

Code for *Selective roles for CBP and p300 as coregulators for androgen-regulated gene expression in advanced prostate cancer cells.*

Dai-Ying Wu

January 23, 2012

1 Preface

In the interests of reproducible research (<http://reproducibleresearch.net>) I have included the code I used to process the data and get the results for this paper.

We ran 24 samples on 2 Illumina HT12v4 microarrays at The Southern California Genotyping Consortium. These samples were processed at the facility with default outlier removal and did not include TIFF images. The resulting data files (idats) were read into Genome Studio and exported without normalization or background correction using the export 'standard probe profile' and export 'control probe profile' feature using the default number of significant digits. Standard error and number of probes were also included in the export (but not used) as were 9 probes with some imputed values (not significant in comparisons of interest).

These two probe files, which can be reconstructed from the 'raw' data on GEO, are the bead summerized datasets that are used for further analysis in R/bioconductor.

2 Read in and Quality Check

Read in bead summerized probes and target file, target file can be extracted from GEO data but I have also included it at the end of this document.

```
> library(limma)
> library(qvalue)
> #x is eset that holds raw values
> #y is eset that holds log2 transformed normalized values
> #z is eset that holds batch corrected values
> x = read.ilmn(files="irina-spp.txt", ctrlfiles="irina-cpp.txt")

Reading file irina-spp.txt ... ..
Reading file irina-cpp.txt ... ..

> targets = read.table("sample description.txt", header=T, row.names=1)
> targets = cbind(targets, Type=paste(targets[,1], targets[,2], sep="_"))
> x$targets = targets = targets[x$targets$SampleNames,]
```

2.1 Raw expression boxplots + MDS clustering

```
> boxplot(log2(x$E[x$genes$Status=="regular",]),range=0,
+ xlab="Arrays",ylab="log2 intensities", main="Regular probes")

> boxplot(log2(x$E[x$genes$Status=="NEGATIVE",]),range=0,
+ xlab="Arrays",ylab="log2 intensities", main="Control probes")
```

Regular probes

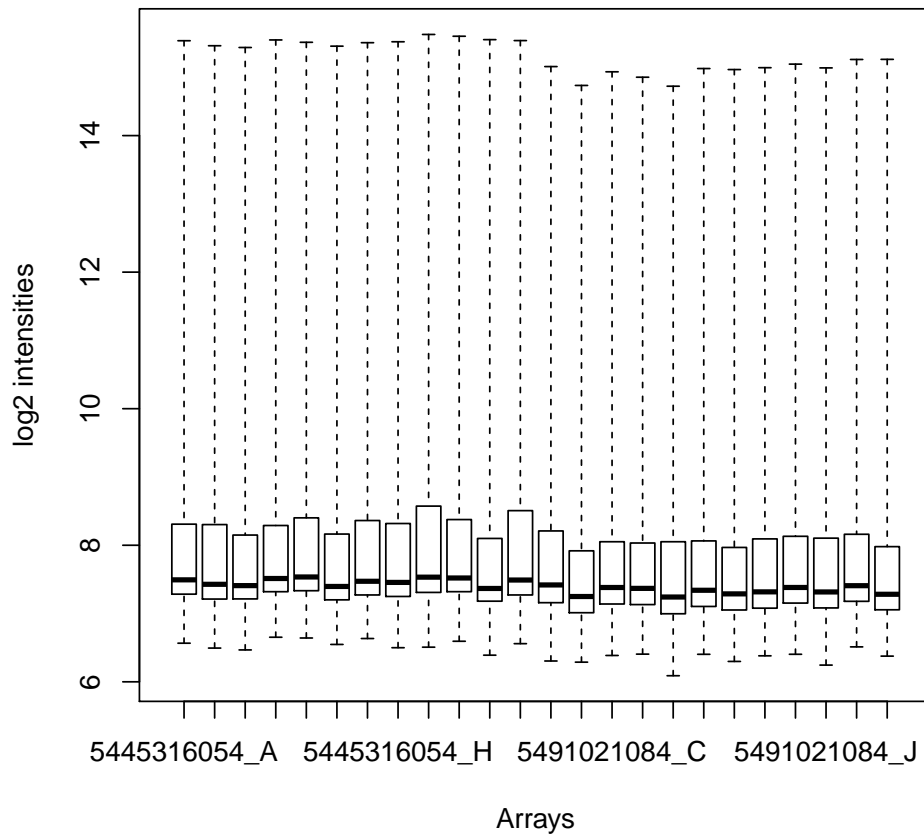


Figure 1: Boxplot of raw expression intensity of regular probes

Control probes

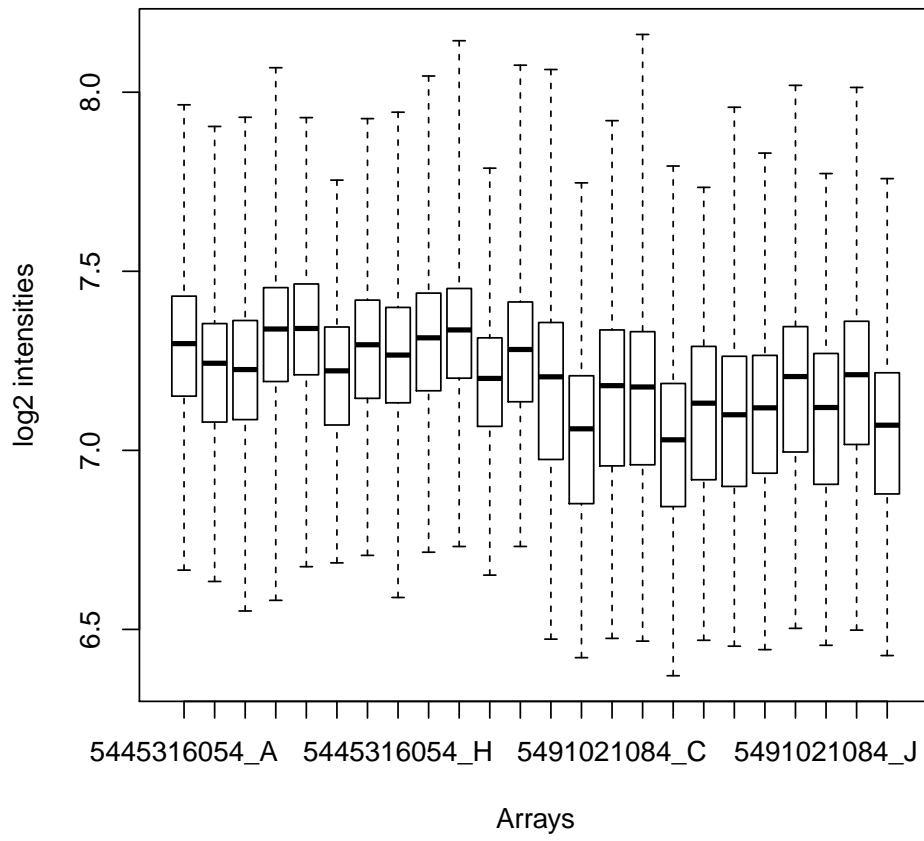


Figure 2: Boxplot of raw expression intensity of control probes

```

> y = neqc(x) #log2 transform + normalize
> plotMDS(y,labels=paste(targets[,1], targets[,2], unclass(targets[,3]), sep="_"),
+ col=unclass(x$targets$Type),xlim = c(-1.5,1.5), ylim=c(-1,1)) #color by type

```

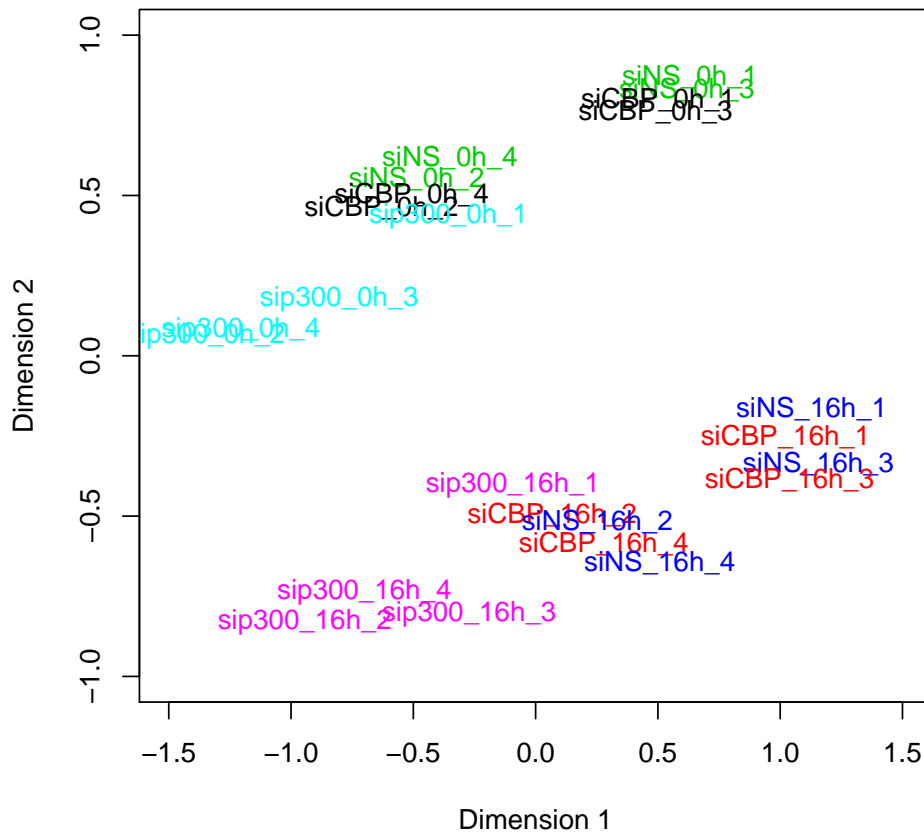


Figure 3: MDS plot of normalized arrays colored by experiment

```

> plotMDS(y,labels=paste(targets[,1], targets[,2], unclass(targets[,3]), sep="_"),
+ col=unclass(x$targets$batch),xlim = c(-1.5,1.5), ylim=c(-1,1)) #color by batch

```

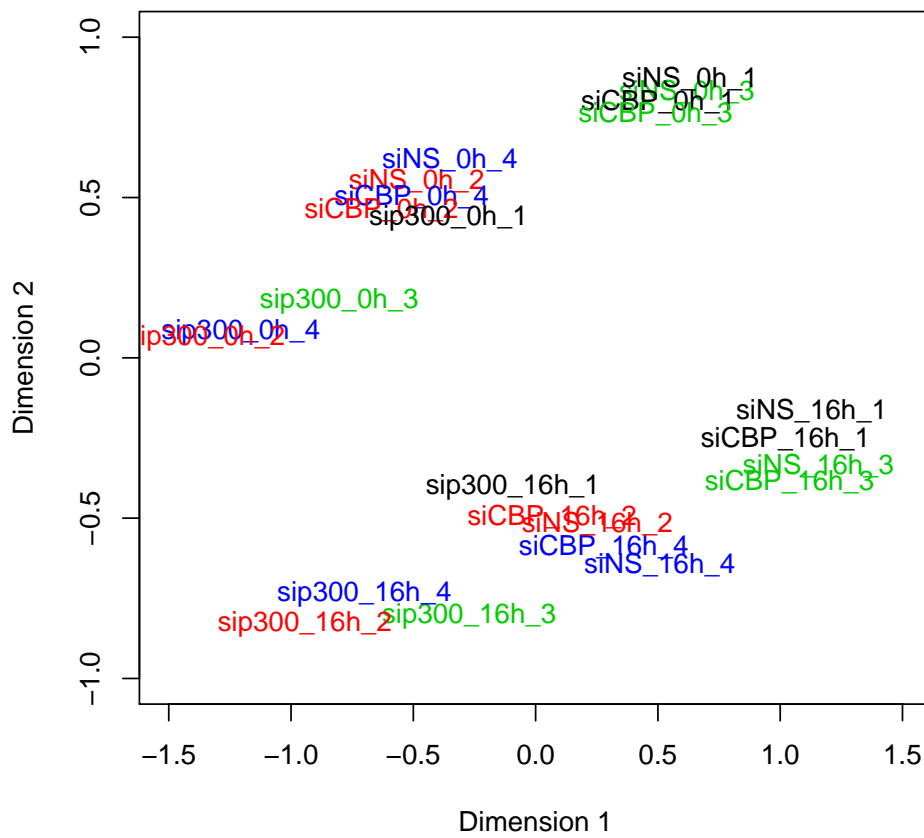


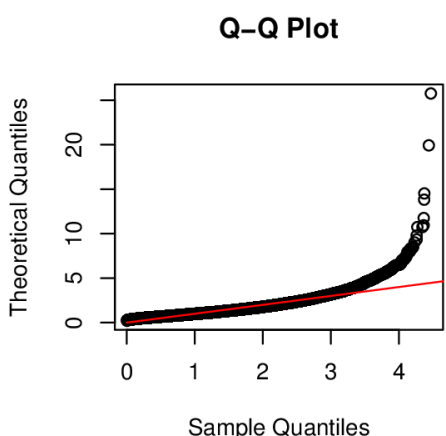
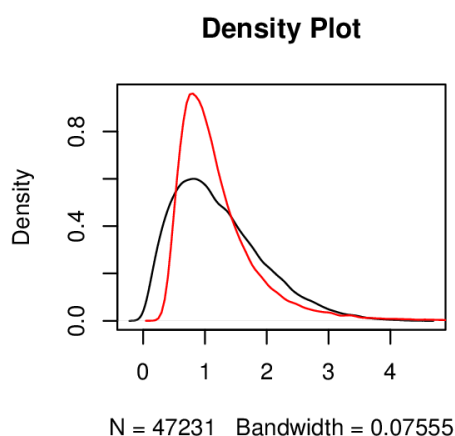
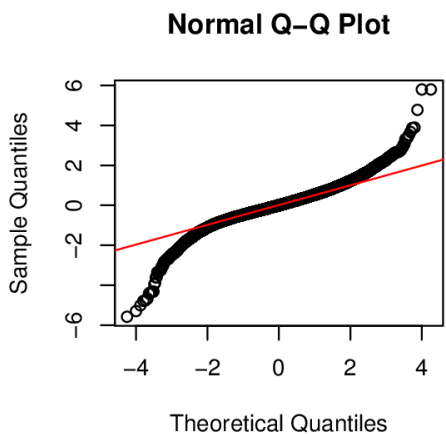
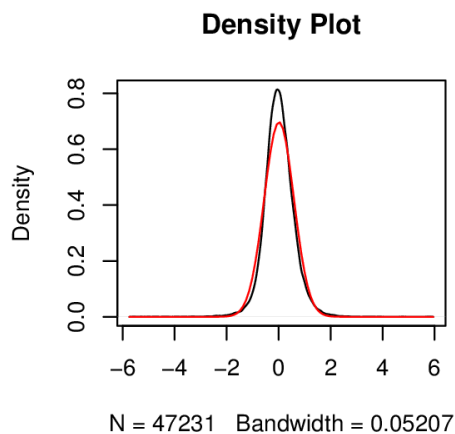
Figure 4: MDS plot of normalized arrays colored by batch

From the above plots, there might be some batch effects that keep the CBP and NS groups together (1+3, 2+4) `Combat.R` is run to remove these effects

```
> source("../ComBat_mod.R") #slightly modified to accept different input
> cb_targets = cbind("Array_Name"=colnames(y$E),
+ "Sample_Name"=as.character(y$targets$Type), "Batch"=unclass(y$targets$batch))
> cb_adj = ComBat_mod(y$E, cb_targets, write=F, skip=1) #takes a while
```

```
Reading Sample Information File
Reading Expression Data File
Found 4 batches
Found 0 covariate(s)
Standardizing Data across genes
Fitting L/S model and finding priors
Finding parametric adjustments
Adjusting the Data
```

```
> colnames(cb_adj) = colnames(y$E) #combat chops up 1st col name
> z = y
> z$E = cb_adj
```



```

> plotMDS(z, labels=paste(targets[,1], targets[,2], unclass(targets[,3]), sep="_"),
+ col=unclass(x$targets$Type), xlim = c(-1.5,1.5), ylim=c(-1,1)) #color by type

```

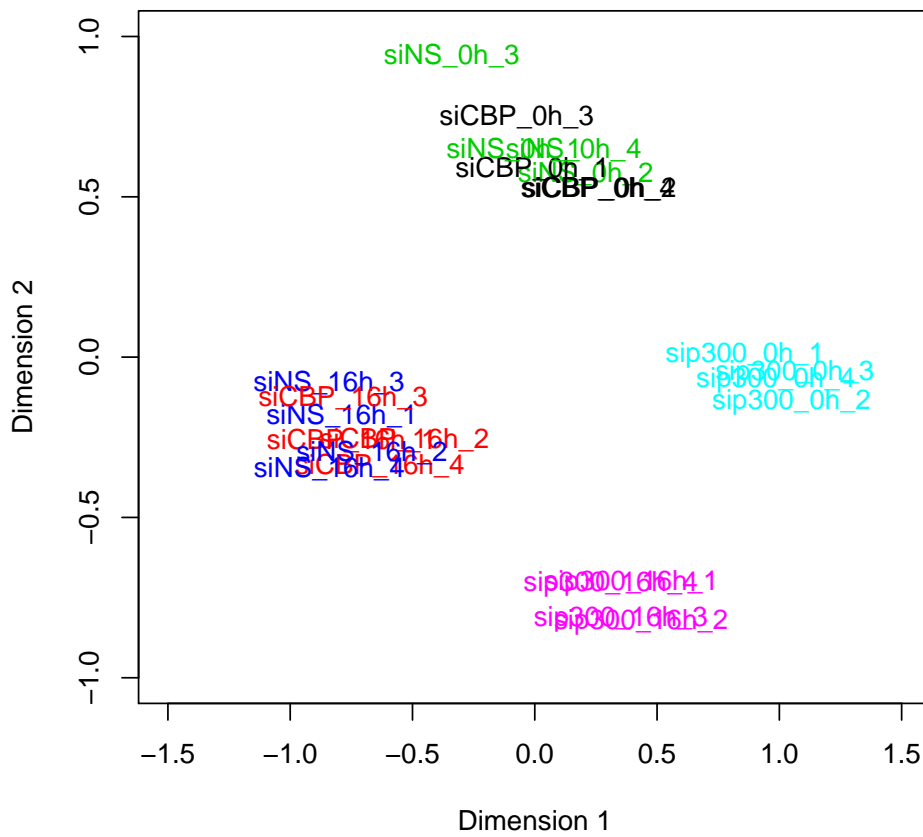


Figure 5: MDS plot of normalized, batch corrected arrays colored by experiment

3 Identify differentially regulated genes

Use eBayes from limma package to find CBP regulated genes, p300 regulated genes and DHT-regulated genes. (see paper for details)

3.1 CBP regulated

```
> sel = z$targets[,1] != "sip300" & z$targets[,2] == "16h" #cbp regulated
> lumisub = z$E[,sel]
> pd = z$targets[colnames(lumisub),] #phenotype data
> des = matrix(0, ncol(lumisub), length(levels(factor(pd$treat))))
> for(i in 1:length(levels(factor(pd$Type)))){
+ des[pd$Type==levels(factor(pd$Type))[i],i]=1
+ }
> colnames(des) = levels(factor(pd$treat))
> des

      siCBP siNS
[1,]      1    0
[2,]      0    1
[3,]      0    1
[4,]      1    0
[5,]      1    0
[6,]      0    1
[7,]      1    0
[8,]      0    1

> cm = rbind(1,-1) #assume col2 is NS
> if(!grepl(colnames(des)[2], "siNS")) { cm = -cm }
> cm

      [,1]
[1,]      1
[2,]     -1

> fit = lmFit(lumisub,des)
> fit2 = contrasts.fit(fit,cm)
> efit = eBayes(fit2)
> cbp_efit = efit
> cbp_efit$qv = qvalue(efit$p.value)$qvalues
> sig_fdr = which(cbp_efit$qv<0.1)
> sig16cbp = rownames(efit[sig_fdr,][order(efit$p.value[sig_fdr]),]) #illumina IDs
> length(sig16cbp) #7

[1] 7

> z$genes[match(sig16cbp, z$genes[,1]), 2]

[1] "CREBBP" "SERPINE2" "MAPK9" "ANXA9" "C19orf4" "SC01" "TMEM20"
```

3.2 p300 regulated

```
> sel = z$targets[,1] != "siCBP" & z$targets[,2] == "16h" #p300 regulated
> lumisub = z$E[,sel]
> pd = z$targets[colnames(lumisub),]
```



```

> des = matrix(0, ncol(lumisub), length(levels(factor(pd$treat))))
> for(i in 1:length(levels(factor(pd$Type)))){
+ des[pd$Type==levels(factor(pd$Type))[i],i]=1
+ }
> colnames(des) = levels(factor(pd$treat))
> des

      siNS sip300
[1,]    1     0
[2,]    0     1
[3,]    1     0
[4,]    0     1
[5,]    0     1
[6,]    1     0
[7,]    0     1
[8,]    1     0

> cm = rbind(1,-1) #assume col2 is NS
> if(!grepl(colnames(des)[2], "siNS")) { cm = -cm }
> cm

      [,1]
[1,]   -1
[2,]    1

> fit = lmFit(lumisub,des)
> fit2 = contrasts.fit(fit,cm)
> efit = eBayes(fit2)
> p300_efit = efit
> p300_efit$qv = qvalue(efit$p.value)$qvalues
> sig_fdr = which(p300_efit$qv<0.05)
> sig16p300 = rownames(efit[sig_fdr,][order(efit$p.value[sig_fdr]),])
> length(sig16p300) #4582

[1] 4582

> head(z$genes[match(sig16p300, z$genes[,1]), 2])

[1] "PCDHB2" "NTNG1" "PHLDA2" "CAB39L" "CAB39L" "CBR3"

```

3.3 DHT regulated

```

> sel = z$targets[,1] == "siNS" #hormone regulated
> lumisub = z$E[,sel]
> pd = z$targets[colnames(lumisub),]
> des = matrix(0, ncol(lumisub), length(levels(factor(pd$hour))))
> for(i in 1:length(levels(factor(pd$Type)))){
+ des[pd$Type==levels(factor(pd$Type))[i],i]=1
+ }
> colnames(des) = levels(factor(pd$hour))
> des

      Oh 16h
[1,]    1    0
[2,]    1    0

```

```

[3,] 0 1
[4,] 0 1
[5,] 0 1
[6,] 1 0
[7,] 0 1
[8,] 1 0

> cm = rbind(1,-1)
> fit = lmFit(lumisub,des)
> fit2 = contrasts.fit(fit,cm)
> efit = eBayes(fit2)
> hr_efit = efit
> hr_efit$qv = qvalue(efit$p.value)$qvalues
> sig_fdr = which(hr_efit$qv<0.05)
> hor_reg = rownames(efit[sig_fdr,][order(efit$p.value[sig_fdr]),])
> length(hor_reg) #676

[1] 676

> head(z$genes[match(hor_reg, z$genes[,1]), 2])

[1] "KLK2" "RHO" "SNAI2" "ACSL3" "MICAL1" "SGK1"

> table(efit[sig_fdr,]$coefficients>0) #up and down regulated genes

FALSE TRUE
 416 260

> table(is.element(hor_reg, sig16p300)) #DHT regulated AND p300 regulated

FALSE TRUE
 357 319

```

4 Output

4.1 GEO spreadsheet

GEO output for Illumina expression excel template

```

> out = matrix(0, ncol=ncol(z$E)*2, nrow=nrow(z$E))
> colnames(out) = as.character(1:(ncol(z$E)*2))
> for(i in 1:ncol(z$E)) {
+   out[, (2*(i-1)+1)] = z$E[,i]
+   out[, (2*(i-1)+2)] = z$other[[1]][,i]
+   colnames(out)[(2*(i-1)+1)] = colnames(z$E)[i]
+   colnames(out)[(2*(i-1)+2)] = "Detection Pval"
+ }
> rownames(out) = rownames(z$E)
> head(out[,1:8])

```

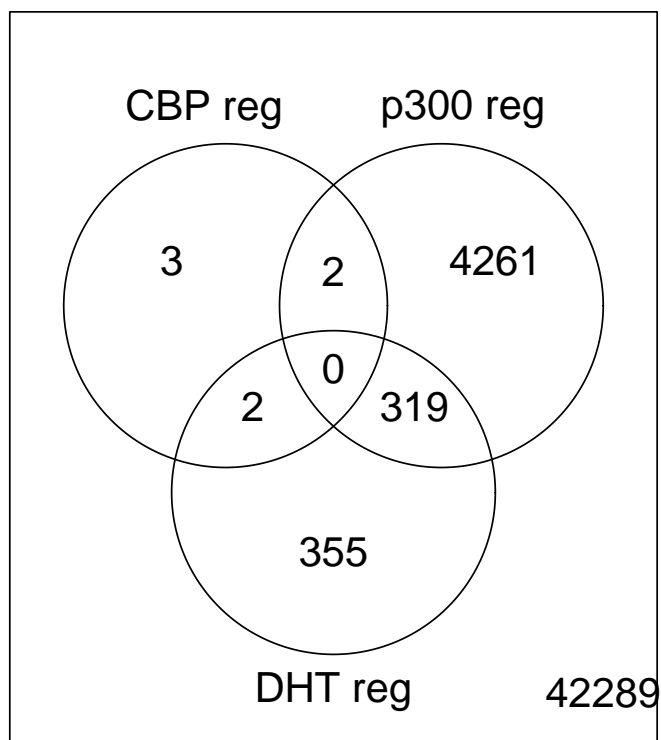
	5445316054_A	Detection	Pval	5445316054_B	Detection	Pval
ILMN_1762337	5.347191	0.2272727		5.430834	0.16753250	
ILMN_2055271	5.349342	0.2259740		5.940997	0.01688312	
ILMN_1736007	5.567351	0.1155844		5.317307	0.26623380	
ILMN_2383229	5.065763	0.5051948		4.978239	0.71818180	
ILMN_1806310	5.178655	0.3779221		5.815979	0.02467532	

ILMN_ID	4.755641	0.8662338	4.763895	0.85584410
ILMN_1779670	4.755641	0.8662338	4.763895	0.85584410
5445316054_C	Detection	Pval	5445316054_D	Detection Pval
ILMN_1762337	5.118083	0.4363636	5.155345	0.46363640
ILMN_2055271	5.478488	0.2298701	5.911265	0.02597403
ILMN_1736007	5.040926	0.5792208	5.431196	0.19220780
ILMN_2383229	5.152320	0.4480520	5.525455	0.12857140
ILMN_1806310	5.095255	0.4532467	5.155710	0.41428570
ILMN_1779670	4.838303	0.8077922	4.778316	0.82727270

```
> #write.table(out, file="GEO_norm.txt", sep="\t", quote=F) #rerun w/x for raw
```

4.2 Venn Diagram

```
> a = vennCounts(cbind(CBPreg=cbp_efit$qv<0.1,
+ p300reg=p300_efit$qv<0.05, hormreg=hr_efit$qv<0.05))
> vennDiagram(a, names=c("CBP reg", "p300 reg", "DHT reg"))
> #figure 1c is based on this, figure in paper is generated using Vennerable library
> # properly weighted venn digram looked terrible due to low number of CBP regulated genes
```



5 Other

5.1 Targets file

```
> read.table("sample description.txt", header=T, row.names=1) #targets file
```

	treatments	hour	batch
5445316054_A	siNS	0h	8.25.10A
5445316054_B	sip300	0h	8.25.10B
5445316054_C	siNS	0h	8.20.10
5445316054_D	siCBP	16h	8.25.10B
5445316054_E	siCBP	0h	8.18.10
5445316054_F	siNS	16h	8.25.10B
5445316054_G	sip300	16h	8.25.10A
5445316054_H	siNS	16h	8.18.10
5445316054_I	sip300	0h	8.20.10
5445316054_J	siCBP	16h	8.18.10
5445316054_K	siCBP	0h	8.25.10A
5445316054_L	sip300	16h	8.18.10
5491021084_A	sip300	0h	8.25.10A
5491021084_B	siCBP	16h	8.20.10
5491021084_C	sip300	16h	8.25.10B
5491021084_D	siNS	16h	8.25.10A
5491021084_E	sip300	16h	8.20.10
5491021084_F	siCBP	0h	8.25.10B
5491021084_G	siCBP	16h	8.25.10A
5491021084_H	siCBP	0h	8.20.10
5491021084_I	siNS	0h	8.18.10
5491021084_J	sip300	0h	8.18.10
5491021084_K	siNS	16h	8.20.10
5491021084_L	siNS	0h	8.25.10B

5.2 R/bioconductor version

```
> sessionInfo()
```

```
R version 2.13.0 (2011-04-13)  
Platform: x86_64-pc-linux-gnu (64-bit)
```

```
locale:
```

```
[1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C  
[3] LC_TIME=en_US.UTF-8      LC_COLLATE=en_US.UTF-8  
[5] LC_MONETARY=C            LC_MESSAGES=en_US.UTF-8  
[7] LC_PAPER=en_US.UTF-8     LC_NAME=C  
[9] LC_ADDRESS=C            LC_TELEPHONE=C  
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
```

```
attached base packages:
```

```
[1] stats      graphics  grDevices  utils      datasets  methods   base
```

```
other attached packages:
```

```
[1] qvalue_1.26.0 limma_3.8.2
```

```
loaded via a namespace (and not attached):
```

```
[1] tcltk_2.13.0
```

5.3 Modifications to Combat.R

Combat.R can be found here: <http://jlab.byu.edu/ComBat/Download.html>
Header comments were removed before the diff command. These modifications were made so I did not have to write out and read in a file everytime I wanted to rerun Combat.R with different parameters. Function arguments were changed to allow for passing in matrix of expression values.

```
$ diff ComBat.R ComBat_mod.R
1c1,2
< ComBat <- function(expression_xls, sample_info_file, type='txt', write=T, covariates='all', par.prior=T, filter=F, skip=0, prior.plots=T){
---
> ComBat_mod <- function(expression_xls, sample_info_file, type='txt', write=T,
> covariates='all', par.prior=T, filter=F, skip=0, prior.plots=T){
4c5,6
< saminfo <- read.table(sample_info_file, header=T, sep='\t', comment.char='')
---
> saminfo = sample_info_file #alternate loading, not fully done yet
> #saminfo <- read.table(sample_info_file, header=T, sep='\t', comment.char='')
8,9c10,11
< if(type=='csv'){
< dat <- read.csv(expression_xls,header=T,as.is=T)
---
> #if(type=='csv'){
> # dat <- read.csv(expression_xls,header=T,as.is=T)
13c15
< colnames(dat)=scan(expression_xls,what='character',nlines=1,sep=',',quiet=T)[1:ncol(dat)]
---
> # colnames(dat)=scan(expression_xls,what='character',nlines=1,sep=',',quiet=T)[1:ncol(dat)]
15,20c17,23
< }
<     else{
< dat <- read.table(expression_xls,header=T,comment.char='',fill=T,sep='\t', as.is=T)
< dat <- dat[,trim.dat(dat)]
< colnames(dat)=scan(expression_xls,what='character',nlines=1,sep='\t',quiet=T)[1:ncol(dat)]
< }
---
> # }
> #else{
> # dat <- read.table(expression_xls,header=T,comment.char='',fill=T,sep='\t', as.is=T)
> # dat <- dat[,trim.dat(dat)]
> # colnames(dat)=scan(expression_xls,what='character',nlines=1,sep='\t',quiet=T)[1:ncol(dat)]
> # }
> dat = expression_xls
23,24c26,27
<     dat <- dat[,-c(1:skip)]
<     else{geneinfo=NULL}
---
>     dat <- dat[,-c(1:skip)]
> }else{geneinfo=NULL}
137c140
< output_file <- paste('Adjusted',expression_xls,'.xls',sep='_')
---
> output_file <- paste('Adjusted expression.xls',sep='_')
```