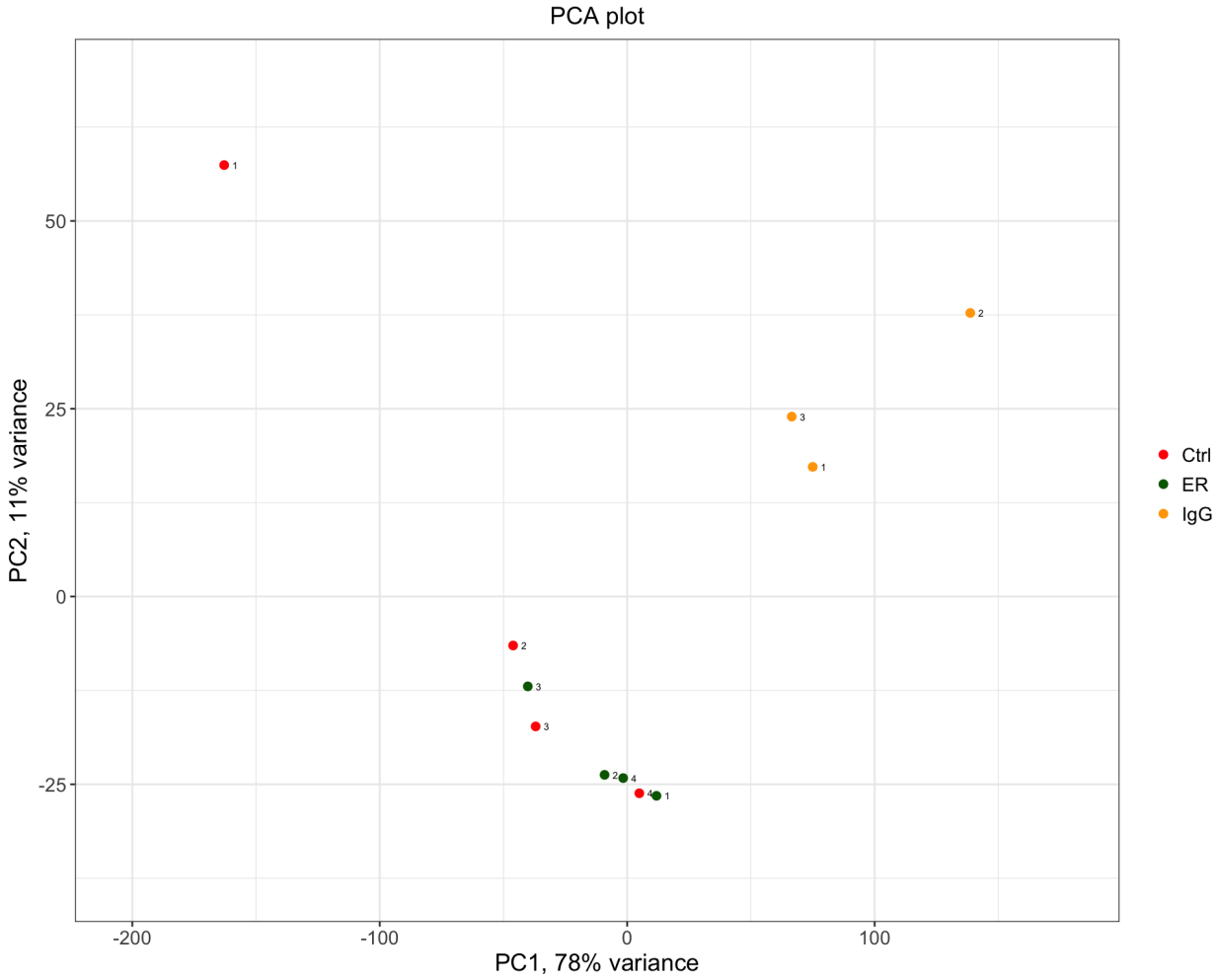


Figure 97: **Figure 6. Principle Component Analysis based on raw intensities. **

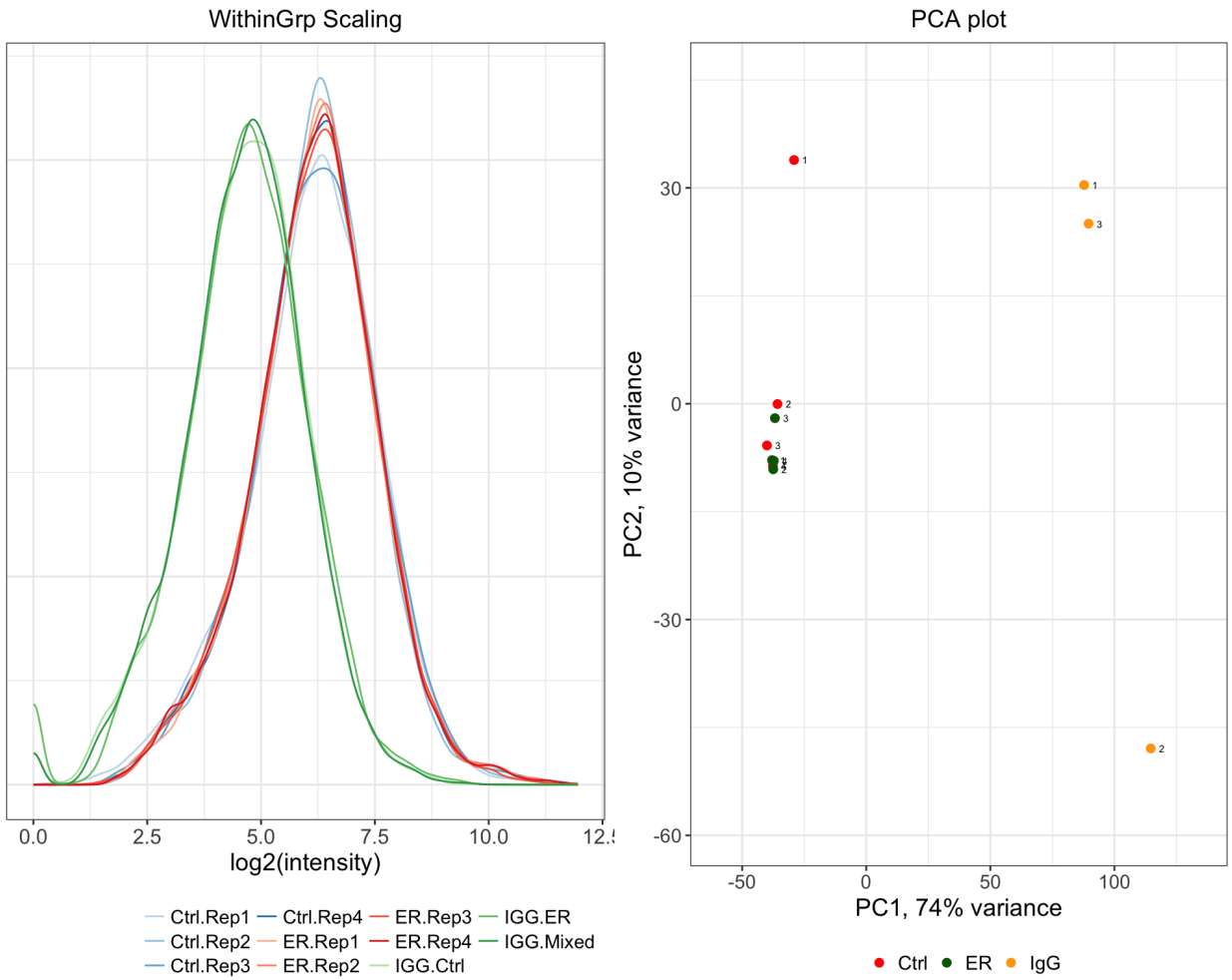


Figure 98: **Figure 7. Effects of normalisation on peptide intensities. **

6.3 Full data set analysis

The following section shows the results for analysis of all 8 samples.

6.3.1 Effect of within group normalisation

Normalisation was carried out using median scaling. The experimental samples and the IgG control samples were normalised separately. Figure 7 show the effects of normalisation on the intensity plots and the principle component analysis.

6.3.2 Differential analysis results

Figure 8 shows differential abundancy results. No proteins were statistically differentially abundant. Figure 7 shows an MA plot and a volcano plot of the differental analysis results.

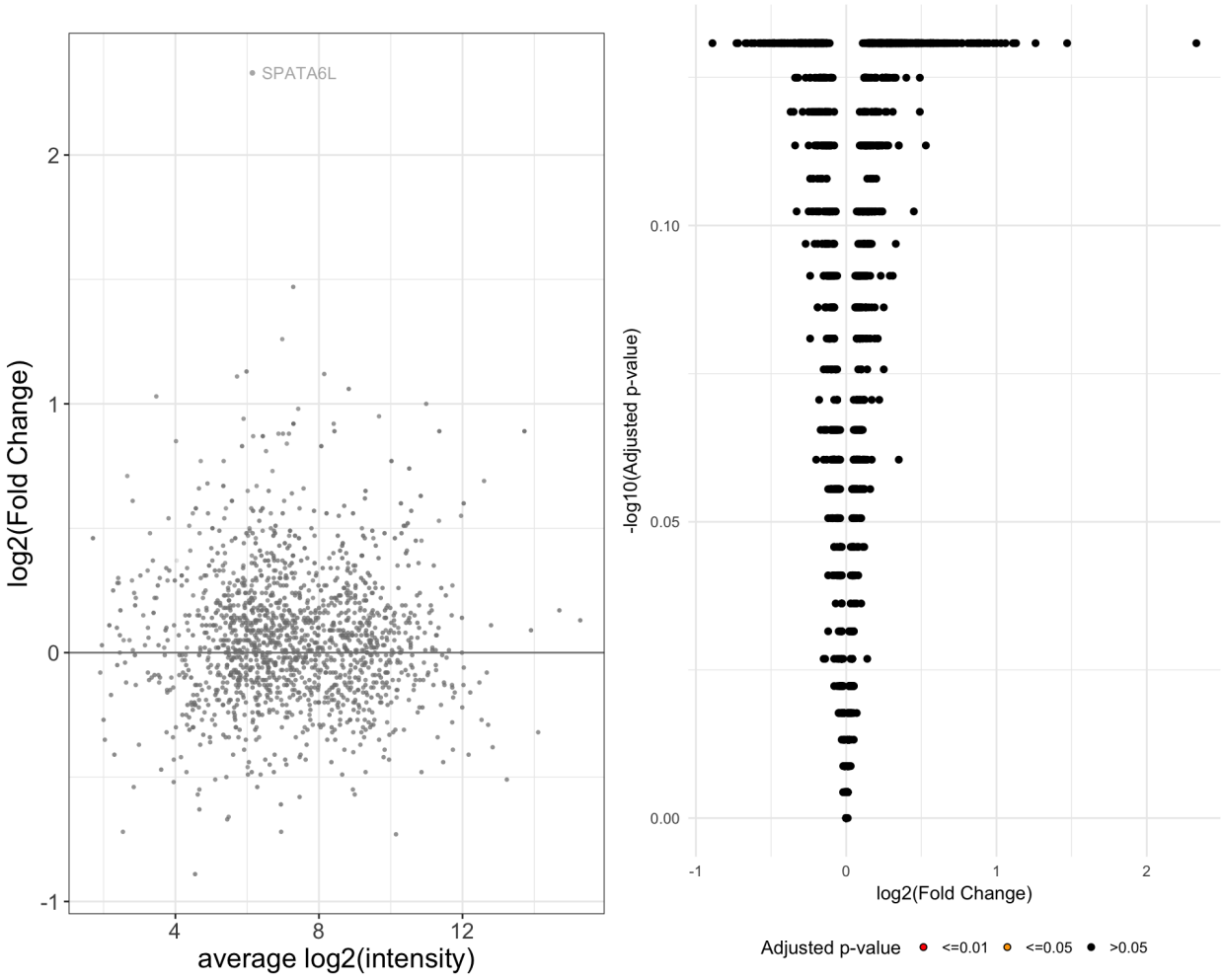


Figure 99: **Figure 9. MA plot and volcano plot of differential protein abundancy. **

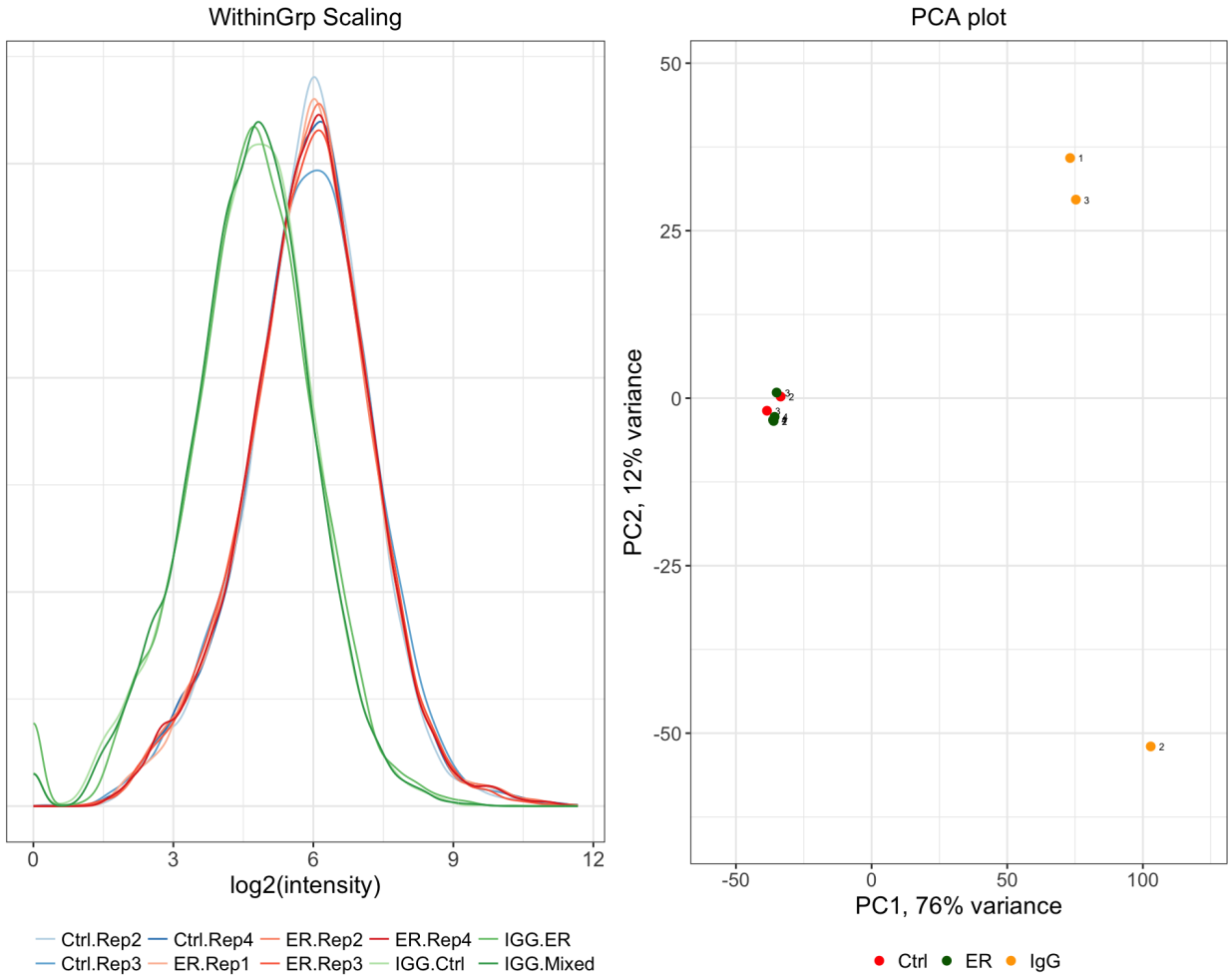


Figure 100: **Figure 10. Effects of normalisation on peptide intensities. **

6.4 Remove sample Ctrl 1

The following section shows the same analysis, but with the Ctrl rep 1 sample removed

6.4.1 Effect of within group normalisation

Normalisation was carried out using median scaling. The experimental samples and the IgG control samples were normalised separately. Figure 10 show the effects of normalisation on the intensity plots and the principle component analysis.

6.4.2 ER - Ctrl

Figure 11 shows differential abundancy results for the contrasts ER_v_Ctrl. Figure 10 shows an MA plot and a volcano plot of the differental analysis results

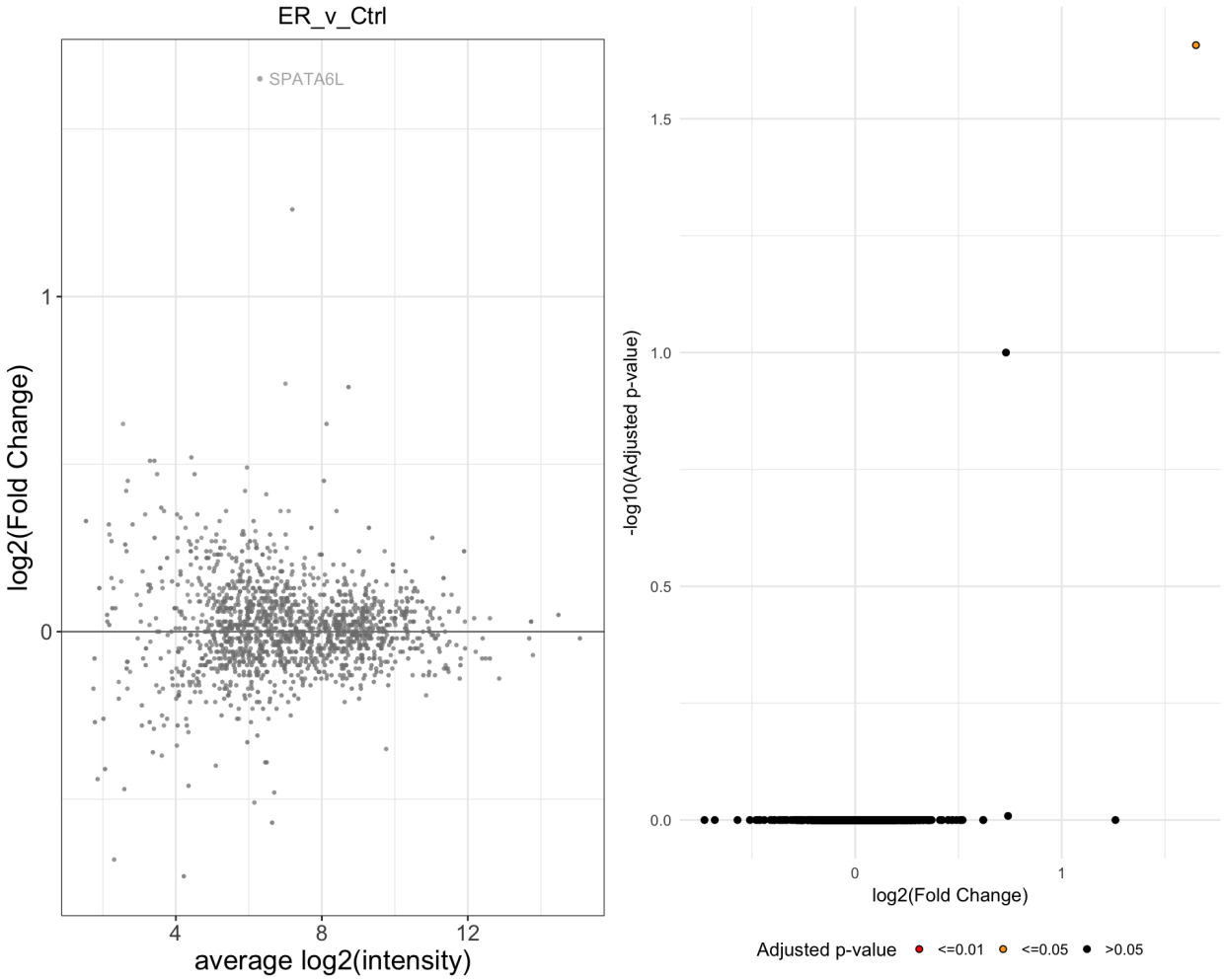


Figure 101: **Figure 12. MA plot and volcano plot of differential protein abundancy. **

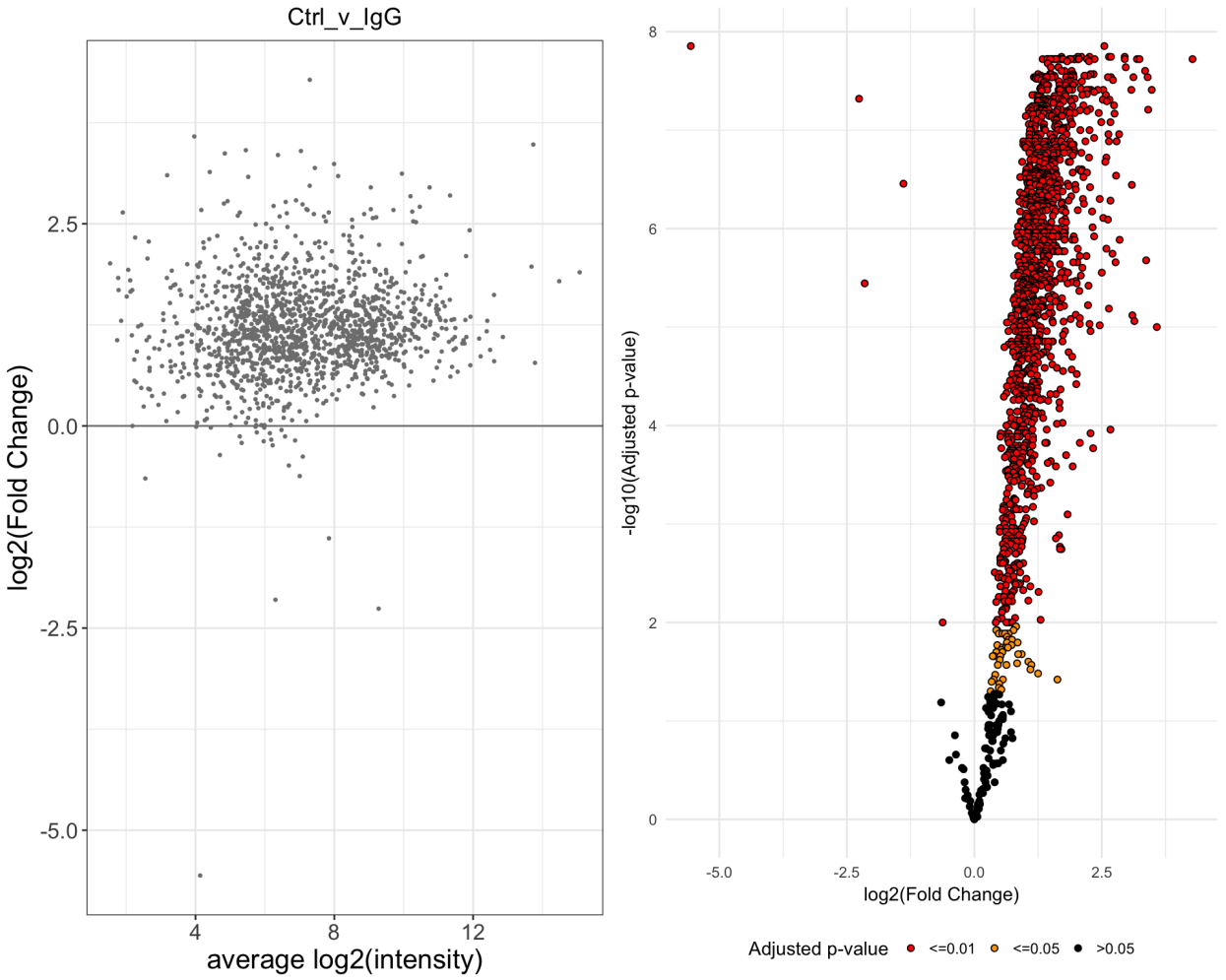


Figure 102: **Figure 15. MA plot and volcano plot of differential protein abundancy. **

6.4.3 Ctrl - IgG

Figure 13 shows differential abundancy results for the contrasts Ctrl_v_IgG. Figure 14 shows an MA plot and a volcano plot of the differental analysis results.

6.4.4 ER - IgG

Figure 16 shows differential abundancy results for the contrasts ER_v_IgG. Figure 17 shows an MA plot and a volcano plot of the differental analysis results.

7 GRHL2 qPLEX-RIME Analysis

7.1 Effect of GRHL2 overexpression on eRNA transcription

```
library("ggpubr")
```

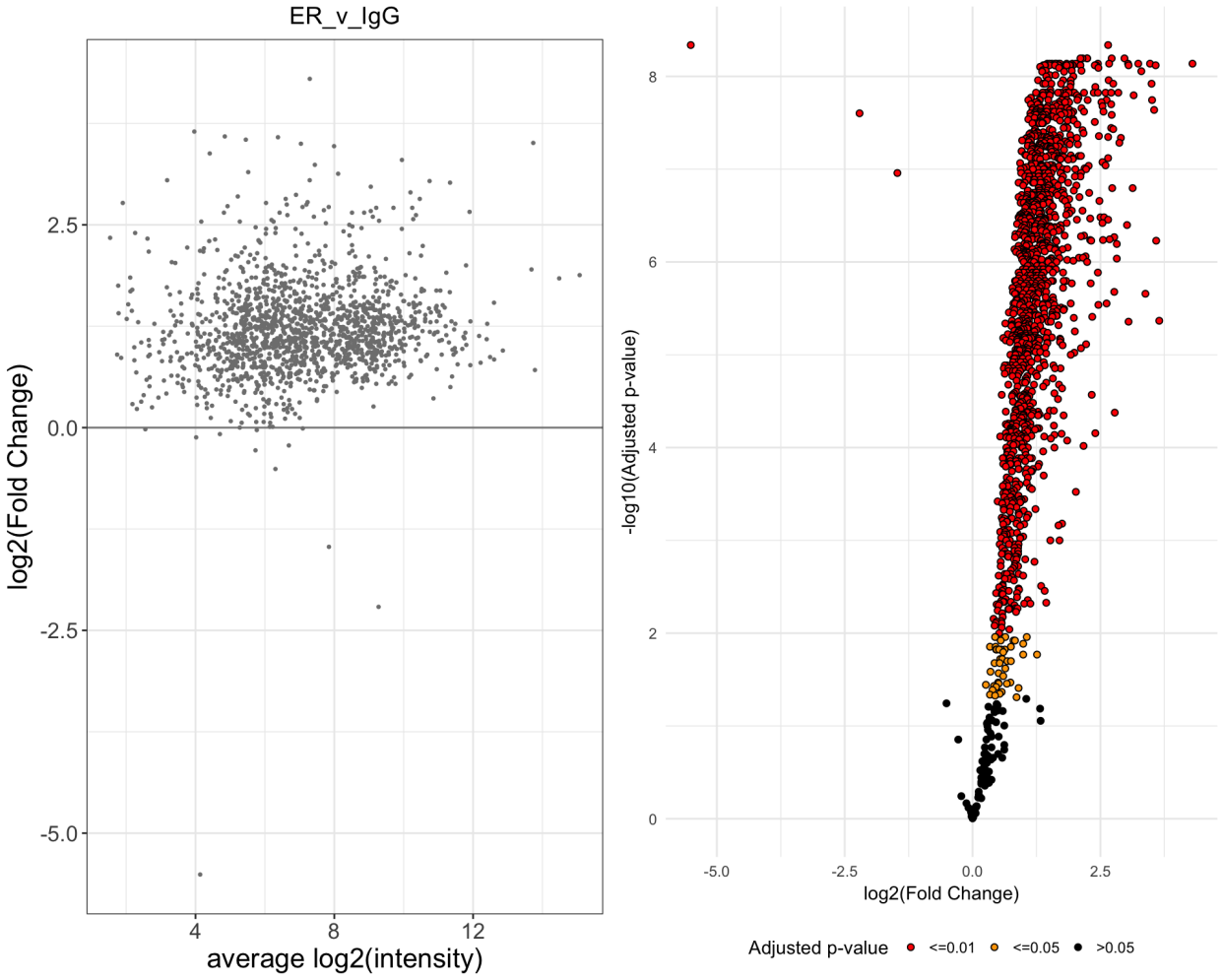


Figure 103: **Figure 18. MA plot and volcano plot of differential protein abundancy. **

```
eRNA<-read.csv("txt/eRNA_values.csv")

oe<-eRNA[eRNA$Experiment=="overexpression",]

oe_comparisons <- list( c("Control","GRHL2") )
oe_p <- ggboxplot(oe, x = "Treatment", y = "eRNA",
               add = "jitter", color="Treatment",
               facet.by="Gene", ylab="Relative eRNA levels")
oe_p <- oe_p + stat_compare_means(comparisons = oe_comparisons,
                        method = "wilcox.test" ,
                        paired=TRUE)
oe_p
```

7.2 Effect of GRHL2 knockdown on eRNA transcription

```
si<-eRNA[eRNA$Experiment=="knockdown",]

si_comparisons <- list( c("siCtrl","siGRHL2") )
p <- ggboxplot(si, x = "Treatment", y = "eRNA",
               add = "jitter", color="Treatment",
               facet.by="Gene", ylab="Relative eRNA levels")
p + stat_compare_means(comparisons = si_comparisons,
                        method = "wilcox.test" ,
                        paired = TRUE)
```

7.2.1 siRNA combined test

```
wilcox.test(si$eRNA[si$Treatment=='siCtrl'],
            si$eRNA[si$Treatment=='siGRHL2'],
            paired=TRUE, alternative="less")
```

```
##
##  Wilcoxon signed rank test
##
## data:  si$eRNA[si$Treatment == "siCtrl"] and si$eRNA[si$Treatment == "siGRHL2"]
## V = 45, p-value = 0.04071
## alternative hypothesis: true location shift is less than 0
```

8 Technical Session Info

The following code describes the R environment used to generate this document and will help making it fully reproducible should there be future updates in any of the packages.

```
sessionInfo()
```

```
## R version 3.4.1 (2017-06-30)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: OS X El Capitan 10.11.6
##
## Matrix products: default
```

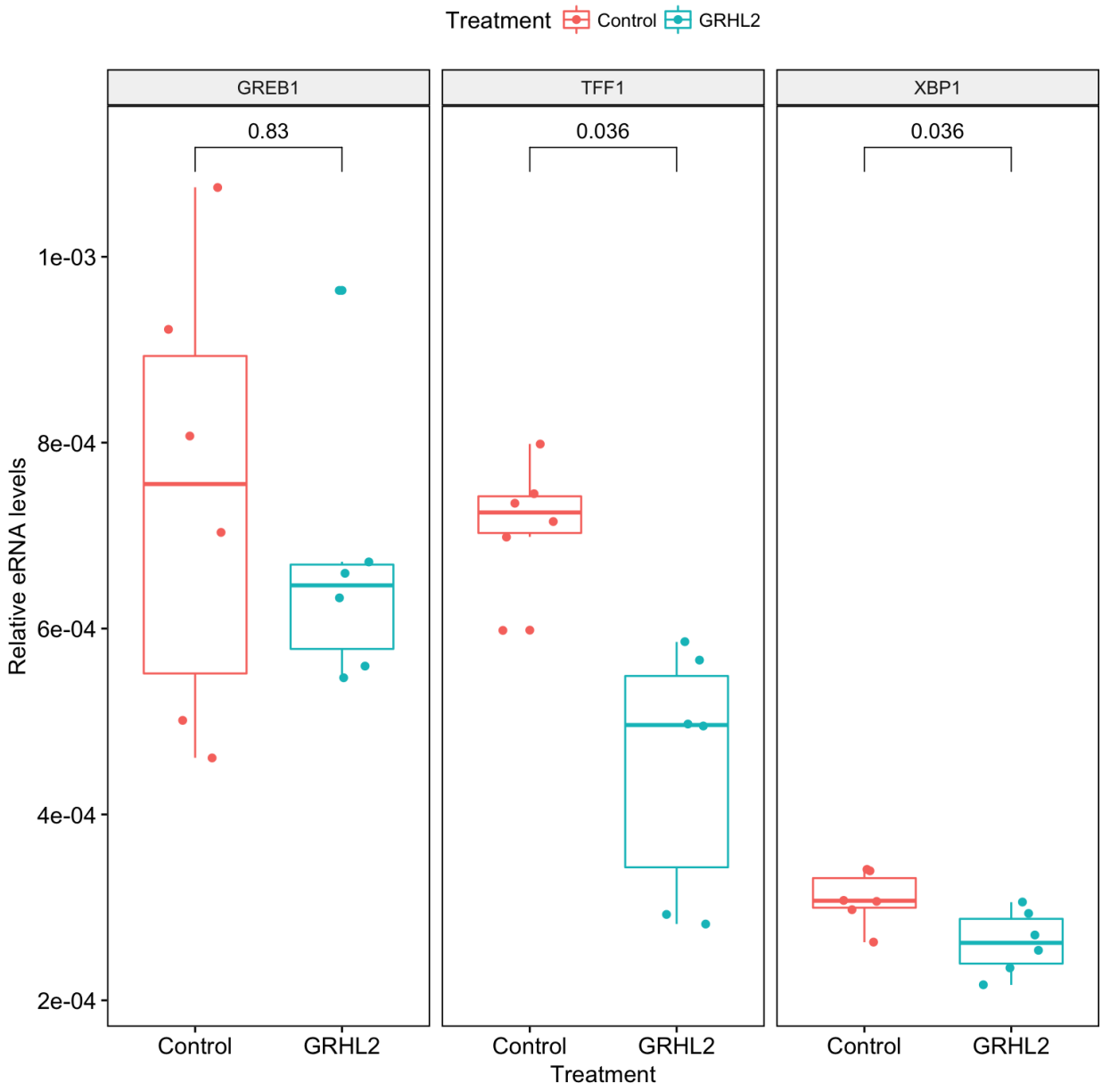


Figure 104: Effect of Overexpression of GRHL2 on eRNA at E2 responsive binding sites.

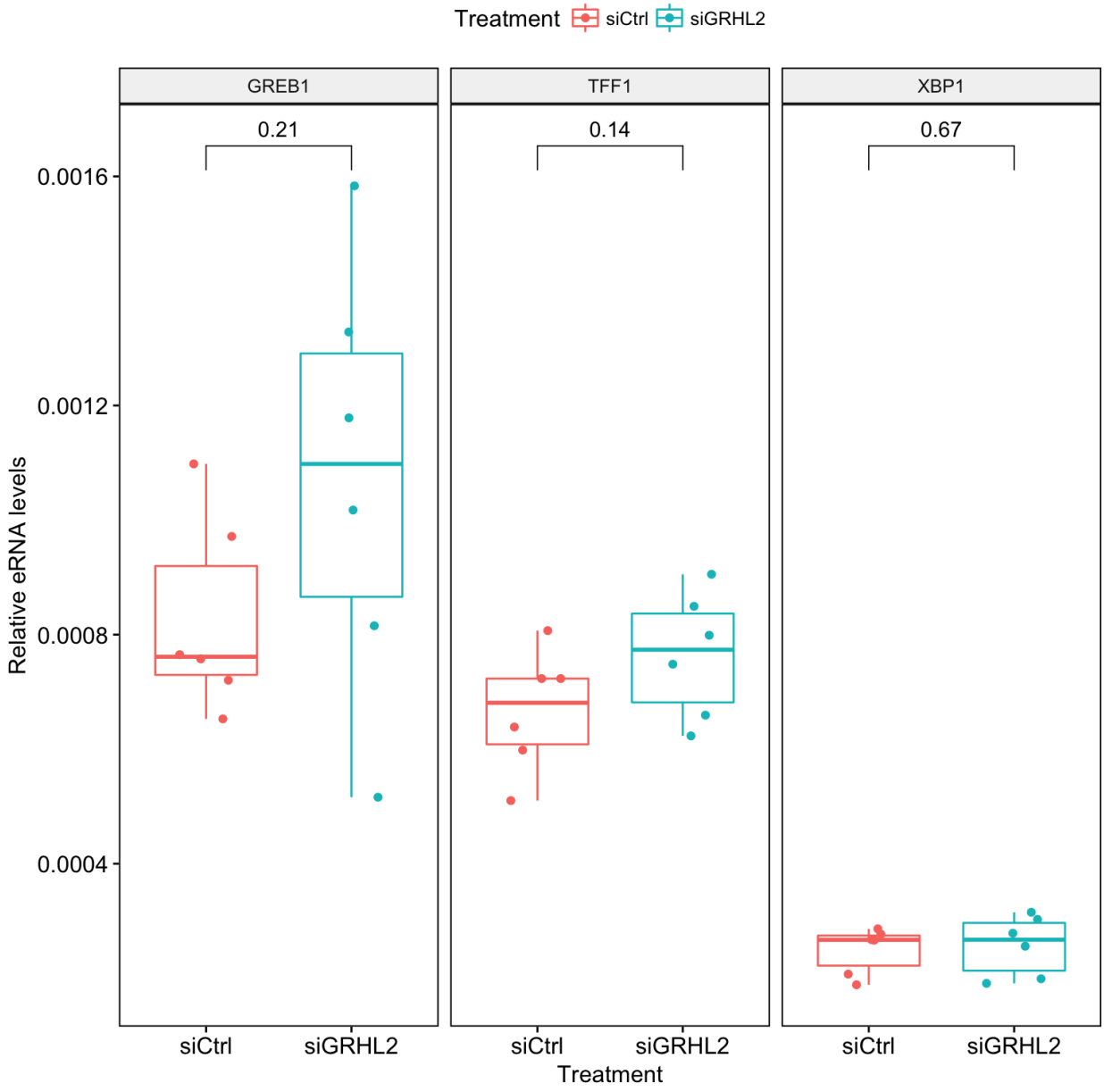


Figure 105: Effect of knockdown of GRHL2 on eRNA at E2 responsive binding sites.

```
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
##  [1] grid      stats4    parallel  stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] ggpubr_0.1.6
##  [2] rmarkdown_1.6
##  [3] DT_0.2
##  [4] readxl_1.0.0
##  [5] qPLEXanalyzer_1.0.0
##  [6] statmod_1.4.30
##  [7] magrittr_1.5
##  [8] plyr_1.8.4
##  [9] UniProt.ws_2.16.0
## [10] RCurl_1.95-4.8
## [11] bitops_1.0-6
## [12] RSQLite_2.0
## [13] TxDb.Hsapiens.UCSC.hg38.knownGene_3.4.0
## [14] knitr_1.17
## [15] bindrcpp_0.2
## [16] preprocessCore_1.38.1
## [17] limma_3.32.10
## [18] xtable_1.8-2
## [19] kfigr_1.2
## [20] ggplot2_2.2.1
## [21] MSnbase_2.2.0
## [22] ProtGenerics_1.8.0
## [23] mzR_2.10.0
## [24] Rcpp_0.12.13
## [25] tibble_1.3.4
## [26] dplyr_0.7.4
## [27] tidyr_0.7.2
## [28] GeneNet_1.2.13
## [29] fdrtool_1.2.15
## [30] longitudinal_1.1.12
## [31] corpcor_1.6.9
## [32] gridExtra_2.3
## [33] org.Hs.eg.db_3.4.1
## [34] vulcan_0.99.36
## [35] caTools_1.17.1
## [36] csaw_1.10.0
## [37] BiocParallel_1.10.1
## [38] wordcloud_2.5
## [39] RColorBrewer_1.1-2
## [40] zoo_1.8-0
## [41] DESeq_1.28.0
## [42] lattice_0.20-35
## [43] locfit_1.5-9.1
```

```
## [44] TxDb.Hsapiens.UCSC.hg19.knownGene_3.2.2
## [45] GenomicFeatures_1.28.5
## [46] AnnotationDbi_1.38.2
## [47] gplots_3.0.1
## [48] ChIPpeakAnno_3.10.2
## [49] VennDiagram_1.6.17
## [50] futile.logger_1.4.3
## [51] Biostrings_2.44.2
## [52] XVector_0.16.0
## [53] viper_1.10.0
## [54] DiffBind_2.5.6
## [55] SummarizedExperiment_1.6.5
## [56] DelayedArray_0.2.7
## [57] matrixStats_0.52.2
## [58] Biobase_2.36.2
## [59] GenomicRanges_1.28.6
## [60] GenomeInfoDb_1.12.3
## [61] IRanges_2.10.5
## [62] S4Vectors_0.14.7
## [63] BiocGenerics_0.22.1
##
## loaded via a namespace (and not attached):
##   [1] sendmailR_1.2-1               highcharter_0.5.0
##   [3] rtracklayer_1.36.6            AnnotationForge_1.18.2
##   [5] vioplot_0.2                   acepack_1.4.1
##   [7] bit64_0.9-7                   data.table_1.10.4-2
##   [9] rpart_4.1-11                  hwriter_1.3.2
##  [11] AnnotationFilter_1.0.0        doParallel_1.0.11
##  [13] lambda.r_1.2                  rlist_0.4.6.1
##  [15] bit_1.1-12                    lubridate_1.7.1
##  [17] httpuv_1.3.5                  assertthat_0.2.0
##  [19] amap_0.8-14                   evaluate_0.10.1
##  [21] BiocInstaller_1.26.1          Rhtslib_1.8.0
##  [23] igraph_1.1.2                  DBI_0.7
##  [25] geneplotter_1.54.0            quantmod_0.4-12
##  [27] htmlwidgets_0.9               purrr_0.2.4
##  [29] corrplot_0.84                 backports_1.1.1
##  [31] BatchJobs_1.6                 geepack_1.2-1
##  [33] annotate_1.54.0               biomaRt_2.32.1
##  [35] imputeLCMD_2.0                TTR_0.23-2
##  [37] ensembldb_2.0.4               Cairo_1.5-9
##  [39] fail_1.3                      BSgenome_1.44.2
##  [41] checkmate_1.8.4               GenomicAlignments_1.12.2
##  [43] xts_0.10-1                    mnormt_1.5-5
##  [45] sparcl_1.0.3                  cluster_2.0.6
##  [47] segmented_0.5-2.2             lazyeval_0.2.0
##  [49] genefilter_1.58.1             edgeR_3.18.1
##  [51] pkgconfig_2.0.1               slam_0.1-40
##  [53] labeling_0.3                  nlme_3.1-131
##  [55] nnet_7.3-12                   bindr_0.1
##  [57] rlang_0.1.2                   sandwich_2.4-0
##  [59] seqinr_3.4-5                  affyio_1.46.0
##  [61] GOstats_2.42.0                AnnotationHub_2.8.3
##  [63] cellranger_1.1.0              rprojroot_1.2
```

```
##  [65] graph_1.54.0                Matrix_1.2-11
##  [67] base64enc_0.1-3             pheatmap_1.0.8
##  [69] png_0.1-7                   rjson_0.2.15
##  [71] KernSmooth_2.23-15          geeM_0.10.0
##  [73] blob_1.1.0                  qvalue_2.8.0
##  [75] stringr_1.2.0               regioneR_1.8.1
##  [77] ShortRead_1.34.2            brew_1.0-6
##  [79] ggsignif_0.4.0              tmvtnorm_1.4-10
##  [81] scales_0.5.0                memoise_1.1.0
##  [83] GSEABase_1.38.2             cp4p_0.3.5
##  [85] gdata_2.18.0                zlibbioc_1.22.0
##  [87] compiler_3.4.1              pcaMethods_1.68.0
##  [89] DESeq2_1.16.1               Rsamtools_1.28.0
##  [91] ade4_1.7-8                  affy_1.54.0
##  [93] systemPipeR_1.10.2          DAPARdata_1.4.0
##  [95] Category_2.42.1             htmlTable_1.9
##  [97] Formula_1.2-2               MASS_7.3-47
##  [99] tidyselect_0.2.3            vsn_3.44.0
## [101] stringi_1.1.5               highr_0.6
## [103] yaml_2.1.14                 norm_1.0-9.5
## [105] latticeExtra_0.6-28         MALDIquant_1.17
## [107] ggrepel_0.7.0               tools_3.4.1
## [109] DAPAR_1.8.7                 imp4p_0.5
## [111] foreach_1.4.4               foreign_0.8-69
## [113] idr_1.2                     mzID_1.14.0
## [115] digest_0.6.12               Iso_0.0-17
## [117] shiny_1.0.5                 broom_0.4.3
## [119] MESS_0.5.0                  httr_1.3.1
## [121] psych_1.7.8                 colorspace_1.3-2
## [123] XML_3.98-1.9                truncnorm_1.0-7
## [125] splines_3.4.1               RBGL_1.52.0
## [127] multtest_2.32.0             gmm_1.6-1
## [129] jsonlite_1.5                futile.options_1.0.0
## [131] BBmisc_1.11                 R6_2.2.2
## [133] Hmisc_4.0-3                 htmltools_0.3.6
## [135] mime_0.5                    glue_1.1.1
## [137] class_7.3-14                interactiveDisplayBase_1.14.0
## [139] codetools_0.2-15            mvtnorm_1.0-6
## [141] mixtools_1.1.0              curl_3.0
## [143] gtools_3.5.0                openxlsx_4.0.17
## [145] GO.db_3.4.1                 survival_2.41-3
## [147] munsell_0.4.3               e1071_1.6-8
## [149] GenomeInfoDbData_0.99.0     iterators_1.0.9
## [151] impute_1.50.1               reshape2_1.4.2
## [153] gtable_0.2.0
```