Project 3: Concordance of Microarray and RNA-Seq Differential Gene Expression

Evie Wan, Salam Al-Abdullatif, Eetu Eklund, Mary T. Yohannes

4/10/2020

Introduction

Before the introduction of RNA sequencing, the standard for genome expression profiling was the use of microarray platforms. Thousands of studies have been based on the use of microarrays to analyze gene expression, but new technologies bring the promise of improvement. Focusing on evaluating the performance of microarray platforms, the FDA initiated the Microarray Quality Control (MAQC) consortium in 2006 to test the reliability of microarray technologies [4]. With the initial attempts to compare microarrays and RNA-seq showing mixed results, the third part of the MAQC consortium, also known as the Sequencing Quality Control (SEQC) project, was launched to determine the reliability of RNA-seq data[6]. To comprehensively conclude the advantages of using RNA-seq, Tong et. al conducted a study that was designed to compare the differentially expressed genes (DEGs) found through microarray analysis and RNA-seq approaches. They found that the concordance between sequencing and microarray analysis was positively correlated, with some indication that RNA-seq performs better in identifying differentially expressed genes. In addition to comparing DEG lists, the study was also designed to generate comparisons of modes of action (MOA) and the chemicals and vehicles used to administer the treatment. The aim of this project is to reproduce the results of the comparison performed in the Tong et al. paper. Selecting a subset of samples to analyze, we align rat short reads to the reference genome, perform differential analysis on RNA-seq data as well as microarray expression data, and finally we compare the results from our analysis to the paper.

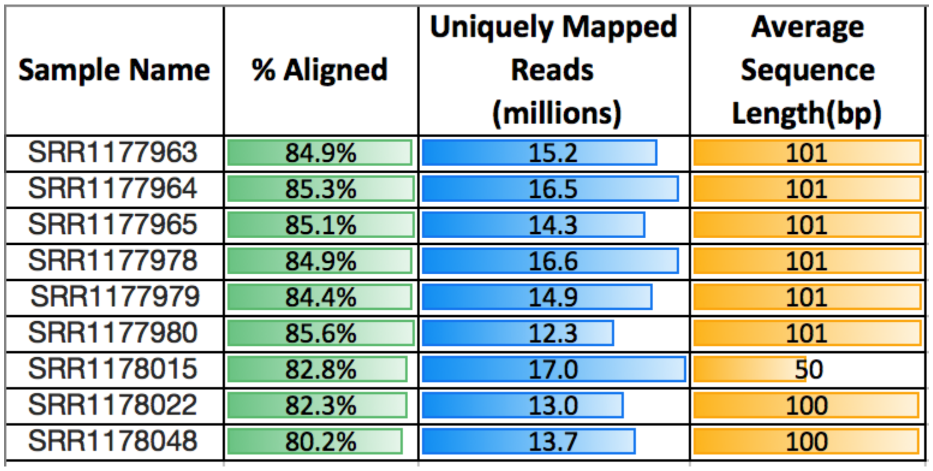
Data & Method

We chose toxgroup 5 which had 9 samples of three different mechanisms of action (MOA): DNA damage, orphan nuclear hormone receptors (CAR/PXR), and peroxisome proliferator-activated receptor alpha (PPARA). AFLATOXIN_B1 (DNA damage), MICONAZOLE (CAR/PXR), and PIRINIXIC_ACID (PPARA) were the three chemical agents considered in our analysis. These caused toxicity via their respective MOAs.

There are 6 pre-processed control samples, which will not be further processed in Data curation. First, FastQC was run on each sample to assess the quality of raw sequence data. 9 html files were generated. The samples were then aligned against the rat reference genome using STAR. The alignment script was written as a qsub script. Other than specifying input files and output file prefix, '--outSAMtype BAM SortedByCoordinate' was used to sort the output in the .bam files, '--outFilterType BySJout' was used to reduce the number of spurious junctions, '--alignSJDBoverhangMin' and '--alignSJDBoverhangMin 1' were used to specify minimum overhang(stretch of unpaired nucleotides) of unannotated and annotated junctions. Multiqc was used to document QC assessments from both STAR and FASTQC. Visualizations of alignment and data quality statistics are reported below.

The second column in green in Table 1 shows the percentage of uniquely mapped reads ranging from ~80% to ~85%. A good quality sample generally will have at least 75% of the reads uniquely mapped. The other two columns in Table 1 show the number of uniquely mapped reads (blue) and the average sequence length (yellow). It should be noted that sample SRR1178015 has average sequence length that is only half of that of the other samples. However, since the number of uniquely mapped reads is high and sample, this sample was not removed from our analysis. Figure 1 and Figure 2 show similar information in a different form of visualization, reinforcing the fact that sample quality is good. In Figure 1, we can see that the number of uniquely mapped reads (blue) far exceeds the number of unmapped reads (red). Similarly, in Figure 2, we can see that in all samples the majority of reads are mapped reads.

Table 1. Percentage and number of uniquely mapped reads and average sequence length of samples. Chosen samples have good quality based on the percentage of uniquely mapped reads



Sample Name	% Aligned	Uniquely Mapped Reads (millions)	Average Sequence Length(bp)
SRR1177963	84.9%	15.2	101
SRR1177964	85.3%	16.5	101
SRR1177965	85.1%	14.3	101
SRR1177978	84.9%	16.6	101
SRR1177979	84.4%	14.9	101
SRR1177980	85.6%	12.3	101
SRR1178015	82.8%	17.0	50
SRR1178022	82.3%	13.0	100
SRR1178048	80.2%	13.7	100

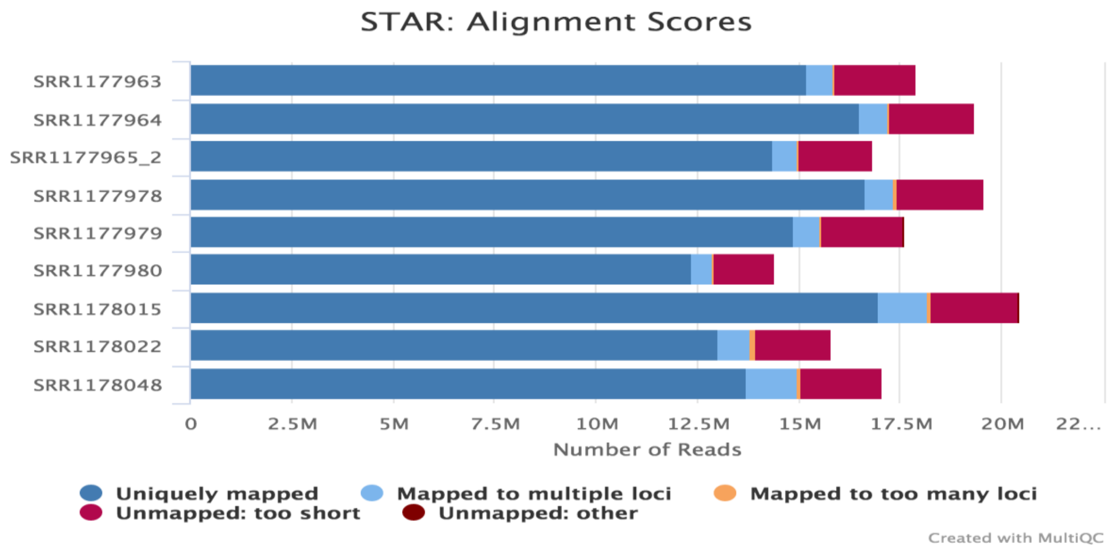


Figure 1. STAR alignment results showing the number of uniquely mapped reads, number of reads mapped to multiple loci, and number of unmapped reads in chosen samples

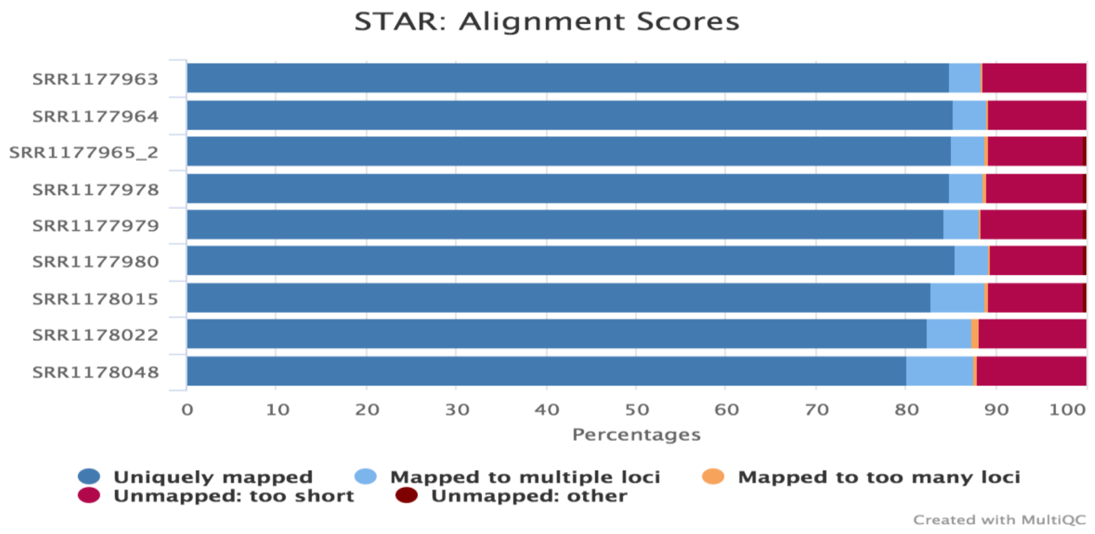


Figure 2. Percentage of uniquely mapped reads and unmapped reads in chosen samples

Another metric that has indication in data quality is duplicated reads. Figure 3 shows that in most samples duplicated reads are more than 50% of total reads. High rates of read duplicates in RNA-seq libraries could indicate over-amplification in PCR or poor library complexity caused by low sample input. It could also be caused by high abundance of a small number of genes such as ribosomal or mitochondrial housekeeping genes. For this reason, the common practice is usually to not remove duplicated reads.

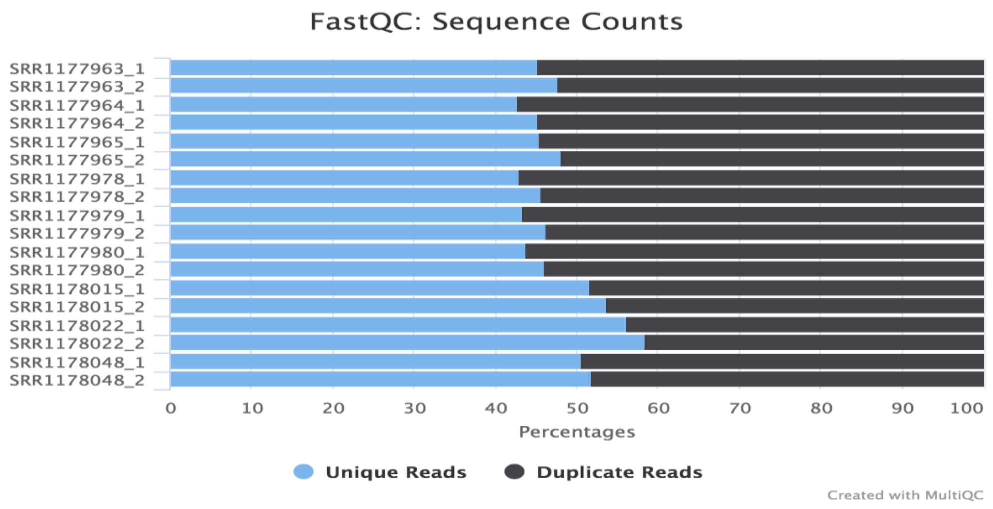


Figure 3. Percentage of unique and duplicated reads in all chosen samples

Another indication of library quality is per base sequence content. In a random library we can expect to see little to no difference between different bases. Therefore, each of the four lines representing different bases should run horizontally across the graph. Figure 4 shows that other than fluctuations at the very beginning, the base contents are very stable. This indicates that there are no differences between libraries in complexity or amplification, in other words, no biases in the data leading to different percent base content. Only the per base content of one sample is shown here as an example. All other samples have similar trends.

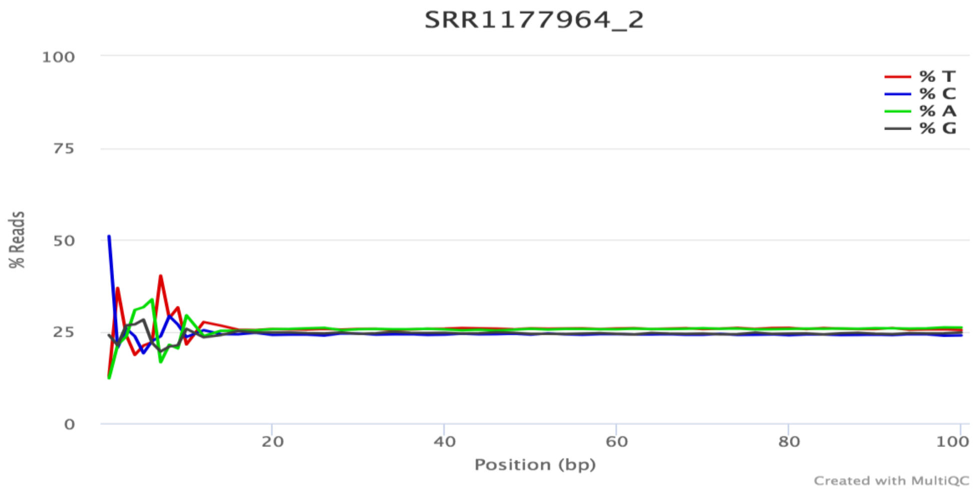


Figure 4. Per Base Sequence Content in Sample SRR1177964_2

Phred scores are logarithmically linked to error probabilities. Figure 5 shows that the mean Phred scores for the majority of the sequence is satisfactory (in green or yellow region). It is normal to see decreasing sequence quality and this would not reduce confidence in data quality as long as the majority of sequence positions have scores in the green region. Similarly, Figure 6 shows that the vast majority of the reads have high quality scores in the green region for all samples.

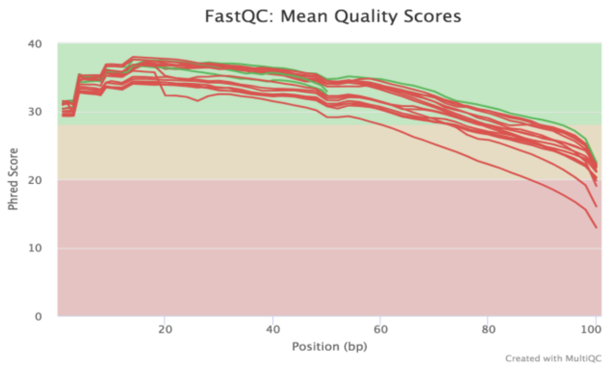


Figure 5. Mean Phread Score and the corresponding position

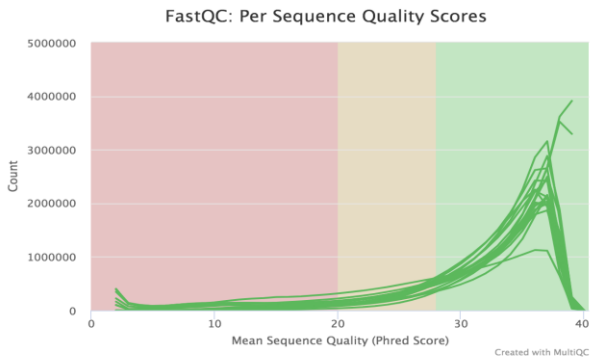


Figure 6. Mean Phred Score per sequence

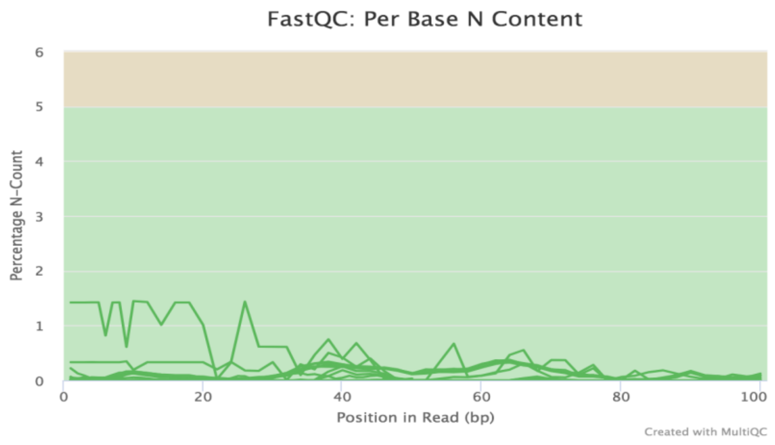


Figure 7. Percentage of non-identified base at each position

An N is substituted when a sequencer is unable to make a base call with sufficient confidence. Figure 7 shows that at all positions the percentage of N-content is lower than 2%. At most positions N-content is lower than 1%. These Multiqc visualizations gave us confidence in the quality of chosen samples. Although in order to more comprehensively address high duplication rate, we will need to look into the duplicated reads or use statistical methods such as the Poisson model to estimate library complexity.

The STAR alignment files were used to create read counts using the featureCounts program from the Subread package v1.6.2. The program requires the use of an annotation file, then each of the nine samples are run through featurecounts to produce individual counts files. Afterwards, multiqc is run on the counts files directory, and the output is examined for interesting features. The counts for each sample are combined into a single data frame for further analysis. The counts for each sample are shown on a boxplot in Figure 8 below. The average counts number across all genes were found to be similar between samples, with a mean of 1115 counts per gene on average across the nine samples. The multiqc report shows that the total number of reads varied between samples, as shown in Figure 9, but the percentage of assigned reads to unassigned reads remained relatively constant across samples.

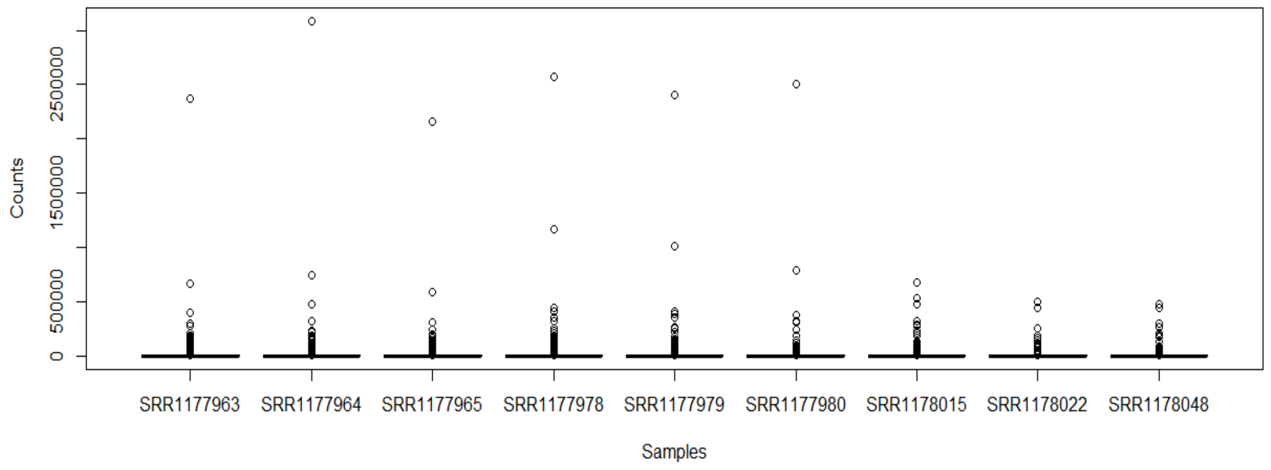


Figure 8. The number of read counts across genes in each sample

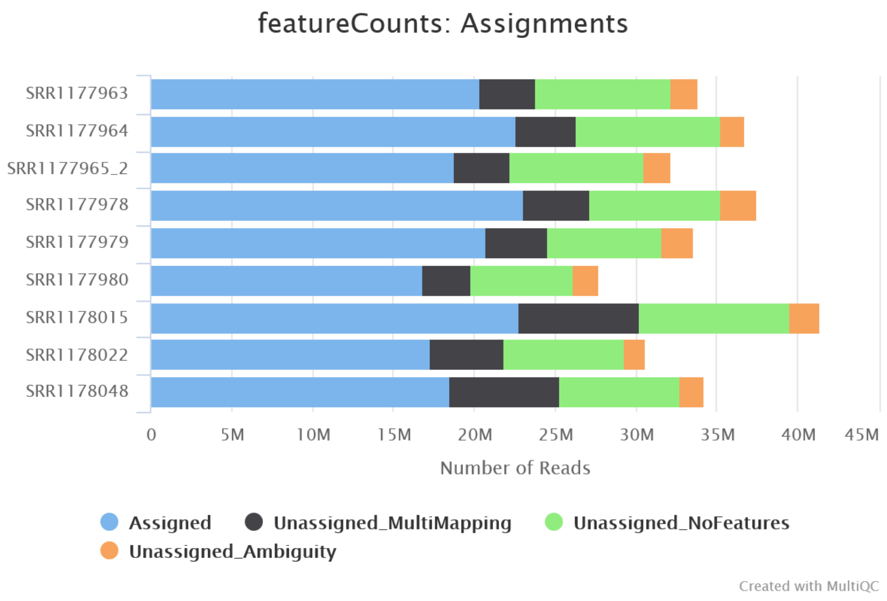


Figure 9. Multiqc report shows varying read counts between samples with a roughly consistent percentage of assigned to total reads

Next, the counts matrix was combined with the read counts from a control dataset for differential expression analysis. DEGs were found using the DESeq2 package v1.26.0 on R. Each sample was annotated for its MOA and chemical vehicle. Three modes of action are present in our samples: DNA damage, CAR/PXR, and PPARA. Each MOA is represented by 3 samples, and 3 controls are matched to each MOA through finding shared vehicle annotations. The PPARA samples correspond to use of Pirinixic Acid as the chemical, the DNA Damage samples correspond to usage of Aflatoxin B1, and the CAR/PXR samples correspond to Miconazole. We create a separate counts matrix for each of the three MOAs, composed of 6 columns with 3 samples and 3 controls. DESeq2 is run on the counts matrices for each MOA to produce a normalized counts matrix as well as differential expression statistics. The differentially expressed genes were filtered by p-adjusted values < 0.05 and reported in increasing value.

The differential expression analysis of the RNA-Seq data with DESeq2 resulted in three differential expression results and three normalized counts matrices for each treatment. Base R functions[5] such as subset(), merge(), sapply() and intersect() function from the dplyr package[3] were utilized to manipulate the DESeq results. Significant differentially expressed genes from each treatment were obtained by establishing a False Discovery Rate (FDR) cutoff of less than 0.05 and an absolute value log2FoldChange of greater than 1. A gene set enrichment analysis was then performed using Database for Annotation, Visualization and Integrated Discovery (DAVID) [1,2] and the results were compared to the pathway enrichment results reported in the Tong et al. paper Supplementary Table 4.

To look for segregation by MOA, a heatmap-based hierarchical clustering of the RNA-Seq samples was created. Each normalized expression matrix had six columns: three sets for treatment and three sets for control. The same sets of controls were compared to DNA damage and PPARA treatments. Thus, for the heatmap matrix, an average normalized count value, which was obtained from the two separate treatment matrices, was used for that control set. The heatmap matrix was constructed by merging all three normalized count matrices on gene-level and contained 5 columns: three sets for each treatment and 6 controls. Since we were interested in looking at the effect of each treatment, their clustering, and the differentially expressed genes that were significant in any of the groups, we took the union of differentially expressed genes that were significant in each treatment, removed duplicates, and subsetted the complete heatmap matrix with it. This not only accounted for genes that were differentially expressed in multiple treatments, but also for the ones that were significant in one of the groups and not the others. The final subsetted heatmap matrix was plotted using the heatmap() function in R.

Results & Discussion

Differentially expressed genes from RNA-seq were determined using DESeq2 analysis. For each MOA, the algorithm produced a normalized counts file as well as a list of differentially expressed genes with their Log2 fold change, p-value, p-adjusted value, and other relevant statistics. The DEGs were filtered at p-adjusted values < 0.05, and the number of significant genes for analysis are reported as follows: 6305 genes for CAR/PXR, 3164 genes for PPARA, and 1262 genes for DNA Damage. These results are summarized in Table 2. The results of the differential expression analysis are shown in Figures 10 and 11. Additionally, the top 10 genes for each mode of action are summarized in Table 3 below.

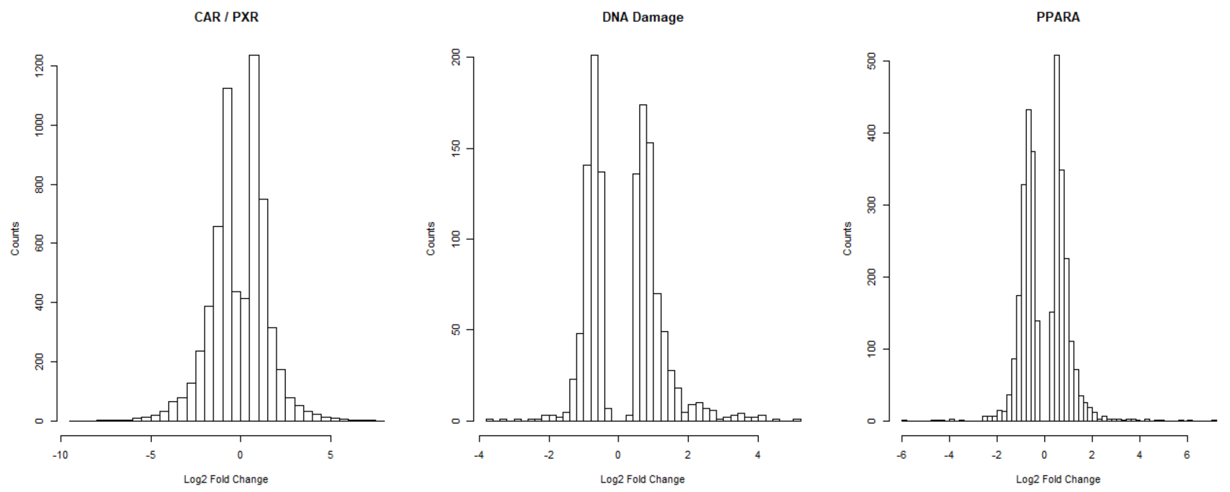


Figure 10. Counts vs Log 2 Fold Change values for each mode of action as determined by DESeq2 analysis

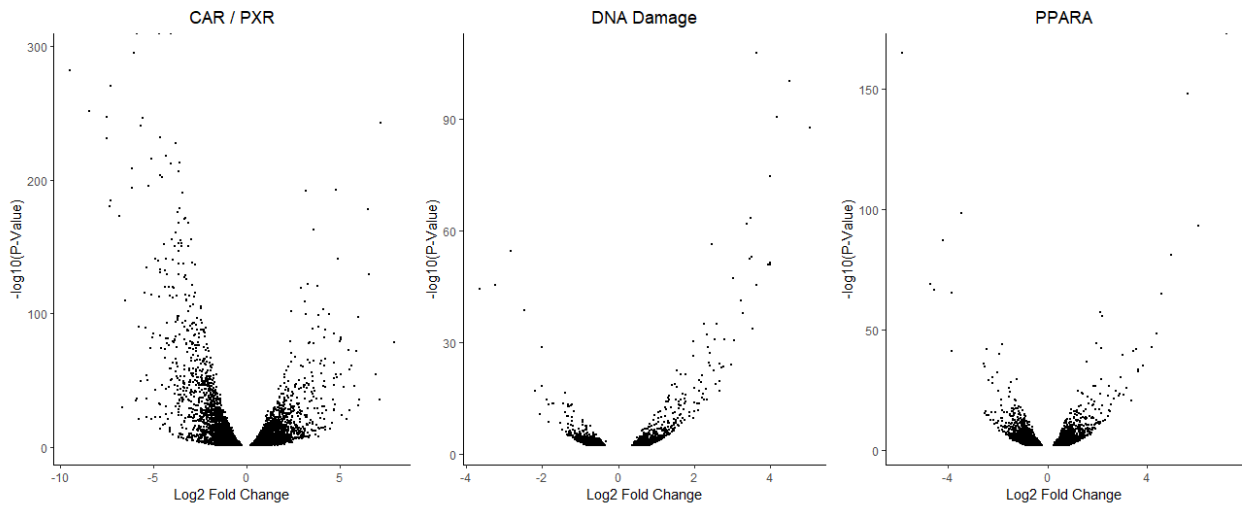


Figure 11. Log2 Fold Change values vs - log of nominal p-value for each treatment mode of action. CAR/PXR shows the largest number of differentially expressed genes through visual inspection.

Table 2. Number of DEG at p.adjust < 0.05 from RNA-seq analysis

CAR / PXR	PPARA	DNA DAMAGE
6305	3164	1262

Table 3: Top 10 DEGs from each treatment using DESeq2

CAR / PXR	PPARA	DNA DAMAGE
NM_017047	NM_024162	NM_020540
NM_001012122	NM_012737	NM_001106632
NM_001025720	NM_019157	NM_021835
NM_131903	NM_001013975	NM_012623
NM_001257095	NM_001109180	NM_017259
NM_001105754	NM_131903	NM_053769
NM_019292	NM_001025639	NM_001003403
NM_182474	NM_017158	NM_199386
NM_031732	NM_001013098	NM_031348
NM_201423	NM_001014063	NM_001106062

The differentially expressed genes from microarray were determined using limma. Each MOA was compared to controls in a liver RMA normalized matrix. This produced a list probes from microarray that included their log fold change, average expression, t statistic, p-value, adjusted p-value, and beta. After filtering for DEGs at adjusted p-value < 0.05 the number of significant probes for analysis were: CAR / PXR - 8890, PPARA - 7985, DNA DAMAGE - 2191 (Table 4). Results from limma analysis are portrayed in Figures 12 and 13. Table of top 10 DEGs is represented in Table 5.

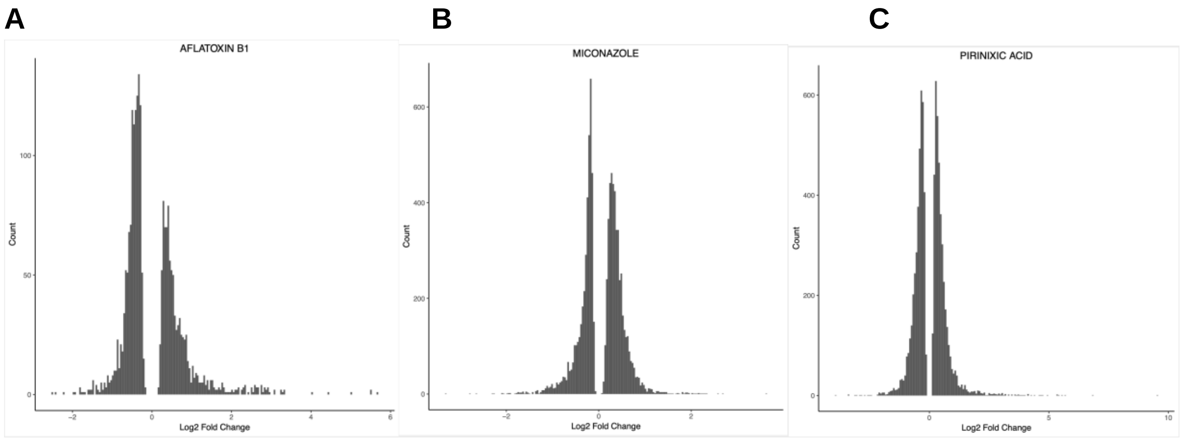


Figure 12. Counts of the Log2 Fold Change values of the significantly DE genes for the three MOA as determined by limma

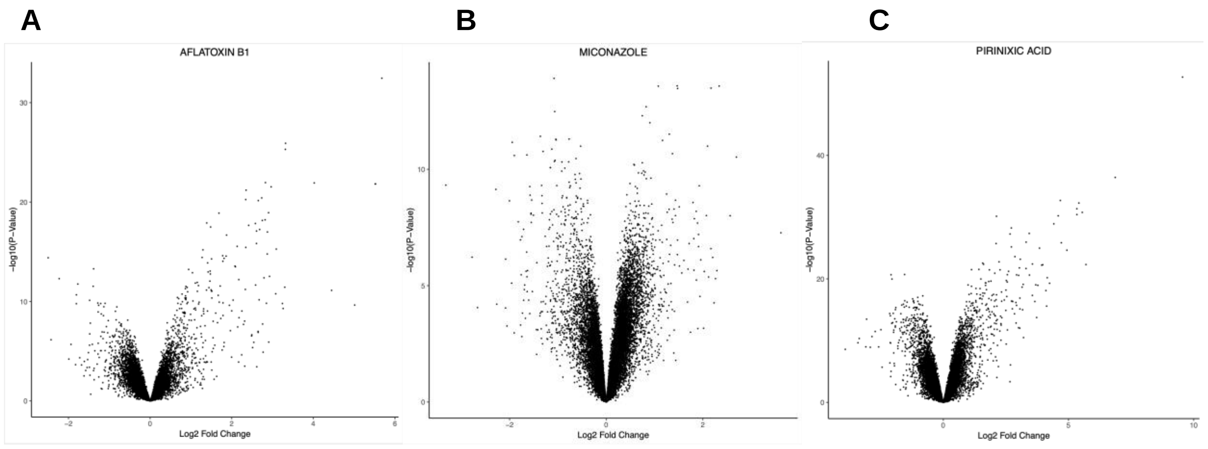


Figure 13. Log2 Fold Change vs -log of nominal p-value for each of the three treatments. Results come from limma analysis. Miconazole treatment results in a uniform volcano plot, while Aflatoxin B1 and Pirinixic Acid seem to be skewed to the right towards positive fold change.

Table 4. Number of DEGs at p.adjust < 0.05 from microarray analysis

CAR / PXR	PPARA	DNA DAMAGE
8890	7985	2191

Table 5. Top 10 DEGs from each treatment using microarray

CAR / PXR	PPARA	DNA DAMAGE
1389113_at	1398250_at	1392754_at
1370698_at	1388211_s_at	1384449_at
1368905_at	1390358_at	1374245_at
1387118_at	1367742_at	1371036_at
1371076_at	1377867_at	1390317_at
1391570_at	1370491_a_at	1370583_s_at
1370613_s_a	1380536_at	1385132_at
1369012_at	1375845_at	1367764_at
1392267_at	1367680_at	1388325_at
1387759_s_at	1384474_a	1388932_at

Concordance was calculated as the overlap of DEGs between-platform (RNA-seq and microarray regression estimates) against the number of DEGs identified by RNA-seq. The files were adjusted for p-values < 0.05. This was then adjusted to remove the contribution of random chance by formula given in Tong et. al. Concordance for DEGs above median and below median were also calculated. Figures 14 and 15 summarize the concordance results for each MOA.

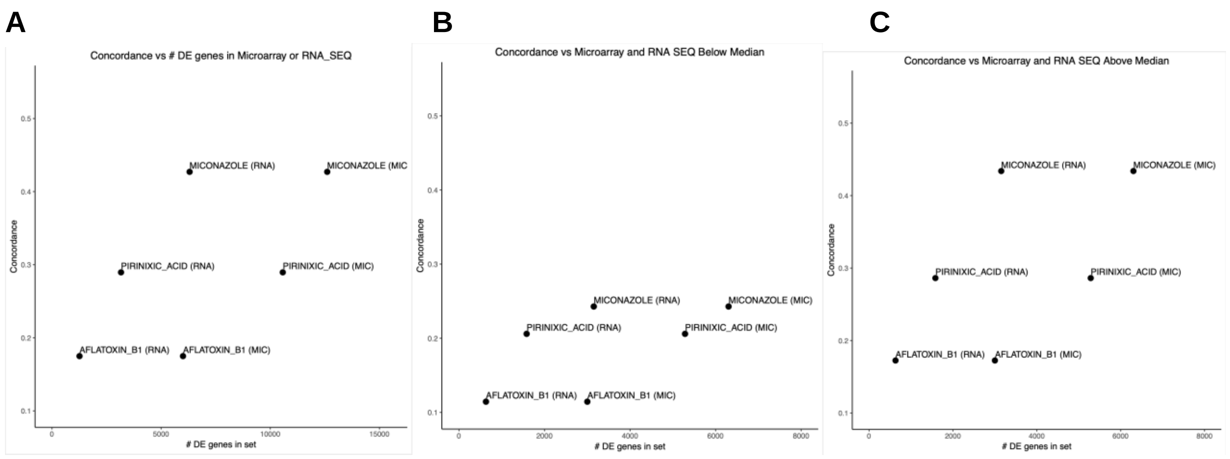


Figure 14. Concordance vs # of DE genes in set for each of the three treatments. A includes all DE genes, B includes all below median DE genes, C includes all Above median DE genes. Miconazole seems to have the highest concordance overall. Concordance at below median seems to be the lowest overall.

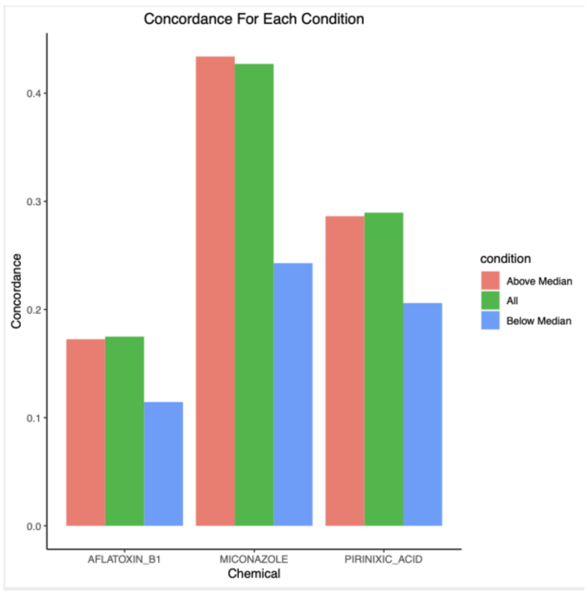


Figure 15. Concordance for each condition shown at all DE genes, above DE median genes, and below DE median genes. Concordance between microarray and RNA-Seq DE genes seems to be highest using all DE genes or Above Median DE genes.

Concordance for Miconazole (CAR/PXR) was the highest overall, while concordance for Aflatoxin (DNA Damage) was lowest overall for each analysis. Concordance when using all genes and when filtered down to above median DEG expression were the highest for each MOA. When filtered down to below median expression the concordance at each MOA was the lowest. This could be explained by the fact that RNA-seq is better at detecting low expression levels than microarrays. Miconazole (CAR/PXR) seems to have the lowest rate of lower differential gene expression (Figure 12,13) which explains Miconazole's highest concordance levels. Concordance seems to be overall a little lower for our analysis than in Tong et. al, which could be caused by not filtering for 8 probesets mapping unambiguously. Concordance seems to also have a positive correlation with the # DE genes in set as shown in Figure 14.

Table 6: Enriched pathways and scores identified using DE genes from each treatment analyses and compared to results reported in the Tong et al. paper

DNA damage - AFLATOXIN_B1		
Enriched Pathways	Enrichment Score:	Pathway in Tong et al. paper
Cellular response	3.1	-
Xenobiotic metabolism	2.74	Xenobiotic metabolism signaling
Transcription regulation	2.06	Cell cycle: G2/M DNA damage checkpoint regulation
Receptor activity	1.74	-
Transport	1.7	-
CAR/PXR - MICONAZOLE		
Enriched Pathways	Enrichment Score:	Pathways in Tong et al. paper
Cellular component: Microsome	11.35	-
Signaling	10.69	Aryl hydrocarbon receptor signaling
Metabolism of xenobiotics	8.17	Xenobiotic metabolism signaling
Blood coagulation	7.72	-
Pyridoxal phosphate	6.47	-
PPARA - PIRINIXIC_ACID		
Enriched Pathways	Enrichment Score:	Pathways in Tong et al. paper
fatty acid beta-oxidation	10.2	Fatty acid β-oxidation I, Fatty acid β-oxidation III
Peroxisome	5.77	-
Lipid metabolism	4.79	Fatty acid α-oxidation, Fatty acid β-oxidation I, Fatty acid β-oxidation III
Signaling	4.32	Aryl hydrocarbon receptor signaling
NAD binding	4.07	Oxidative ethanol degradation III

After establishing an FDR and log2FoldChange cutoff, there were 310 differentially expressed genes for DNA damage, 3093 for CAR/PXR, and 655 for PPARA. These genes were used for DAVID analysis of each treatment group. For CAR/PXR, it had the most number of differentially expressed genes from the DESeq2 analysis and since DAVID has a limit of 3000 genes, the top 1000 up-regulated and 1000 down-regulated genes (2000 intotal) were inputted in DAVID to determine enriched pathways. As seen in Table 6 above, some of the enriched pathways that are identified by our DE genes from each analysis are in agreement with the processes from the Tong et al. paper. From our top five functional clusters for the PPARA MOA chemical group, four of the enriched pathways were among the pathways enriched in the paper. This might have been because the number of enriched pathways for PPARA presented in the paper was a lot more than the other two chemical groups. Xenobiotic metabolism is a common enriched pathway for DNA damage and CAR/PXR while aryl hydrocarbon receptor signaling was enriched for both

CAR/PXR and PPARA. The paper examined 27 chemicals and five MOAs while our analysis was based on three chemicals from three MOAs. This could explain the difference that is seen between the pathways that were enriched in our analysis and the analysis in the paper.

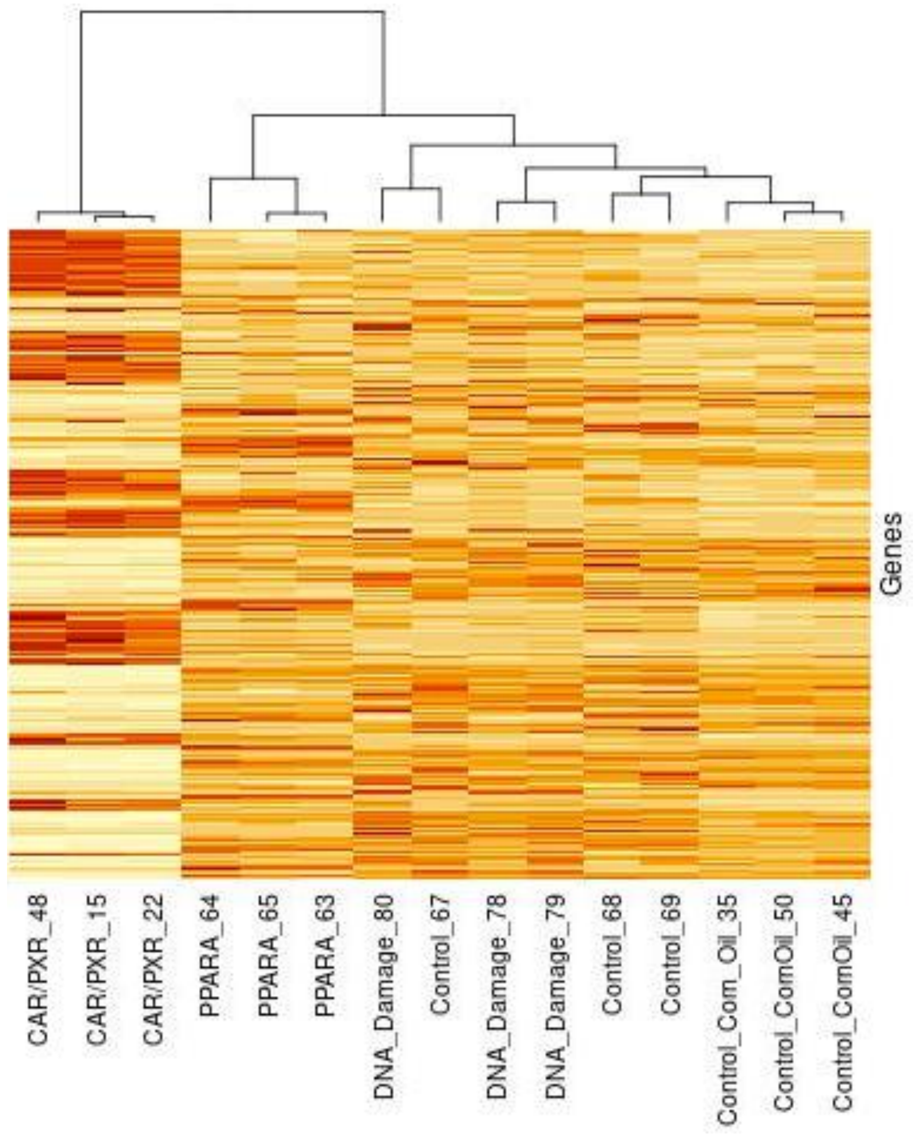


Figure 16. Heatmap-based hierarchical clustering of 15 RNA-Seq samples: DNA damage (3), CAR/PXR (3), PPARA (3) & controls (6), based on DESeq normalized counts. Samples from each MOA clustered together and CAR/PXR had the most significant differentially expressed genes.

According to the clustered heatmap, the samples for each MOA are clustered together except for one of the DNA damage and control samples that cluster together. This is because the control group is shared between PPARA and DNA damage and values for the normalized count were averaged between the two treatment tables. If we took out the controls and just plotted the MOAs (heatmap can be found at the end of the paper within the R code), we can see that all samples from each MOA group cluster together. Furthermore, samples from CAR/PXR seem to have way more significant genes as compared to the other samples. This is correct as it also had the most number of differentially expressed genes from the DESeq2 analysis of the RNA-Seq data.

Conclusion

We compare the results of our differential expression analysis to the paper's conclusions and find several common results. The DESeq2 results from RNA-seq show a similar number of differentially expressed genes in each mode of action, with the exception of the DEGs found from the CAR/PXR samples, which were even higher than we expected. One potential reason for this difference is the filtering thresholds imposed on the dataset after differential analysis. Our RNA-seq dataset is filtered by adjusted p-values < 0.05, but Tong et al. does not discuss their filtering process in much depth, so we can predict that their filtering process leaves a shorter list of DEGs. Looking at our RNA-seq results to the microarray results, there is a general trend across each MOA that appears to remain consistent by visual comparison in both scatter plots of Log2 Fold Change vs - Log p-value as well as histograms of counts vs Log2 Fold Change. Concordance is the highest when the set is limited to above median gene expression and is lower for genes with lower expression because RNA-seq is better at detecting low expression than microarray. Though the pathway enrichment results of our DE genes were not entirely the same as those reported in the paper, there was still an agreement among the ones that we found to have been represented in our DAVID analysis and the paper's results. The heatmap provided additional visualization of the clustering among MOAs and confirmed that samples from each MOA cluster together.

References

1- Huang DW, Sherman BT, Lempicki RA. Systematic and integrative analysis of large gene lists using DAVID Bioinformatics Resources. Nature Protoc. 2009;4(1):44-57.

2- Huang DW, Sherman BT, Lempicki RA. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. Nucleic Acids Res. 2009;37(1):1-13.

3- Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2020). dplyr: A Grammar of Data Manipulation. R package version 0.8.4. https://CRAN.R-project.org/package=dplyr

4- Making the most of microarrays. *Nat Biotechnol* **24,** 1039 (2006). https://doi.org/10.1038/nbt0906-1039

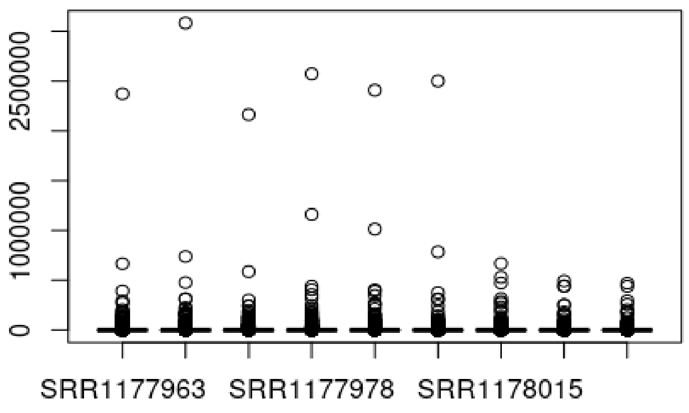
5- R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

6- Wang, Charles et al. “The concordance between RNA-seq and microarray data depends on chemical treatment and transcript abundance.” *Nature biotechnology* vol. 32,9 (2014): 926-32. doi:10.1038/nbt.3001

Project R-code

### Programmer
Read counts

```
SRR1177963 <- read.csv("../salam/SRR1177963.csv")
SRR1177964 <- read.csv("../salam/SRR1177964.csv")
SRR1177965 <- read.csv("../salam/SRR1177965.csv")
SRR1177978 <- read.csv("../salam/SRR1177978.csv")
SRR1177979 <- read.csv("../salam/SRR1177979.csv")
SRR1177980 <- read.csv("../salam/SRR1177980.csv")
SRR1178015 <- read.csv("../salam/SRR1178015.csv")
SRR1178022 <- read.csv("../salam/SRR1178022.csv")
SRR1178048 <- read.csv("../salam/SRR1178048.csv")
genes <- SRR1178048[, 1]
combined <- data.frame(gene_id = genes, SRR1177963[, 7], SRR1177964[, 7], SRR11
77965[, 7],
    SRR1177978[, 7], SRR1177979[, 7], SRR1177980[, 7], SRR1178015[, 7], SRR1178
022[, 7], SRR1178048[, 7])
colnames(combined) <- c("gene_id", "SRR1177963", "SRR1177964", "SRR1177965",
    "SRR1177978", "SRR1177979", "SRR1177980", "SRR1178015", "SRR1178022", "SRR1
178048")
write.csv(combined, "counts.csv")
boxplot(combined[, 2:10], breaks = 50)
```



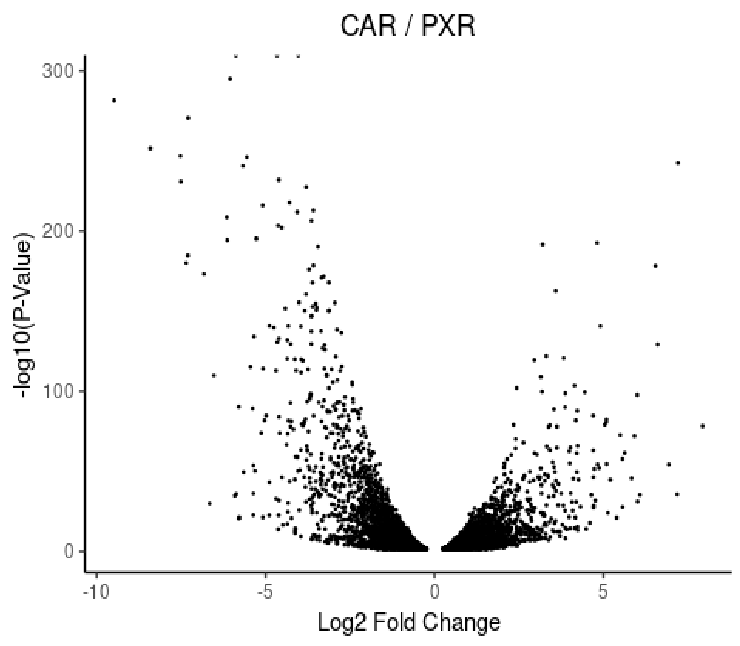
```
control_counts <- read.csv("../salam/control_counts.csv")
metadata <- read.csv("../salam/toxgroup_5_rna_info.csv")
rownames(combined) <- combined[, 1]
combined <- combined[, 2:10]
rownames(control_counts) <- control_counts[, 1]
control_counts <- control_counts[, which(colnames(control_counts) %in% metadata
$Run)]
counts <- cbind(combined, control_counts)
```

```r
write.csv(combined, "../salam/counts.csv")
write.csv(counts, "../salam/counts_control_combined.csv")
```

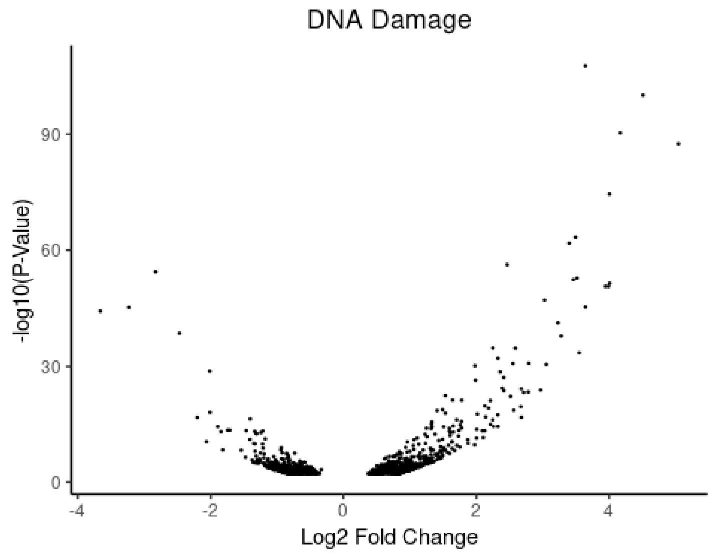
DESeq2 differential expression of RNA-Seq

```r
# load counts
cnts <- read.csv('../salam/counts_control_combined.csv',row.names=1)
# sample information
info <- read.csv('../salam/toxgroup_5_rna_info.csv')

info <- info[which(info$mode_of_action %in% c("CAR/PXR", "Control")),]
info <- info[which(info$vehicle == info[1,]$vehicle),]
cnts <- cnts[, which(colnames(cnts) %in% (info$Run))]

# filter out rows that have any zeros for funzies
cnts <- subset(cnts,rowSums(cnts==0)==0)

# create the DESeq object
dds <- DESeqDataSetFromMatrix(
  countData = cnts,
  colData = info,
  design= ~ mode_of_action
)

## factor levels were dropped which had no samples

##   it appears that the last variable in the design formula, 'mode_of_action',
##   has a factor level, 'Control', which is not the reference level. we recomm
end
##   to use factor(...,levels=...) or relevel() to set this as the reference le
vel
##   before proceeding. for more information, please see the 'Note on factor le
vels'
##   in vignette('DESeq2').

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe charac
ters
##   for column names in R. [This is a message, not an warning or error]

# relevel mode_of_action as factor
dds$mode_of_action <- relevel(dds$mode_of_action, ref='Control')

# run DESeq
dds <- DESeq(dds)

## estimating size factors
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe charac
```

```
ters
##   for column names in R. [This is a message, not an warning or error]

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe charac
ters
##   for column names in R. [This is a message, not an warning or error]

## final dispersion estimates

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe charac
ters
##   for column names in R. [This is a message, not an warning or error]

## fitting model and testing

res <- results(dds, contrast=c("mode_of_action", "CAR/PXR", "Control"))
res <- lfcShrink(dds, coef=2)

## using 'normal' for LFC shrinkage, the Normal prior from Love et al (2014).
##
## Note that type='apeglm' and type='ashr' have shown to have less bias than ty
pe='normal'.
## See ?lfcShrink for more details on shrinkage type, and the DESeq2 vignette.
## Reference: https://doi.org/10.1093/bioinformatics/bty895

res <- res[which(res$padj < 0.05),]
res <- res[order(res$padj),]

# write out DE results
write.csv(res,'../salam/deseq_CAR_PXR_results.csv')

# write out matrix of normalized counts
write.csv(counts(dds,normalized=TRUE),'../salam/deseq_CAR_PXR_norm_counts.csv')

res_car <- res
df_car <- data.frame(logFC = res_car$log2FoldChange, pval = res_car$pvalue)
ggplot(df_car, aes(x=logFC, y=-log10(pval))) + geom_point(size=.25) +
  labs(title="CAR / PXR",x="Log2 Fold Change", y = "-log10(P-Value)") + theme_c
lassic() +
  theme(plot.title = element_text(hjust = 0.5))
```

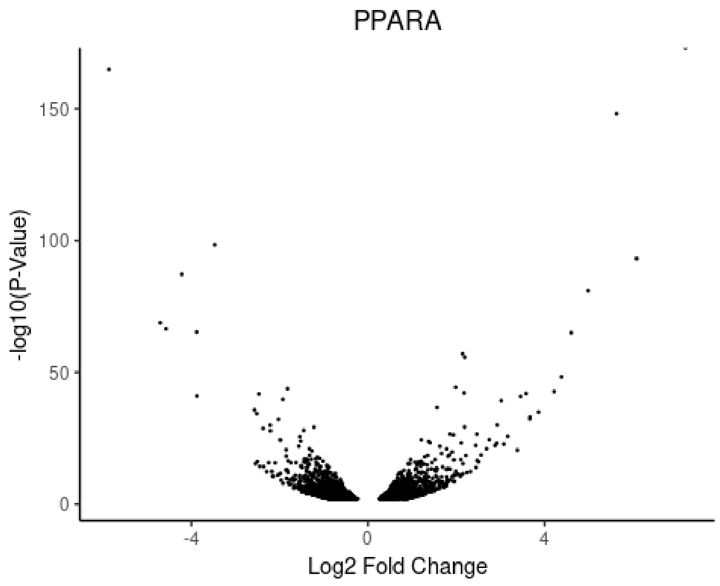


```
cnts <- read.csv('../salam/counts_control_combined.csv',row.names=1)
# sample information
info <- read.csv('../salam/toxgroup_5_rna_info.csv')

info <- info[which(info$mode_of_action %in% c("DNA_Damage", "Control")),]
info <- info[which(info$vehicle == info[1,]$vehicle),]
cnts <- cnts[, which(colnames(cnts) %in% (info$Run))]

# filter out rows that have any zeros for funzies
cnts <- subset(cnts,rowSums(cnts==0)==0)

# create the DESeq object
dds <- DESeqDataSetFromMatrix(
  countData = cnts,
  colData = info,
  design= ~ mode_of_action
)

## factor levels were dropped which had no samples

# relevel mode_of_action as factor
dds$mode_of_action <- relevel(dds$mode_of_action, ref='Control')

# run DESeq
dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates
## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

res <- results(dds, contrast=c("mode_of_action", "DNA_Damage", "Control"))
res <- lfcShrink(dds, coef=2)

## using 'normal' for LFC shrinkage, the Normal prior from Love et al (2014).
##
## Note that type='apeglm' and type='ashr' have shown to have less bias than ty
pe='normal'.
## See ?lfcShrink for more details on shrinkage type, and the DESeq2 vignette.
## Reference: https://doi.org/10.1093/bioinformatics/bty895

res <- res[which(res$padj < 0.05),]
res <- res[order(res$padj),]

# write out DE results
write.csv(res,'../salam/deseq_DNA_Damage_results.csv')

# write out matrix of normalized counts
write.csv(counts(dds,normalized=TRUE),'../salam/deseq_DNA_Damage_norm_counts.cs
v')

res_dna <- res
df_dna <- data.frame(logFC = res_dna$log2FoldChange, pval = res_dna$pvalue)

ggplot(df_dna, aes(x=logFC, y=-log10(pval))) + geom_point(size=.25) +
    labs(title="DNA Damage",x="Log2 Fold Change", y = "-log10(P-Value)") + them
e_classic() +
    theme(plot.title = element_text(hjust = 0.5))
```



```
# load counts
cnts <- read.csv('../salam/counts_control_combined.csv',row.names=1)
# sample information
info <- read.csv('../salam/toxgroup_5_rna_info.csv')

info <- info[which(info$mode_of_action %in% c("PPARA", "Control")),]
info <- info[which(info$vehicle == info[1,]$vehicle),]
cnts <- cnts[, which(colnames(cnts) %in% (info$Run))]

# filter out rows that have any zeros for funzies
cnts <- subset(cnts,rowSums(cnts==0)==0)

# create the DESeq object
dds <- DESeqDataSetFromMatrix(
  countData = cnts,
  colData = info,
  design= ~ mode_of_action
)

## factor levels were dropped which had no samples

# relevel mode_of_action as factor
dds$mode_of_action <- relevel(dds$mode_of_action, ref='Control')

# run DESeq
dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

res <- results(dds, contrast=c("mode_of_action", "PPARA", "Control"))
res <- lfcShrink(dds, coef=2)

## using 'normal' for LFC shrinkage, the Normal prior from Love et al (2014).
##
## Note that type='apeglm' and type='ashr' have shown to have less bias than ty
pe='normal'.
## See ?lfcShrink for more details on shrinkage type, and the DESeq2 vignette.
## Reference: https://doi.org/10.1093/bioinformatics/bty895

res <- res[which(res$padj < 0.05),]
res <- res[order(res$padj),]

# write out DE results
```

```
write.csv(res,'../salam/deseq_PPARA_results.csv')

# write out matrix of normalized counts
write.csv(counts(dds,normalized=TRUE),'../salam/deseq_PPARA_norm_counts.csv')

res_ppara <- res
df_ppara <- data.frame(logFC = res_ppara$log2FoldChange, pval = res_ppara$pvalu
e)

ggplot(df_ppara, aes(x=logFC, y=-log10(pval))) + geom_point(size=.25) +
    labs(title="PPARA",x="Log2 Fold Change", y = "-log10(P-Value)") + theme_cla
ssic() +
    theme(plot.title = element_text(hjust = 0.5))
```



### Analyst
limma Aflatoxin

```
# sample info dataframe with array_id and chemical columns
#samples <- read.csv('groups/group_5_mic_info.csv',as.is=TRUE)
samples <- read.csv("/project/bf528/project_3/groups/group_5_mic_info.csv",as.i
s=TRUE)
samples <- samples[which(samples$chemical %in% c("AFLATOXIN_B1", "Control")),]

# the full RMA normalized matrix of all experiments
rma <- read.table('/projectnb/bf528/project_3/samples/liver-normalization-rma.t
xt',
                  sep='\t',
                  as.is=TRUE,
                  header=TRUE,
                  row.names=1)

# subset the full expression matrix to just those in this comparison
```

```
rma.subset <- rma[paste0('X',samples$array_id)]

### CHANGE design to be for each treatment instead of all of them. this means w
e do the analysis a total of 4 times.

# construct a design matrix modeling treatment vs control for use by limma
design <- model.matrix(
  ~factor(
    samples$chemical,
    levels=c('Control','AFLATOXIN_B1')#,'MICONAZOLE', 'PIRINIXIC_ACID')
  )
)
colnames(design) <- c('Intercept','AFLATOXIN_B1')#,'MICONAZOLE', 'PIRINIXIC_ACI
D')

# run limma
fit <- lmFit(rma.subset, design)
fit <- eBayes(fit)
t <- topTable(fit, coef=2, n=nrow(rma.subset), adjust='BH')

# write out the results to file
t2<- t[order(t$adj.P.Val),]
write.csv(t2,'limma_results_sorted_AFLATOXIN_B1.csv')
```

limma MICONAZOLE

```
samples <- read.csv('/project/bf528/project_3/groups/group_5_mic_info.csv',as.i
s=TRUE)
samples <- samples[which(samples$chemical %in% c("MICONAZOLE", "Control")),]

# the full RMA normalized matrix of all experiments
rma <- read.table('/projectnb/bf528/project_3/samples/liver-normalization-rma.t
xt',
                  sep='\t',
                  as.is=TRUE,
                  header=TRUE,
                  row.names=1)

# subset the full expression matrix to just those in this comparison
rma.subset <- rma[paste0('X',samples$array_id)]

# construct a design matrix modeling treatment vs control for use by limma
design <- model.matrix(
  ~factor(
    samples$chemical,
    levels=c('Control','MICONAZOLE')#,'MICONAZOLE', 'PIRINIXIC_ACID')
  )
)
colnames(design) <- c('Intercept','MICONAZOLE')#,'MICONAZOLE', 'PIRINIXIC_ACID'
)
```

```
# run limma
fit <- lmFit(rma.subset, design)
fit <- eBayes(fit)
t <- topTable(fit, coef=2, n=nrow(rma.subset), adjust='BH')

# write out the results to file
t2<- t[order(t$adj.P.Val),]
write.csv(t2,'limma_results_MICONAZOLE.csv')
```

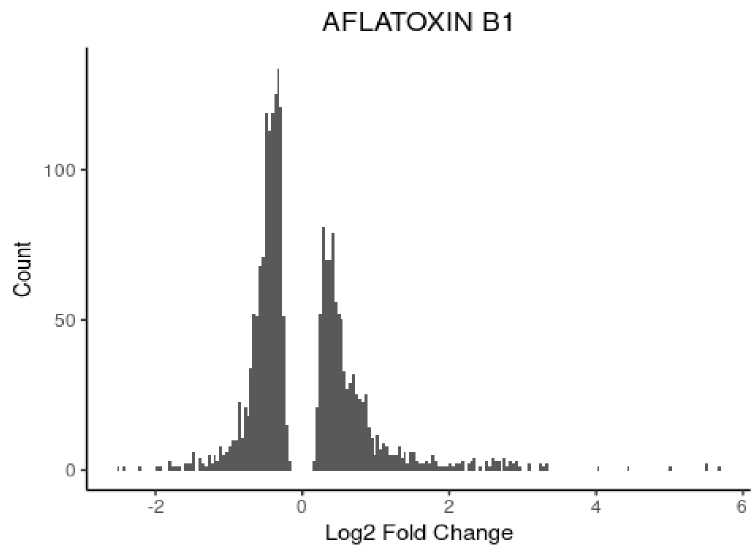
limma Pirinixic Acid

```
samples <- read.csv('/project/bf528/project_3/groups/group_5_mic_info.csv',as.i
s=TRUE)
samples <- samples[which(samples$chemical %in% c("PIRINIXIC_ACID", "Control")),
]

# the full RMA normalized matrix of all experiments
rma <- read.table('/projectnb/bf528/project_3/samples/liver-normalization-rma.t
xt',
                  sep='\t',
                  as.is=TRUE,
                  header=TRUE,
                  row.names=1)

# subset the full expression matrix to just those in this comparison
rma.subset <- rma[paste0('X',samples$array_id)]

# construct a design matrix modeling treatment vs control for use by limma
design <- model.matrix(
  ~factor(
    samples$chemical,
    levels=c('Control','PIRINIXIC_ACID')#,'MICONAZOLE', 'PIRINIXIC_ACID')
  )
)
colnames(design) <- c('Intercept','PIRINIXIC_ACID')#,'MICONAZOLE', 'PIRINIXIC_A
CID')

# run limma
fit <- lmFit(rma.subset, design)
fit <- eBayes(fit)
t <- topTable(fit, coef=2, n=nrow(rma.subset), adjust='BH')

# write out the results to file
t2<- t[order(t$adj.P.Val),]
write.csv(t2,'limma_results_PIRINIXIC_ACID.csv')
```

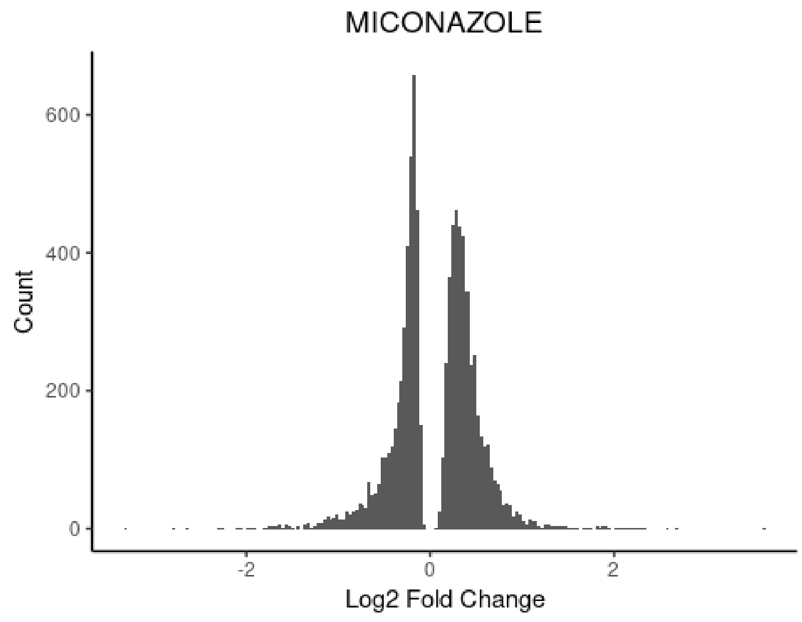
Limma graphs

```
A <- read.csv('limma_results_sorted_AFLATOXIN_B1.csv', header=TRUE)
M <- read.csv('limma_results_MICONAZOLE.csv', header=TRUE)
```

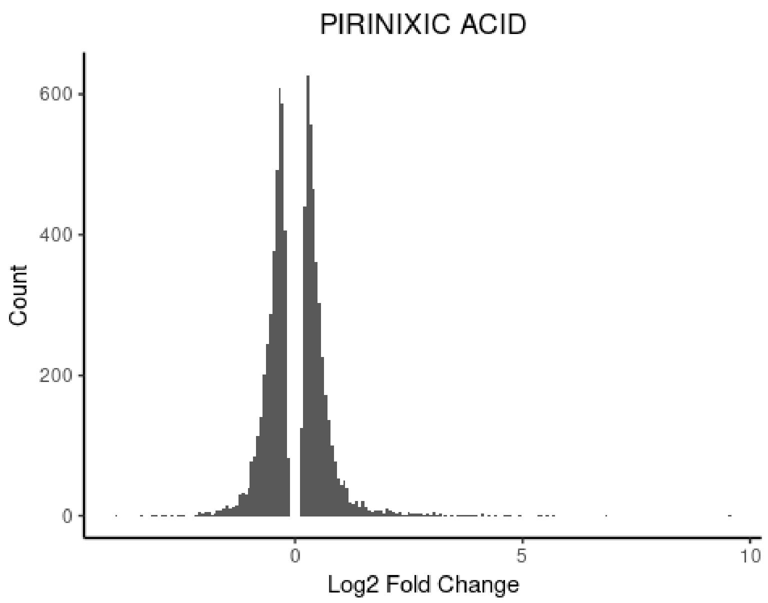
```r
P <- read.csv('limma_results_PIRINIXIC_ACID.csv', header=TRUE)

# creating histograms
ggplot(A[which(A$adj.P.Val<0.05),], aes(x=logFC)) + geom_histogram(bins = 200)
+
labs(title="AFLATOXIN B1",x="Log2 Fold Change", y = "Count") + theme_classic()
+
theme(plot.title = element_text(hjust = 0.5))
```



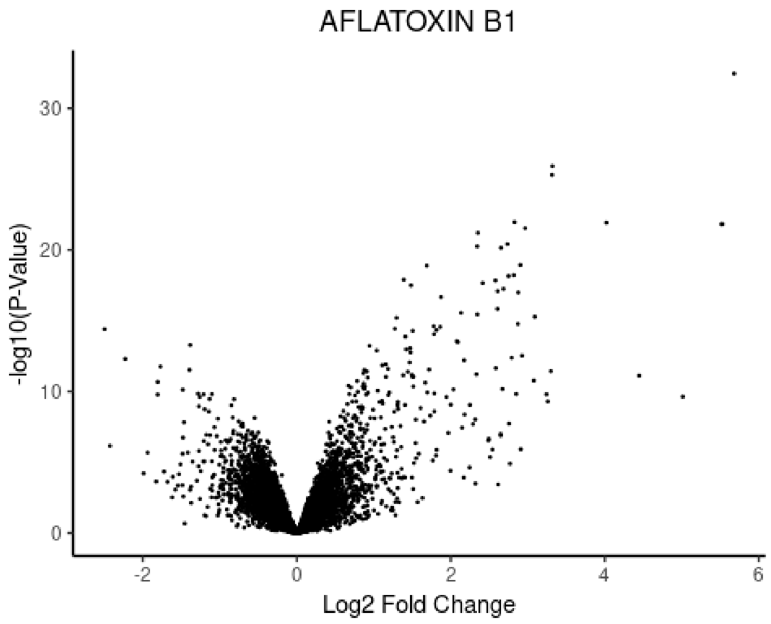
```r
ggplot(M[which(M$adj.P.Val<0.05),], aes(x=logFC)) + geom_histogram(bins = 200)
+
labs(title="MICONAZOLE",x="Log2 Fold Change", y = "Count") + theme_classic() +
theme(plot.title = element_text(hjust = 0.5))
```



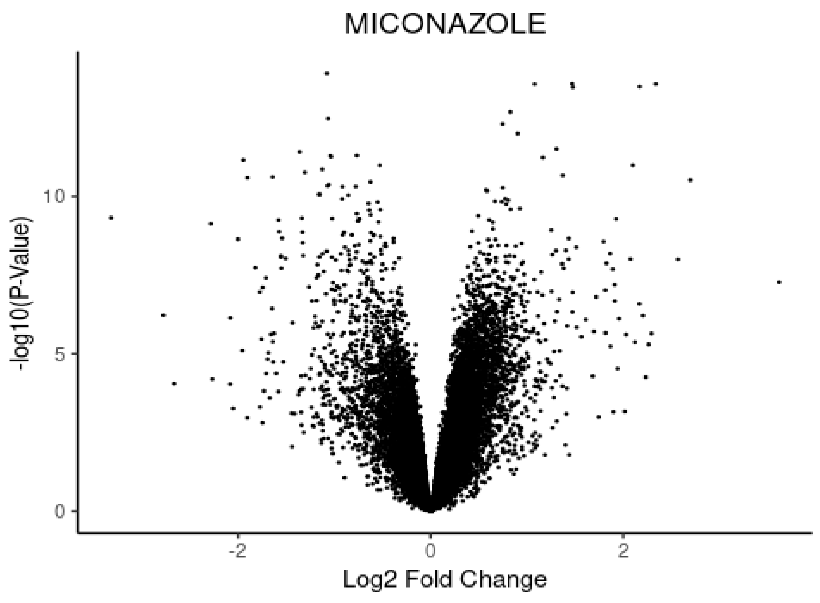
```r
ggplot(P[which(P$adj.P.Val<0.05),], aes(x=logFC)) + geom_histogram(bins = 200)
+
labs(title="PIRINIXIC ACID",x="Log2 Fold Change", y = "Count") + theme_classic(
) +
theme(plot.title = element_text(hjust = 0.5))
```



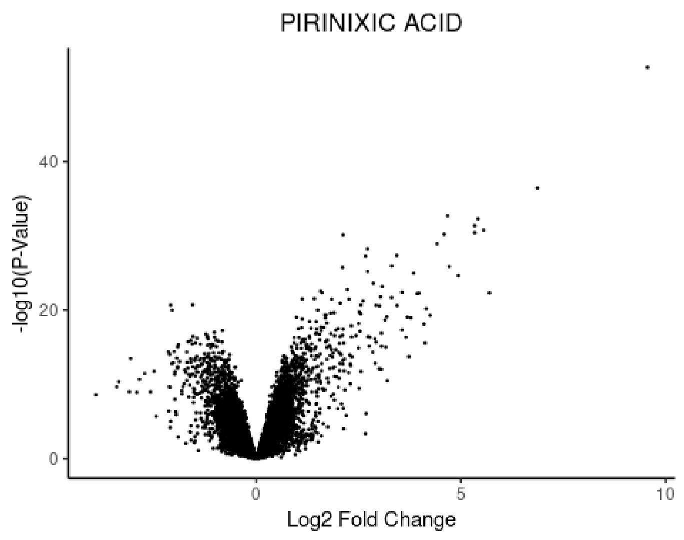
```r
# creating scatterplots
ggplot(A, aes(x=logFC, y=-log10(P.Value))) + geom_point(size=.25) +
labs(title="AFLATOXIN B1",x="Log2 Fold Change", y = "-log10(P-Value)") + theme_
classic() +
theme(plot.title = element_text(hjust = 0.5))
```



```
ggplot(M, aes(x=logFC, y=-log10(P.Value))) + geom_point(size=.25) +
labs(title="MICONAZOLE",x="Log2 Fold Change", y = "-log10(P-Value)") + theme_cl
assic() +
theme(plot.title = element_text(hjust = 0.5))
```



```
ggplot(P, aes(x=logFC, y=-log10(P.Value))) + geom_point(size=.25) +
labs(title="PIRINIXIC ACID",x="Log2 Fold Change", y = "-log10(P-Value)") + them
e_classic() +
theme(plot.title = element_text(hjust = 0.5))
```



calculating concordance

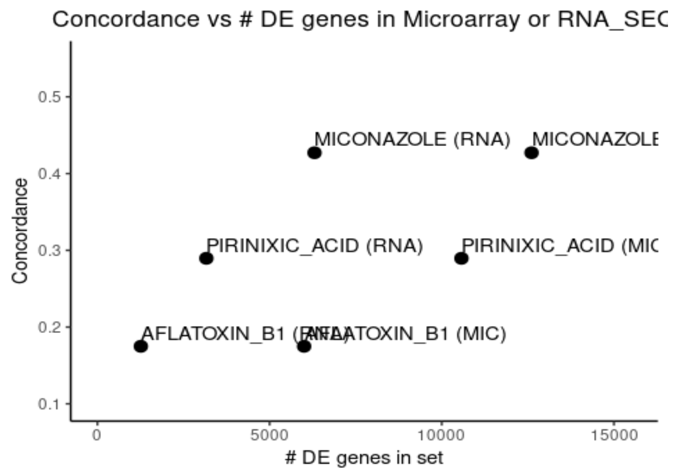
```
# only need REFSEQ and PROBEID
map <- read.csv('/project/bf528/project_3/refseq_affy_map.csv',header=TRUE, sep
= ",")[ ,c('REFSEQ', 'PROBEID')]
```

```r
# AFLATOXIN_B1
mic1 <- read.csv('limma_results_sorted_AFLATOXIN_B1.csv', header=TRUE, sep = ",
")
rna_seq1 <- read.csv('../salam/deseq_output/deseq_DNA_Damage_results.csv', head
er=TRUE, sep = ",")

# MICONAZOLE
mic2 <- read.csv('limma_results_MICONAZOLE.csv', header=TRUE, sep = ",")
rna_seq2 <- read.csv('../salam/deseq_output/deseq_CAR_PXR_results.csv', header=
TRUE, sep = ",")

# PIRINIXIC_ACID
mic3 <- read.csv('limma_results_PIRINIXIC_ACID.csv', header=TRUE, sep = ",")
rna_seq3 <- read.csv('../salam/deseq_output/deseq_PPARA_results.csv', header=TR
UE, sep = ",")

# Transform mic where it maps 8 times
#mic1 <- mic1[which(mic1[,1] %in% map$PROBEID >=8),]

# Filtering to uncorrected P-value < 0.05
mic1 <- mic1[which(mic1$P.Value<0.05),] #& abs(mic$logFC) >1.5),]
rna_seq1 <- rna_seq1[which(rna_seq1$pvalue<0.05),] #& abs(rna_seq$log2FoldChang
e) >1.5),]

mic2 <- mic2[which(mic2$P.Value<0.05),] #& abs(mic$logFC) >1.5),]
rna_seq2 <- rna_seq2[which(rna_seq2$pvalue<0.05),]

mic3 <- mic3[which(mic3$P.Value<0.05),] #& abs(mic$logFC) >1.5),]
rna_seq3 <- rna_seq3[which(rna_seq3$pvalue<0.05),]

# count number of times mic maps to rna_seq using the map as a connector
matched1 <- map[which(map$PROBEID %in% mic1[,1] & map$REFSEQ %in% rna_seq1[,1])
,]
matched2 <- map[which(map$PROBEID %in% mic2[,1] & map$REFSEQ %in% rna_seq2[,1])
,]
matched3 <- map[which(map$PROBEID %in% mic3[,1] & map$REFSEQ %in% rna_seq3[,1])
,]


#Letting N be the number of items in the whole sets,
# n1 and n2 be the numbers of items in two independent sets,
# Number of items in intersection: n1*n2/N
# n0 is the number of items in the observed intersection
N1<-nrow(mic1) # whole set
n1.1<-nrow(matched1) # independent set
n2.1<-nrow(rna_seq1) # independent set
n0.1 <- (n1.1*n2.1)/N1 # intersection
concordance1 <- 2*n0.1/(n1.1+n2.1)
```

```
N2<-nrow(mic2) # whole set
n1.2<-nrow(matched2) # independent set
n2.2<-nrow(rna_seq2) # independent set
n0.2 <- (n1.2*n2.2)/N2 # intersection
concordance2 <- 2*n0.2/(n1.2+n2.2)

N3<-nrow(mic3) # whole set
n1.3<-nrow(matched3) # independent set
n2.3<-nrow(rna_seq3) # independent set
n0.3 <- (n1.3*n2.3)/N3 # intersection
concordance3 <- 2*n0.3/(n1.3+n2.3)

conc <- c(concordance1, concordance2, concordance3,concordance1, concordance2,
concordance3)
conc_hist <- c(concordance1, concordance2, concordance3)
mic<-c(N1,N2,N3)
rna_seq<-c(n2.1,n2.2,n2.3)
comb1<-c(N1,N2,N3,n2.1,n2.2,n2.3)
print(conc)
```

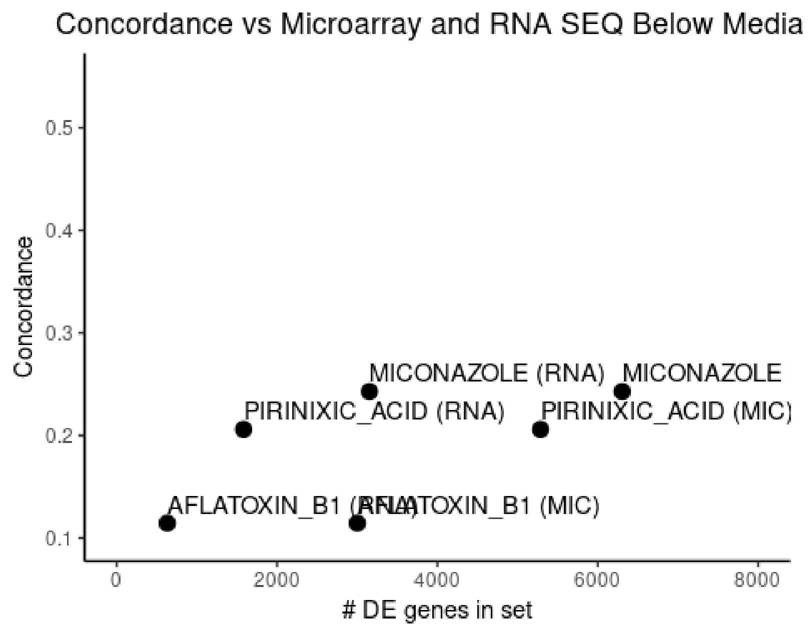
```
## [1] 0.1749437 0.4270943 0.2895639 0.1749437 0.4270943 0.2895639
```

```
df1<-data.frame(mic,conc)
df2<-data.frame(rna_seq,conc)
df_comb1 <-data.frame(comb1,conc)
Name<-c("AFLATOXIN_B1 (MIC)", "MICONAZOLE (MIC)", "PIRINIXIC_ACID (MIC)","AFLAT
OXIN_B1 (RNA)", "MICONAZOLE (RNA)", "PIRINIXIC_ACID (RNA)")

# plotting concordance
ggplot(df_comb1, aes(x=comb1, y=conc)) + geom_point(size=3) +
  labs(title="Concordance vs # DE genes in Microarray or RNA_SEQ",x="# DE genes
in set", y = "Concordance") + theme_classic() +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_text(aes(label=Name),hjust=0, vjust=-.5) + xlim(0, 15500) + ylim(.1, .55
)
```

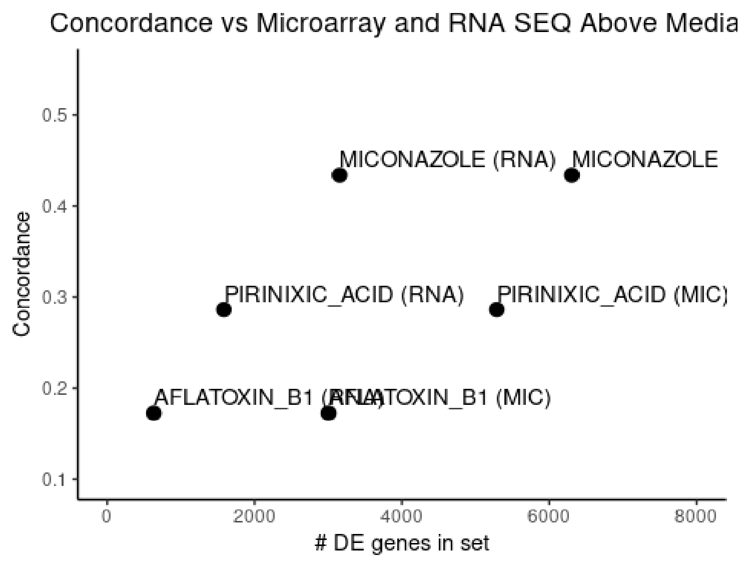


```r
# separate based on above and below median
mic1_below <- mic1[which(mic1$AveExpr<median(mic1$AveExpr)),]
rna_seq1_below <- rna_seq1[which(rna_seq1$baseMean<median(rna_seq1$baseMean)),]
mic1_above <- mic1[which(mic1$AveExpr>median(mic1$AveExpr)),]
rna_seq1_above <- rna_seq1[which(rna_seq1$baseMean>median(rna_seq1$baseMean)),]

mic2_below <- mic2[which(mic2$AveExpr<median(mic2$AveExpr)),]
rna_seq2_below <- rna_seq2[which(rna_seq2$baseMean<median(rna_seq2$baseMean)),]
mic2_above <- mic2[which(mic2$AveExpr>median(mic2$AveExpr)),]
rna_seq2_above <- rna_seq2[which(rna_seq2$baseMean>median(rna_seq2$baseMean)),]

mic3_below <- mic3[which(mic3$AveExpr<median(mic3$AveExpr)),]
rna_seq3_below <- rna_seq3[which(rna_seq3$baseMean<median(rna_seq3$baseMean)),]
mic3_above <- mic3[which(mic3$AveExpr>median(mic3$AveExpr)),]
rna_seq3_above <- rna_seq3[which(rna_seq3$baseMean>median(rna_seq3$baseMean)),]

# matched below
matched1_below <- map[which(map$PROBEID %in% mic1_below[,1] & map$REFSEQ %in% r
na_seq1_below[,1]),]
matched2_below <- map[which(map$PROBEID %in% mic2_below[,1] & map$REFSEQ %in% r
na_seq2_below[,1]),]
matched3_below <- map[which(map$PROBEID %in% mic3_below[,1] & map$REFSEQ %in% r
na_seq3_below[,1]),]

# matched above
matched1_above <- map[which(map$PROBEID %in% mic1_above[,1] & map$REFSEQ %in% r
na_seq1_above[,1]),]
matched2_above <- map[which(map$PROBEID %in% mic2_above[,1] & map$REFSEQ %in% r
na_seq2_above[,1]),]
matched3_above <- map[which(map$PROBEID %in% mic3_above[,1] & map$REFSEQ %in% r
na_seq3_above[,1]),]

# below concordance
N1_below<-nrow(mic1_below) # whole set
n1.1_below<-nrow(matched1_below) # independent set
n2.1_below<-nrow(rna_seq1_below) # independent set
n0.1_below <- (n1.1_below*n2.1_below)/N1_below # intersection
concordance1_below <- 2*n0.1_below/(n1.1_below+n2.1_below)

N2_below<-nrow(mic2_below) # whole set
n1.2_below<-nrow(matched2_below) # independent set
n2.2_below<-nrow(rna_seq2_below) # independent set
n0.2_below <- (n1.2_below*n2.2_below)/N2_below # intersection
concordance2_below <- 2*n0.2_below/(n1.2_below+n2.2_below)

N3_below<-nrow(mic3_below) # whole set
n1.3_below<-nrow(matched3_below) # independent set
n2.3_below<-nrow(rna_seq3_below) # independent set
n0.3_below <- (n1.3_below*n2.3_below)/N3_below # intersection
concordance3_below <- 2*n0.3_below/(n1.3_below+n2.3_below)
```

```r
# above concordance
N1_above<-nrow(mic1_above) # whole set
n1.1_above<-nrow(matched1_above) # independent set
n2.1_above<-nrow(rna_seq1_above) # independent set
n0.1_above <- (n1.1_above*n2.1_above)/N1_above # intersection
concordance1_above <- 2*n0.1_above/(n1.1_above+n2.1_above)
N2_above<-nrow(mic2_above) # whole set
n1.2_above<-nrow(matched2_above) # independent set
n2.2_above<-nrow(rna_seq2_above) # independent set
n0.2_above <- (n1.2_above*n2.2_above)/N2_above # intersection
concordance2_above <- 2*n0.2_above/(n1.2_above+n2.2_above)

N3_above<-nrow(mic3_above) # whole set
n1.3_above<-nrow(matched3_above) # independent set
n2.3_above<-nrow(rna_seq3_above) # independent set
n0.3_above <- (n1.3_above*n2.3_above)/N3_above # intersection
concordance3_above <- 2*n0.3_above/(n1.3_above+n2.3_above)

# below median concordance into data frame
conc_below <- c(concordance1_below, concordance2_below, concordance3_below, con
cordance1_below, concordance2_below, concordance3_below)
conc_below_hist <- c(concordance1_below, concordance2_below, concordance3_below
)
mic_below<-c(N1_below,N2_below,N3_below)
rna_seq_below<-c(n2.1_below,n2.2_below,n2.3_below)
comb2 <- c(mic_below, rna_seq_below)

df1_below<-data.frame(mic_below,conc_below)
df2_below<-data.frame(rna_seq_below,conc_below)
df_comb2 <-data.frame(comb2,conc_below)

# above median concordance into data frame
conc_above <- c(concordance1_above, concordance2_above, concordance3_above, con
cordance1_above, concordance2_above, concordance3_above)
conc_above_hist <- c(concordance1_above, concordance2_above, concordance3_above
)
mic_above<-c(N1_above,N2_above,N3_above)
rna_seq_above<-c(n2.1_above,n2.2_above,n2.3_above)
comb3 <- c(mic_above, rna_seq_above)


df1_above<-data.frame(mic_above,conc_above)
df2_above<-data.frame(rna_seq_above,conc_above)
df_comb3 <-data.frame(comb3,conc)

# plotting below
ggplot(df_comb2, aes(x=comb2, y=conc_below)) + geom_point(size=3) +
  labs(title="Concordance vs Microarray and RNA SEQ Below Median",x="# DE genes
in set", y = "Concordance") + theme_classic() +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_text(aes(label=Name),hjust=0, vjust=-.5) + xlim(0,8000) + ylim(.1, .55)
```

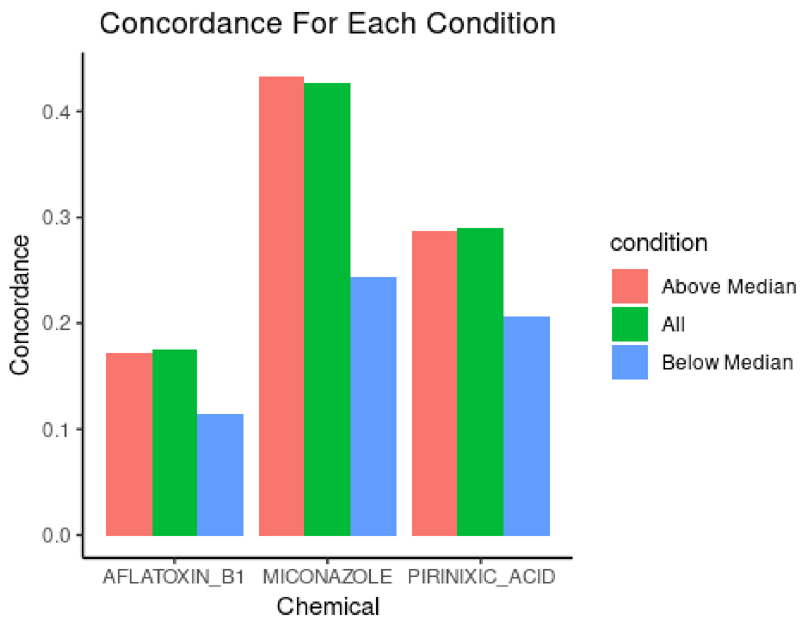


```
# plotting above
ggplot(df_comb3, aes(x=comb3, y=conc_above)) + geom_point(size=3) +
  labs(title="Concordance vs Microarray and RNA SEQ Above Median",x="# DE genes
in set", y = "Concordance") + theme_classic() +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_text(aes(label=Name),hjust=0, vjust=-.5) + xlim(0,8000) + ylim(.1, .55)
```



```
# making histogram of results
condition <- c("All", "All", "All", "Below Median" , "Below Median", "Below Med
ian", "Above Median", "Above Median", "Above Median")
chemical <- c("AFLATOXIN_B1", "MICONAZOLE", "PIRINIXIC_ACID","AFLATOXIN_B1", "M
```

```r
ICONAZOLE", "PIRINIXIC_ACID","AFLATOXIN_B1", "MICONAZOLE", "PIRINIXIC_ACID")
values<-c(conc_hist, conc_below_hist, conc_above_hist)

data<-data.frame(chemical, condition,values)

ggplot(data, aes(fill=condition, y=values, x=chemical)) +
  geom_bar(position="dodge", stat="identity") +
  labs(title="Concordance For Each Condition",x="Chemical", y = "Concordance")
+ theme_classic() +
  theme(plot.title = element_text(hjust = 0.5))
```



### Biologist

gene set enrichment analysis

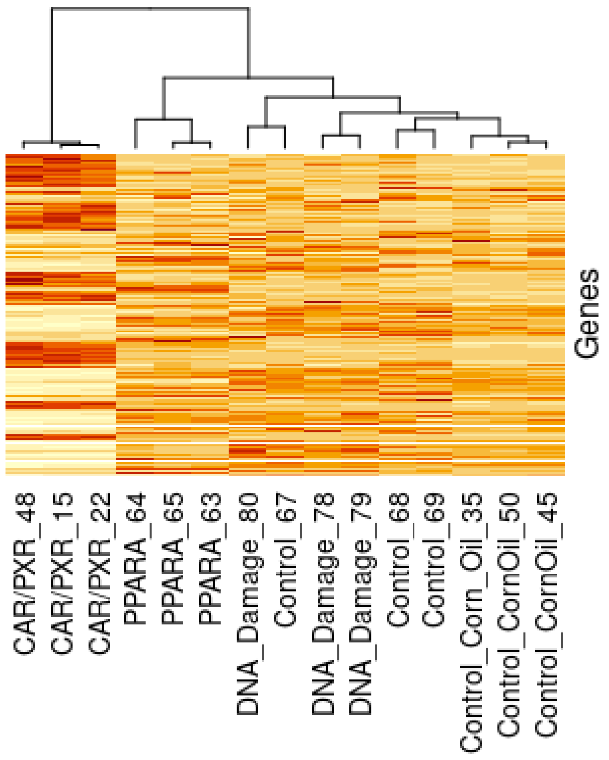
```r
#read in the filtered data (padj < 0.05)
CARPXR <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/deseq_
output/deseq_CAR_PXR_results.csv")
DNAD <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/deseq_ou
tput/deseq_DNA_Damage_results.csv")
PPARA <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/deseq_o
utput/deseq_PPARA_results.csv")

#further subset data to only include genes with |log2FoldChange| > 1
CARPXR <- subset(CARPXR, abs(log2FoldChange) > 1)
DNAD <- subset(DNAD, abs(log2FoldChange) > 1)
PPARA <- subset(PPARA, abs(log2FoldChange) > 1)

#CAR/PXR had too many genes even after filtering so took top 1000 up-regulated
and 1000 down-regulated genes for DAVID analysis
```

```
top1000 <- head(sort(CARPXR$log2FoldChange,decreasing=TRUE), n = 1000)
bottom1000 <- head(sort(CARPXR$log2FoldChange,decreasing=FALSE), n = 1000)
CARPXR_DAVID <- subset(CARPXR, CARPXR$log2FoldChange %in% top1000 | CARPXR$log2
FoldChange %in% bottom1000)
```

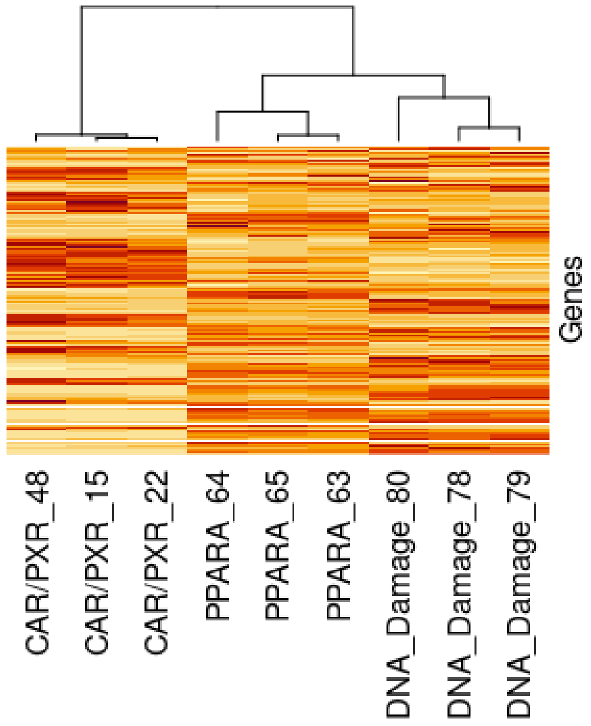
heatmap

```
#read in normalized count files
nc_CARPXR <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/des
eq_output/deseq_CAR_PXR_norm_counts.csv")
nc_DNAD <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/deseq
_output/deseq_DNA_Damage_norm_counts.csv")
nc_PPARA <- read.csv("/projectnb/bf528/users/group_5/project_3/mary/DEgene/dese
q_output/deseq_PPARA_norm_counts.csv")

#set a name for the ID column - easier for merging
colnames(nc_CARPXR)[1] <- "ID"
colnames(nc_DNAD)[1] <- "ID"
colnames(nc_PPARA)[1] <- "ID"

#averaging control values from two norm count tables that used the same set of
control samples
n <- merge(nc_DNAD[,c("ID", "SRR1178067", "SRR1178068", "SRR1178069")], nc_PPAR
A[,c("ID", "SRR1178067", "SRR1178068", "SRR1178069")], by = "ID") #pull out con
trol sample values from each table
control <- as.data.frame(sapply(c("SRR1178067", "SRR1178068", "SRR1178069"), fu
nction(x) rowMeans(n[, grep(x, names(n))]))) #calculate average
control$ID <- n$ID #add ID names to the table
control <- control[,c(4,1,2,3)] #rearrange the columns to have ID first

#merge all samples to produce a heatmap matrix
nc <- merge(nc_DNAD[,1:4], nc_PPARA[,1:4], by = "ID")
nc <- merge(nc, nc_CARPXR, by = "ID")
nc <- merge(nc, control, by = "ID")

#gene ID list for significant genes in any treatment
CARPXR_ID <- CARPXR[,1, drop = F]
DNAD_ID <- DNAD[,1, drop = F]
PPARA_ID <- PPARA[,1, drop = F]
all <- unique(rbind(CARPXR_ID,DNAD_ID,PPARA_ID))

#heatmap x-axis lables
x_axis <- c("DNA_Damage_78", "DNA_Damage_79", "DNA_Damage_80", "PPARA_63", "PPA
RA_64", "PPARA_65", "CAR/PXR_15", "CAR/PXR_22", "CAR/PXR_48", "Control_Corn_Oil
_35", "Control_CornOil_45", "Control_CornOil_50", "Control_67", "Control_68", "
Control_69")

#subset the heatmap matrix by the significant genes and produce final heatmap
allnc <- subset(nc, nc$ID %in% all$X)
rownames(allnc) <- allnc[, 1]
```

```
allnc <- as.matrix(allnc[,-1])
heatmap(allnc, labCol = x_axis, margins = c(9,1.5), labRow = F, ylab = "Genes")
```



```
#heatmap just for treatment groups
heatmap(allnc[,1:9], labCol = x_axis, margins = c(9,1.5), labRow = F, ylab = "G
enes")
```


Genes
CAR/PXR_48
CAR/PXR_15
CAR/PXR_22
PPARA_64
PPARA_65
PPARA_63
DNA_Damage_80
DNA_Damage_78
DNA_Damage_79