

School of Biological Sciences



Functional Genomic Technologies ICA Report

Author: B236494

Saturday 20th April, 2024

B236494 - Affymetrix Microarray Analysis Basic Workflow

This code has been adapted from the BasicAnalysisScript.Rmd file provided by Simon Tomlinson. This code has been altered to account for author specific input files and samples while retaining the main features of the pipeline. The R Script can be generated for this document by running the last chunk which has been turned off by default. Interpretations of the results have been provided alongside the results for convenient viewing. Comments have been added to aid in understanding the flow of execution. Please ensure that the right .CEL files are located in the working directory for the Code to work. This can be downloaded from the GEO website using the supplementary code provided at the end of the report.

Note: An additional analysis tool was identified known as GEO2R, the analysis of an entry in GEO can also be done using this. It is available at [<https://www.ncbi.nlm.nih.gov/geo/geo2r/>]

Pipeline Steps:

Loading Data & Introduction

Using the `read.AnnotatedDataFrame()` the targets are loaded from "B236494_edited_final_targets_file.txt" file into an object. The parameters ensure that the header is accounted for. Following this, the `ReadAffy()` is used to load information from the .CEL files that were previously extracted from the .tar file that was obtained from Gene Expression Omnibus (GEO).

The samples are from the study, "Endometrial epithelial ARID1A is critical for uterine gland function in early pregnancy establishment" [1] by Marquardt *et al.*, with GEO accession number GSE137166. It was an expression profiling by array experiment type.

AT-rich interaction domain 1A (ARID1A) is a large protein switch/sucrose non-fermentable (SWI/SNF) chromatin remodeling complex subunit which acts as a tumor suppressor.

The *Arid1a* gene was known to have a vital function in regulating the development of the endometrial gland which is essential for fertility after maturity and normal uterine function. Marquardt *et al.* presented evidence of how ARID1A binds and proceeds to modulate transcription of the *Foxa2* gene that is essential for endometrial gland functioning. They also showed that deletion of *Arid1a* in a uterine specific context results in a severe defect to gland development and consequentially diminished *Foxa2* and *Lif* expression.

There were two types of mice that were used to study this gene. *Arid1a* d/d and *Arid1a* f/f mice. In *Arid1a* f/f the *Arid1a* has been retained while it has been deleted in the *Arid1a* d/d mice. At 2 weeks of growth, uteri of 3 of each type of mice was selected for RNA extraction and hybridization of Affymetrix microarrays.

The sample names are as follows:

- GSM4072331 *Arid1a* f/f 1
- GSM4072332 *Arid1a* f/f 2
- GSM4072333 *Arid1a* f/f 3
- GSM4072334 *Arid1a* d/d 1
- GSM4072335 *Arid1a* d/d 2
- GSM4072336 *Arid1a* d/d 3

The goals of this analysis pipeline involves a few key steps:

- To assess the quality of the data
- Perform Normalisation of the data using Robust Multi-array Average (RMA) approach
- Functional Enrichment Analysis
- Selection of Differentially Expressed Genes

The Affy and Limma Bioconductor packages will be main tools that will be used throughout this pipeline with descriptions of the code as the execution proceeds.

```
1 ```{r dataload, echo=TRUE, warning=FALSE}
2 # Load the target file into an AnnotatedDataFrame object
3 adf <- read.AnnotatedDataFrame("B236494_edited_final_targets_file.txt",
4                               header=TRUE, row.names=1, as.is=TRUE)
5
6 # Quickly loads all CEL files in the R working directory,
7 # Refer Supplementary Material if this doesn't work
8 mydata <- ReadAffy()
9 # Viewing a summary of the AffyBatch object that has
10 # all the .CEL files information loaded into it
11 print(mydata)
12 ```
```

Build Quality Control Plots

Now that the data has been loaded into the object, the quality of the Microarray data needs to be assessed. This is important to ensure that the data has been successfully generated and that there is no severe contamination during the RNA extraction steps from the samples. Quality control can be performed by the use of box plots and a log density plot. Looking at the results of a box plot can give us an idea of the variation in data between the different replicates from each type of sample. If the difference between the replicates is large, this could imply that the extraction procedure was not done in a robust manner. The density histogram also serves the same function. Looking at these plots we can gain a preliminary idea of the quality of the data.

```
1 ```{r density_plot, echo=TRUE}
2 # png("figures/density_plot.png", width=1000, height=1000)
3 # Quality control plots 1
4 hist(mydata)
5 # dev.off()
6 ```
```

As visible from the output the density plot, the peak of expression seem to be consistent across different sample prior to normalisation which is a sign of good quality, although some variation is distinguishable and expected. This can be observed further in the boxplot below.

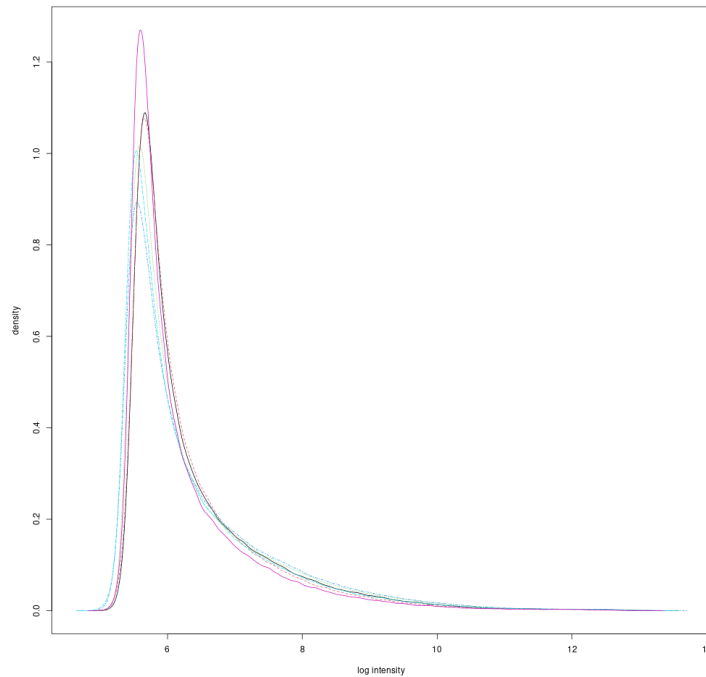


Figure 1: Density Plot

```
1 ```{r boxplot raw, echo=TRUE}
2 # Quality control plots 2
3 # png("figures/boxplot_non_normalised_2.png", width=1000, height=1000)
4 # Specifying colours for each group
5 colours <- c(rep("yellow",3),rep("red",3))
6 # Reduce font size for axis labels to increase readability
7 par(cex.axis=0.9)
8 boxplot(mydata, col=colours, las=1)
9 # dev.off()
10 ```
```

The boxplots results is promising as well. The different measures of the boxplot provide a good overview of what the data is like for each sample and between the two groups, Arid1a f/f mice in yellow and Arid1a mice in red. The median line of all the samples seems to be quite similar which indicates that the overall expression values seem to be consistent across the different samples as seen in the density plot. There is some variation within each group, as seen by the interquartile range (height of boxplots), between replicates but not to a very high level. Thus the replicates possess a good amount of similarity to each other. The lack of high variance between the two groups is representative of similar levels of expression between the two conditions.

Normalise the data using RMA

The data quality control is typically followed by normalisation. This step is important for the steps that follow downstream such as identifying significant genes after performing differential expression analyses (DEA). RMA ensures that the noisy data that can be generated for a number of reason during microarray analyses is negated. It also performs a log transformation of the data which can help in the stablisation of

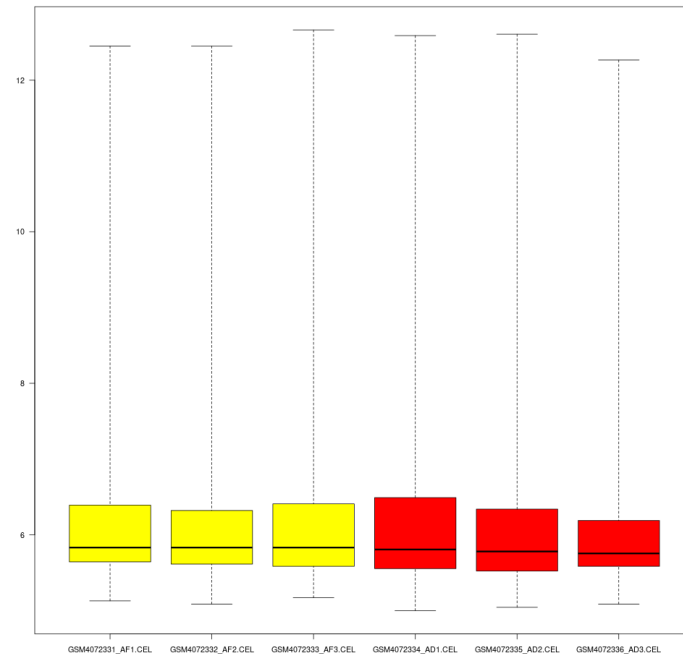


Figure 2: Boxplot Prior to Normalisation

variance across different intensities observed. It finally summarises the complex data to give us a simple expression value for each gene of the samples present. This can be used in DEA.

```
1 ```{r normalise using RMA, echo=FALSE}
2 # Performing the actual rma step
3 eset <- rma(mydata)
4 # uncomment below to view what eset is
5 # eset
6 # A matrix of the expression values using exprs() of expression values
7 values <- exprs(eset)
8 ```
```

The object "values" contains the individual expression values for each gene in our list and the individual expression value for each gene for a particular sample.

| | Arid1a_f/f_1 | Arid1a_f/f_2 | Arid1a_f/f_3 | Arid1a_d/d_1 | Arid1a_d/d_2 | Arid1a_d/d_3 |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 1415670_at | 6.577893 | 6.858240 | 7.010284 | 6.750711 | 7.011346 | 7.075572 |
| 1415671_at | 8.399974 | 8.327613 | 8.459953 | 8.365296 | 8.286804 | 8.439266 |
| 1415672_at | 8.683244 | 8.750602 | 8.738313 | 8.697099 | 8.739039 | 8.672863 |
| 1415673_at | 4.679931 | 4.534878 | 4.514374 | 4.646153 | 4.928907 | 4.878249 |
| 1415674_a_at | 6.521778 | 6.635758 | 6.615498 | 6.483207 | 6.676559 | 6.628417 |
| 1415675_at | 6.631210 | 6.806931 | 6.919652 | 6.581962 | 6.763106 | 6.758566 |
| 1415676_a_at | 8.214126 | 8.388565 | 8.516988 | 8.120327 | 8.460204 | 8.548472 |
| 1415677_at | 6.372277 | 6.621868 | 6.815158 | 6.395894 | 6.706313 | 6.706860 |
| 1415678_at | 8.060656 | 7.813556 | 7.748168 | 7.900237 | 7.600279 | 7.716913 |
| 1415679_at | 8.917447 | 9.018012 | 8.947559 | 8.999199 | 9.018211 | 9.035668 |

Figure 3: values object

Plotting Normalised Data

Now that we have the normalised data, we can plot these values to observe the changes that have been made following the RMA step. A boxplot will be used again to observe the changes. The use of the `mva.pairs()` is also employed which generates a MA plot for all the samples. The MA plot, which is described as an application of Bland-Altman plot, is a method of visually representing genomic data. It works by visualising the differences among measures taken in 2 different samples "by transforming the data onto M (log ratio) and A (mean average) scales, then plotting these values." This can be done for every sample against every sample which is what has been done here for both the normalised and non-normalised data. It provides valuable insight into the differences between samples and there is genuine difference between the groups. If there was no difference then it would imply that there has been no effect on the induced condition on the group of mice whose *Arid1a* gene has been deleted which is not consistent with what is known. It gives confidence that the experiment was performed well.

First, looking at the boxplot,

```
1 ````{r plot_normalised_boxplot, echo=TRUE }
2 # Boxplot to observe the results of normalisation
3 # png("figures/boxplot_normalised_2.png", width=1000, height=1000)
4 # Increase font size for axis labels to increase readability
5 par(cex.axis=1.4)
6 boxplot(values, col=colours, las=1)
7 # dev.off()
8 ````
```

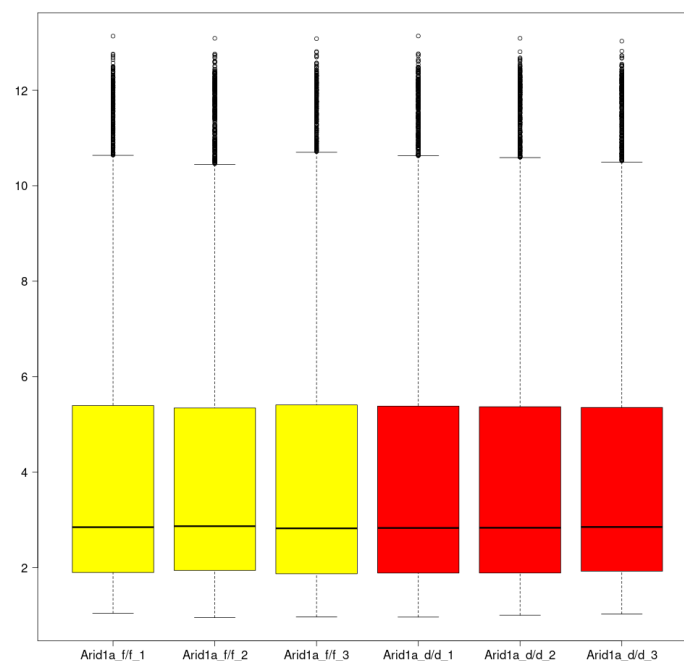


Figure 4: Boxplot Post Normalisation

Normalisation has been performed accurately. The presence of outliers could be interesting and something that we will explore downstream. The interquartile range is seen to be similar across the different groups, which means they should be suitable for comparison for significant genes.

Below, both the MVA plots will be compared.

```
1 ```{r plot_normalised mva without normalisation}
2 # The mva plot for the non-normalised raw data
3 # png("figures/mva_without_normalisation.png", width=1000, height=1000)
4 par(cex.axis=0.3, cex.lab=0.5, cex=0.5)
5 mva.pairs(pm(mydata))
6 # dev.off()
7 ```
```

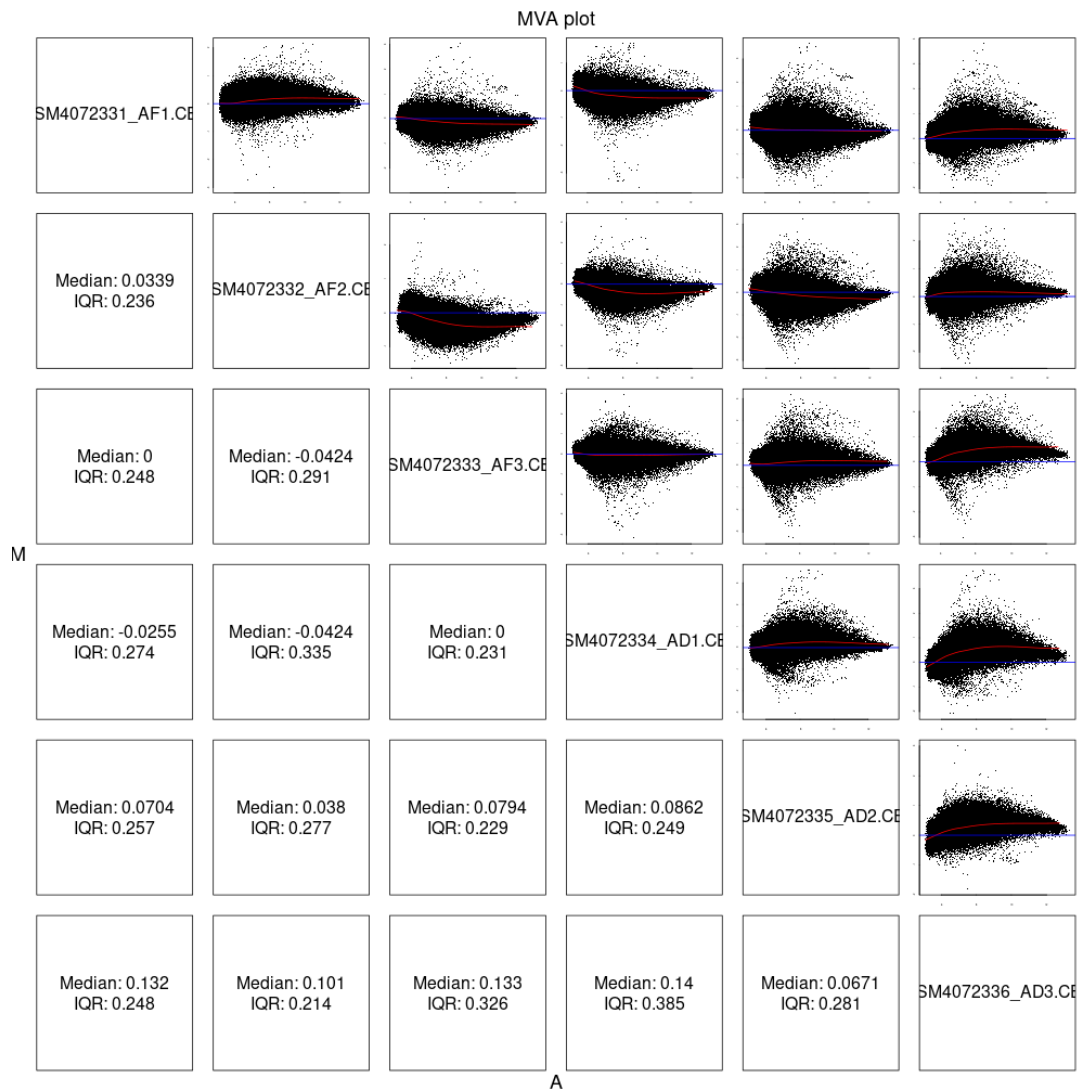


Figure 5: MVA plot for the non-normalised raw data

```
1 ```{r plot_normalised mva with normalisation}
2 # MA plot of all the samples which have been normalised
3 # png("figures/mva_with_normalisation.png", width=1000, height=1000)
4 par(cex.axis=0.7, cex.lab=0.5, cex=0.5)
5 mva.pairs(values)
6 # dev.off()
7 ```
```

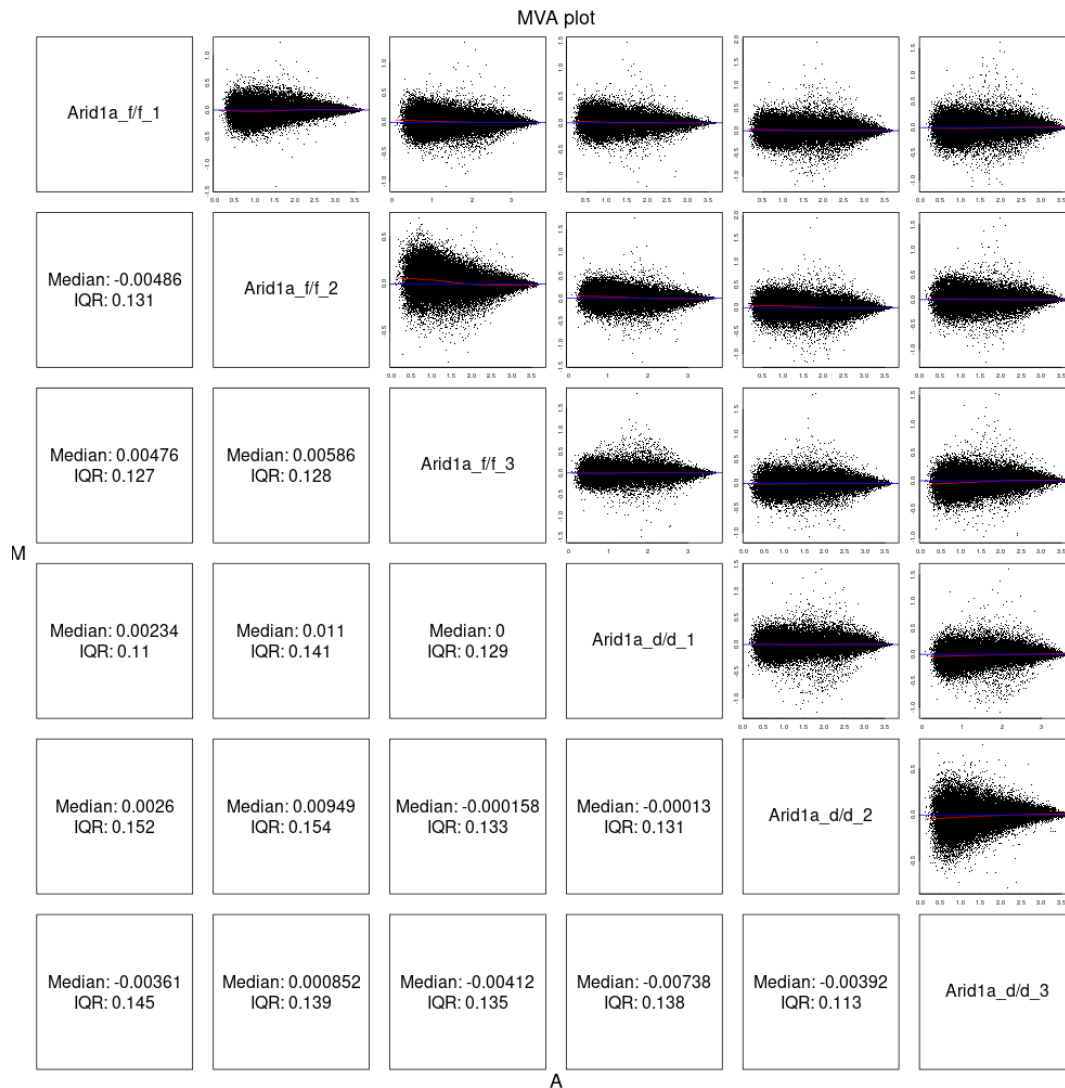


Figure 6: MVA plot for the normalised data

Observing both MVA plots we can make some inferences. The consistency across the replicates from the same group i.e same biological condition implies that normalisation has been carried out successfully and a comparison between these samples is valid. Although there is consistency among the samples in a group there is still a difference between the two groups in the expression. This is promising because this means that the onset of the condition actually has resulted in a change in the expression levels.

As the data has been normalised, we can now proceed to the next stage of identifying the significant genes in the dataset.

Plot Cluster Dendrogram

To get an idea which samples exhibit similar expression patterns, a hierarchical clustering is done. Using the `hclust()` a dendrogram of the hierarchical clusters is visualised.


```

1 ```{r dendrogram, echo=TRUE}
2 # png("figures/Cluster_Dendrogram.png", width=500, height=500)
3 colnames(values) <- rownames(pData(adf))
4 # Performs hierarchical clustering with average linkage based on
5 # Pearsons Correlation Coefficient
6 hc<-hclust(as.dist(1-cor(values, method="pearson")), method="average")
7 plot(hc)
8 # dev.off()
9 ```

```

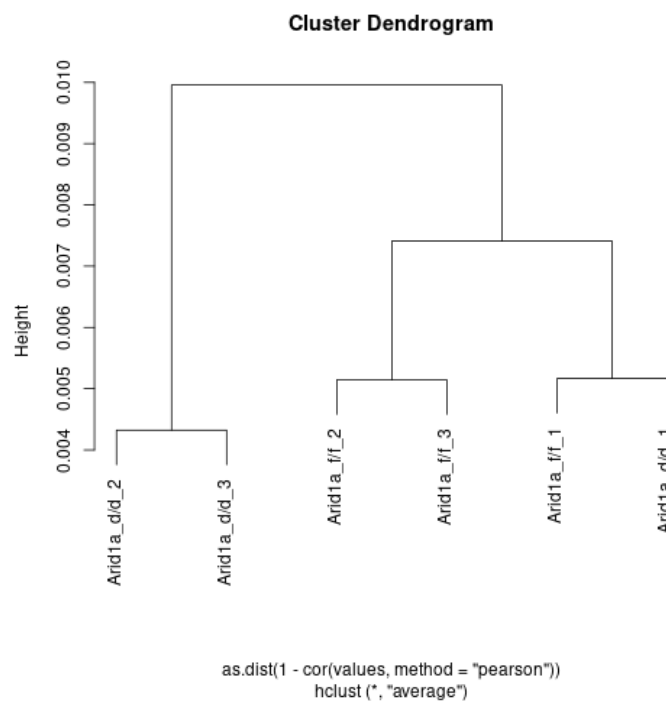


Figure 7: Cluster Dendrogram

We could expect that samples will in 2 clusters of 3 as there are 3 samples for each group but that is not the case here. One of the f_f group has a similar expression pattern to the d_d group while the other samples cluster with samples of the same group. This could be because of an atypical sample or something of biological relevance. More samples would be required to further analyse this clustering which is unfortunately not available.

Performing PCA

Principal Component Analysis (PCA) is a dimensionality reduction technique that converts the original variables into a much smaller number of principal components that explain the variability in the data best. It is an interesting way to see how the samples vary and which samples vary in the most similar fashion. `prcomp()` and `scatterplot3d()` is used to visualise the first 3 Principal components.

```

1 ```{r pca_normalised, echo=TRUE}
2 #png("figures/PCA.#png", width=850, height=600)
3 pca <- prcomp(t(values))
4 # Plot the PCA results
5 s3d<-scatterplot3d(pca$x[,1:3], pch=19, color=rainbow(1))
6 s3d.coords <- s3d$xyz.convert(pca$x[,1:3])
7 text(s3d.coords$x, s3d.coords$y, labels = colnames(values),pos = 3,offset =
8       0.5)
9 # dev.off()
10 ```

```

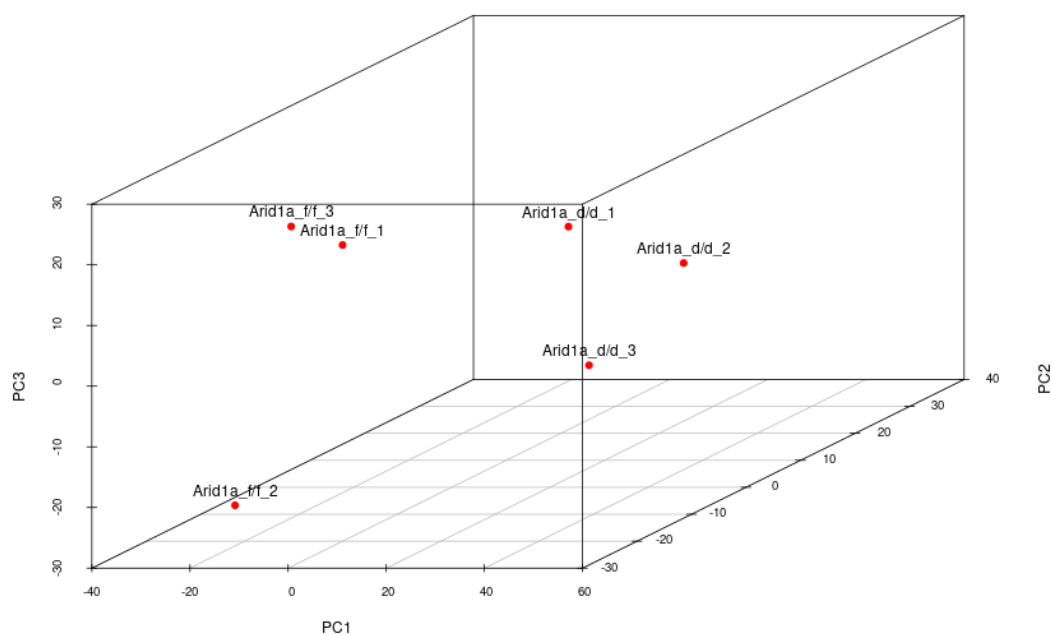


Figure 8: PCA plot of 3 Principal Components

We can see a clear variability between both the groups along PC1 (x axis) which clearly implies that there is clear variability among both the groups of our samples which is what is expected given the nature of the two groups. This further supports our confidence of obtaining significant genes when we try to extract them further down the pipeline.

Perform fold filtering

In this section, we first calculate the average fold change expression differences of the groups, First the group means are calculated then the comparison is done by building the contrast matrix and then the model fitting is performed. `topTable()` is used to obtain a dataframe "myresults" which contains many values for each gene for individual samples, such as logFC, adjusted p-value, B value, etc which is useful for filtering for significant genes.

The top genes with high FC difference are displayed in the image below.

```

1  ```{r fold_filtering, include=FALSE}
2
3  # obtaining a matrix of expression values
4  exprsvals <- exprs(eset)
5  # RMA outputs log2 data while MAS5 outputs linear data
6  # To convert from log
7  exprsvals10 <- 2^exprsvals
8  # check conversion
9  exprsvals[1:10,]
10 # converted
11 exprsvals10[1:10,]
12
13 # More fold filtering
14 # check order of sample names
15 mysamples <- sampleNames(eset)
16 # display the list
17 mysamples
18 # it is useful to obtain a vector of ProbeIDs here
19 probesets <- probeNames(mydata)
20 #display the first 10 ProbeSets
21 probesets[1:10]
22
23 # Build final fold table
24 # Calculate the means
25 Aridla_f_f.mean <- apply(exprsvals10[,c("GSM4072331_AF1.CEL", "GSM4072332_AF2.
    CEL","GSM4072333_AF3.CEL")],1,mean)
26
27 Aridla_d_d.mean <- apply(exprsvals10[,c("GSM4072334_AD1.CEL", "GSM4072335_AD2.
    CEL","GSM4072336_AD3.CEL")],1,mean)
28 #calculate some fold changes
29 Aridla_fold_change <- Aridla_f_f.mean/Aridla_d_d.mean
30
31 #build a summary table to hold all the data
32
33 all.data= cbind(Aridla_fold_change)
34 # check the column names
35 colnames(all.data)
36 # write the table of means as an output
37 write.table(all.data,file="B236494_group_means.txt", quote=F,
38 sep="\t",col.names=NA)
39 ```

```

| | Arid1a_fold_change |
|--------------|--------------------|
| 1421404_at | 14.040558 |
| 1434202_a_at | 11.993181 |
| 1453801_at | 5.652052 |
| 1455898_x_at | 5.239522 |
| 1452320_at | 5.087654 |
| 1437052_s_at | 5.085540 |
| 1434203_at | 4.855596 |
| 1427133_s_at | 4.828117 |
| 1456428_at | 4.615851 |
| 1439543_at | 4.160353 |
| 1416111_at | 3.937733 |
| 1437361_at | 3.933379 |
| 1439568_at | 3.583643 |
| 1429286_at | 3.436704 |
| 1422612_at | 3.413425 |

Figure 9: Genes with highest FC between groups

Beginning statistical analysis

In this section, the statistical tests will be performed using the `makeContrasts()`, `lmFit()`, `eBayes()`, and `topTable()`. At the end we obtain a txt file with the results from the `topTable()` with the values mentioned earlier.

```

1 ``{r limma_stats, echo=TRUE}
2 # Check original sample order
3 sampleNames(eset)
4 # Rename the samples
5 sampleNames(eset) <-
6 c("GSM4072331_AF1.CEL", "GSM4072332_AF2.CEL", "GSM4072333_AF3.CEL", "GSM4072334_
7   AD1.CEL", "GSM4072335_AD2.CEL", "GSM4072336_AD3.CEL")
8 # Check the samples have renamed
9 sampleNames(eset)
10 ``
11 ``{r building_annotation, echo=TRUE}
12 # Building annotation for differential gene identification
13 # establish annotation for MOE430v2
14 # which annotation do we need
15 # modified from
16 # http://gettinggeneticsdone.blogspot.co.uk/2012/01/
17 # annotating-limma-#results-with-gene.html
18 eset@annotation
19
20 # build an annotation table
21 ID <- featureNames(eset)
22 Symbol <- getSYMBOL(ID, "mouse4302.db")
23 Name <- as.character(lookup(ID, "mouse4302.db", "GENENAME"))
24 tmp <- data.frame(ID=ID, Symbol=Symbol, Name=Name, stringsAsFactors=F)
25 tmp[tmp=="NA"] <- NA #fix padding with NA characters
26 #assign as feature data of the current Eset

```

```

27 fData(eset) <- tmp
28
29 # Check for rows that dont have an EntrezID
30 missing_entrezid_count <- sum(is.na(fData(eset)$ENTREZID))
31
32 print(missing_entrezid_count)
33 ```
34 ```{r limma_statistical_analysis, echo=TRUE}
35 # Build the design matrix
36 design <- model.matrix(~-1+factor(c(1,1,1,2,2,2)))
37 colnames(design) <- c("Arid1a_f_f","Arid1a_d_d")
38
39 # This instructs Limma which comparisons to make
40 contrastmatrix <- makeContrasts(Arid1a_f_f-Arid1a_d_d,levels=design)
41 contrastmatrix
42
43 # issue these commands to fit the model
44 #and make the contrasts
45 fit <- lmFit(eset, design)
46
47 fit2 <- contrasts.fit(fit, contrastmatrix)
48
49 # this last part essentially moderates the t-statistic using
50 # the borrowed variance approach described in class
51 fit2 <- eBayes(fit2)
52
53 # get the results
54 topTable(fit2,coef=1,adjust="fdr")
55 myresults <-topTable(fit2,coef=1, adjust="fdr", number=nrow(eset))
56 write.table(myresults,"B236494_myresults.txt")
57
58 # Results from sorting by logFC
59 myresults_sorted_by_abs_logFC <- myresults[order(-abs(myresults$logFC)), ]
60 top_10_topleft_FC <- head(myresults_sorted_by_abs_logFC, 10)
61
62 # Results from sorting by adjusted p-value
63 myresults_sorted_by_adj_p_value <- myresults[order(myresults$adj.P.Val), ]
64 top_10_topleft_by_adj_p_value <- head(myresults_sorted_by_adj_p_value, 10)
65
66 # Finding genes with common gene symbol
67 common_genes_logFC_and_adj_p_value <- intersect(top_10_topleft_FC$Symbol, top_
68 10_topleft_by_adj_p_value$Symbol)
69 print(common_genes_logFC_and_adj_p_value)
70
71 # Querying original df to obtain information
72 # about gene symbols which are common
73 common_genes_logFC_and_adj_p_value_information <- myresults[myresults$Symbol %
74 in% common_genes_logFC_and_adj_p_value, ]
75 print(common_genes_logFC_and_adj_p_value_information)
76 ```

```

Looking at the "myresults" dataframe we can sort by different parameters and compare what genes are significant based on these values.

These are the results of sorting by logFC and adjusted p-value:

There are actually some gene symbols which are present in both lists. These are displayed below:

| | ID | Symbol | Name | logFC | AveExpr | t | P.Value | adj.P.Val | B |
|--|--------------|---------|---|-----------|----------|-----------|--------------|-------------|-----------|
| | 1434202_a_at | Fam107a | family with sequence similarity 107, member A | 3.994493 | 4.796256 | 7.162173 | 9.342991e-05 | 0.029262376 | 1.9254999 |
| | 1421404_at | Cxcl15 | chemokine (C-X-C motif) ligand 15 | 3.711493 | 3.725324 | 11.564940 | 2.717877e-06 | 0.006414173 | 4.9494952 |
| | 1455996_x_at | Prap1 | proline-rich acidic protein 1 | -3.427950 | 6.098618 | -7.101665 | 9.919010e-05 | 0.030432468 | 1.8698763 |
| | 1419167_at | Prap1 | proline-rich acidic protein 1 | -3.285508 | 6.135464 | -6.565275 | 1.715247e-04 | 0.040291321 | 1.3554890 |
| | 1454691_at | Nrxn1 | neurexin I | -2.521008 | 3.601459 | -6.302187 | 2.270802e-04 | 0.047635099 | 1.0886217 |
| | 1453801_at | Them5 | thioesterase superfamily member 5 | 2.483172 | 3.258623 | 16.864196 | 1.460978e-07 | 0.002522325 | 6.9176558 |
| | 1455898_x_at | Slc2a3 | solute carrier family 2 (facilitated glucose transporter), mem... | 2.435006 | 5.857540 | 6.010194 | 3.130837e-04 | 0.053894608 | 0.7806458 |
| | 1437052_s_at | Slc2a3 | solute carrier family 2 (facilitated glucose transporter), mem... | 2.359687 | 7.255791 | 5.978429 | 3.244229e-04 | 0.054074020 | 0.7463758 |
| | 1452320_at | Lrp2 | low density lipoprotein receptor-related protein 2 | 2.318852 | 3.309815 | 10.104758 | 7.555234e-06 | 0.010090185 | 4.1372088 |
| | 1427133_s_at | Lrp2 | low density lipoprotein receptor-related protein 2 | 2.318153 | 4.286835 | 9.190835 | 1.533047e-05 | 0.012571264 | 3.5429495 |

Figure 10: Top 10 genes sorted by logFC (topTable)

| | ID | Symbol | Name | logFC | AveExpr | t | P.Value | adj.P.Val | B |
|--|--------------|---------------|---|-----------|----------|-----------|--------------|-------------|----------|
| | 1456428_at | Cxcl15 | chemokine (C-X-C motif) ligand 15 | 2.198971 | 3.152199 | 20.33287 | 3.348572e-08 | 0.001510240 | 7.690656 |
| | 1453801_at | Them5 | thioesterase superfamily member 5 | 2.483172 | 3.258623 | 16.86420 | 1.460978e-07 | 0.002522325 | 6.917656 |
| | 1432339_at | 4933432I03Rik | RIKEN cDNA 4933432I03 gene | -1.835624 | 3.417871 | -16.56918 | 1.677785e-07 | 0.002522325 | 6.837358 |
| | 1455186_a_at | Urah | urate (5-hydroxyiso-) hydrolase | -1.301675 | 6.224570 | -15.15968 | 3.361917e-07 | 0.003790645 | 6.414126 |
| | 1425510_at | Mark1 | MAP/microtubule affinity regulating kinase 1 | 1.181600 | 4.905757 | 13.86973 | 6.712648e-07 | 0.005558508 | 5.960645 |
| | 1448265_x_at | Mpzl2 | myelin protein zero-like 2 | -1.749585 | 5.920743 | -13.69758 | 7.394747e-07 | 0.005558508 | 5.894653 |
| | 1439568_at | Greb1 | gene regulated by estrogen in breast cancer protein | 1.826744 | 3.217314 | 13.10424 | 1.041416e-06 | 0.005576738 | 5.656307 |
| | 1423844_s_at | Cbs | cystathionine beta-synthase | 1.093131 | 4.304038 | 12.71355 | 1.315100e-06 | 0.005576738 | 5.489600 |
| | 1441662_at | Cyp4x1 | cytochrome P450, family 4, subfamily x, polypeptide 1 | -1.741845 | 3.198246 | -12.69951 | 1.326336e-06 | 0.005576738 | 5.483456 |
| | 1453442_at | 2310043M15Rik | RIKEN cDNA 2310043M15 gene | -1.510514 | 4.949019 | -12.63547 | 1.378966e-06 | 0.005576738 | 5.455301 |

Figure 11: Top 10 genes sorted by adjusted p-value (topTable)

```
[1] "Cxc115" "Them5"
```

Figure 12: Common gene symbols in both logFC and adjusted p value lists

| | ID | Symbol | Name | logFC | AveExpr | t | P.Value | adj.P.Val | B |
|--|--------------|--------|-----------------------------------|-----------|----------|-----------|--------------|-------------|-----------|
| | 1456428_at | Cxcl15 | chemokine (C-X-C motif) ligand 15 | 2.1989714 | 3.152199 | 20.332871 | 3.348572e-08 | 0.001510240 | 7.690656 |
| | 1453801_at | Them5 | thioesterase superfamily member 5 | 2.4831717 | 3.258623 | 16.864196 | 1.460978e-07 | 0.002522325 | 6.917656 |
| | 1421404_at | Cxcl15 | chemokine (C-X-C motif) ligand 15 | 3.7114931 | 3.725324 | 11.564940 | 2.717877e-06 | 0.006414173 | 4.949495 |
| | 1431211_s_at | Them5 | thioesterase superfamily member 5 | 1.3810733 | 3.231990 | 9.947131 | 8.501520e-06 | 0.010090185 | 4.039837 |
| | 1429354_at | Them5 | thioesterase superfamily member 5 | 0.4196961 | 1.736084 | 3.605875 | 6.861461e-03 | 0.218544329 | -2.264333 |

Figure 13: Which gene IDs have these common gene symbols in both logFC and adjusted p value lists

Functional Enrichment Analysis

In this section, Functional Enrichment Analysis (FEa) will be performed. This involves a few keys steps; Preparation of the gene set object for enrichment analysis, Check expression data for apt annotation, Perform the actual FEA, share the output.

The reason why functional enrichment analysis is done is to find gene sets that are highly over-represented in certain biological functions or pathways (from MsigDB "Mm.H" here). The analysis method is chosen based on the study. Here, romer (Rotation gene set tests of MEan Ranks) is a more suitable choice due to a number of reasons. It is a competitive test where it pits the different gene sets against one another. It evaluates changes at the level of gene sets which is suitable as it can consider the behaviour of genes as a collective within certain defined pathways or processes. It is also suitable to detect very subtle changes in the expression across a gene set which seems to be the case from the plots earlier. We can expect the

expression to be downregulated as we know the d_d group has the gene deleted so the expression would be diminished. There are also other options such as mroast and camera but romer seems to be the best choice here.

```

1  ```{r functional_enrichment_setup , echo=TRUE}
2  Mm.H <- readRDS("/shared_files/MSigDB/Mm.h.all.v7.1.entrez.rds")
3  # Show the annotation keys in this database
4  keytypes(mouse4302.db)
5  # Check sampleNames
6  sampleNames(eset)
7  ```
8  ```{r process_annotation_for_enrichment, echo=TRUE}
9  # Here we select from the annotation a number of keys with the primary key
   being PROBEID
10 res <- select(mouse4302.db, keys = rownames(eset), columns = c("ENTREZID", "
   ENSEMBL", "SYMBOL"), keytype="PROBEID")
11 # find the index of each row of the expression set in the
12 # annotation object res
13 idx <- match(rownames(eset), res$PROBEID)
14 # Use the index to set the phenotypic data in the ExpressionSet
15 fData(eset) <- res[idx, ]
16 head(fData(eset), 10)
17 # Find all rows that dont have an EntrezID and remove them
18 eset_t<-eset[is.na(fData(eset)$ENTREZID)==0,]
19 ```
20 ```{r convert_indices, echo=TRUE}
21 # convert to indexes
22 H.indices <- ids2indices(Mm.H, fData(eset_t)$ENTREZID)
23 # Running romer
24 results_romer <- romer(eset_t, index=H.indices, design=design, contrast=
   contrastmatrix[,1], adjust.method = "BH")
25 # Uncomment to View the results of Romer
26 results_romer
27 # Results saved to a txt file
28 write.table(results_romer, "B236494_romer_final_enrichment.txt", sep="\t")
29
30 ```

```

| | NGenes | Up | Down | Mixed |
|-------------------------------------|--------|--------|--------|--------|
| HALLMARK_ESTROGEN_RESPONSE_LATE | 561 | 0.0251 | 0.9750 | 0.0001 |
| HALLMARK_WNT_BETA_CATENIN_SIGNALING | 151 | 0.0475 | 0.9526 | 0.0001 |
| HALLMARK_ANGIOGENESIS | 157 | 0.0558 | 0.9443 | 0.0001 |
| HALLMARK_INFLAMMATORY_RESPONSE | 551 | 0.1178 | 0.8823 | 0.0001 |
| HALLMARK_TNFA_SIGNALING_VIA_NFKB | 529 | 0.1418 | 0.8583 | 0.0001 |
| HALLMARK_KRAS_SIGNALING_UP | 629 | 0.1883 | 0.8118 | 0.0001 |
| HALLMARK_ESTROGEN_RESPONSE_EARLY | 654 | 0.2741 | 0.7260 | 0.0001 |
| HALLMARK_XENOBIOTIC_METABOLISM | 563 | 0.0977 | 0.9024 | 0.0003 |
| HALLMARK_UV_RESPONSE_DN | 751 | 0.2708 | 0.7293 | 0.0003 |
| HALLMARK_CHOLESTEROL_HOMEOSTASIS | 200 | 0.2382 | 0.7619 | 0.0004 |
| HALLMARK_PEROXISOME | 336 | 0.4289 | 0.5712 | 0.0004 |
| HALLMARK_NOTCH_SIGNALING | 115 | 0.0617 | 0.9384 | 0.0006 |
| HALLMARK_APOPTOSIS | 476 | 0.2062 | 0.7939 | 0.0012 |
| HALLMARK_ANDROGEN_RESPONSE | 396 | 0.4267 | 0.5734 | 0.0014 |

Figure 14: Romer results sorted by Mixed Category Column

Looking at the results, a few main results come to notice, when sorted by "Mixed" category:

"HALLMARK_ESTROGEN_RESPONSE_LATE", "HALLMARK_APOPTOSIS" and "HALLMARK_ESTROGEN_RESPONSE_EARLY" have low p-values in the "Mixed" category. We know that the deletion of the *Arid1a* gene has effects on hormones and development of the endometrium. This could be signifying the complex bidirectional dysregulation that happens when this gene is removed and the resultant hormonal imbalances and associated cell death that could be occurring.

Selecting Differentially expressed genes in the dataset

In this section, the selection of differentially expressed genes from the current expression set will be explored.

Here using a Volcano plot and the `topTreat()`, an attempt to select differentially expressed genes will be made and compared to the results from the previous section to establish if it is consistent or the result varies between different techniques and functions.

```

1  ```{r volcano_plot_limma_basic, echo=TRUE, fig.cap="Volcano plot of
    differential expression using limma"}
2  #png("figures/volcano_plot_limma.png", width=1500, height=750)
3  # Using the volcanoplot function to create the plot
4  volcanoplot(fit2, coef=1, style="p-value", highlight=10,
5              names=rownames(fit2$coefficients),
6              xlab="Log2 Fold Change", ylab="-Log10 P-value",
7              pch=20, cex=1, main="Volcano Plot")
8  #dev.off()
9  ```

```

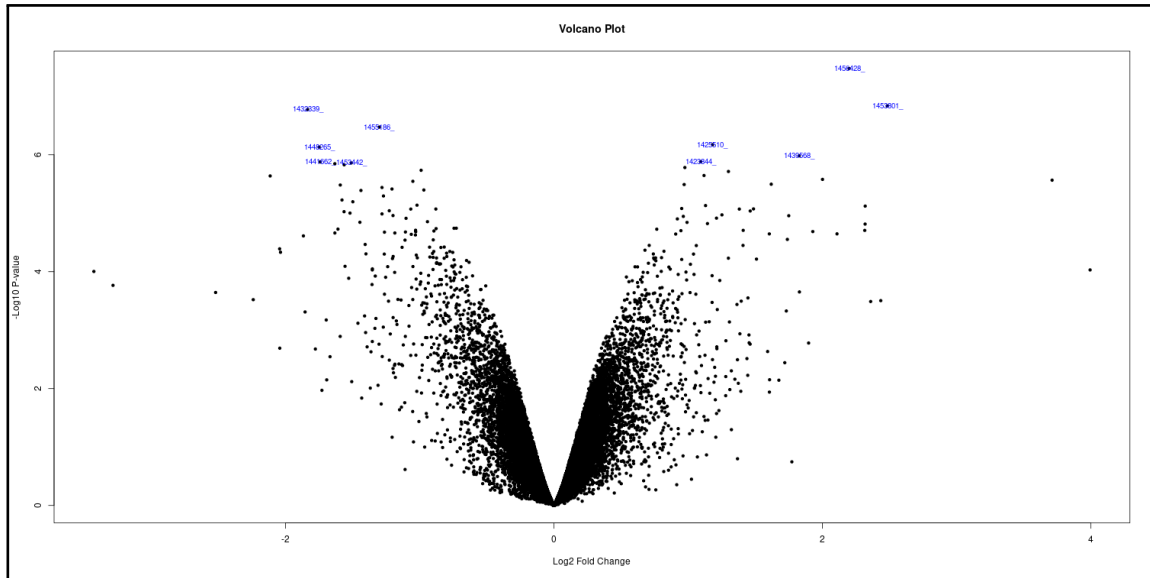



Figure 15: Volcano plot with top 10 significant genes

```

1 ```{r topTreat code }
2 # Extract top genes using topTreat, focusing on those exceeding the fold-change
  threshold
3 topTreat_myresults <- topTreat(fit2, coef=1, adjust="BH", sort.by="p", number=
  Inf)
4 write.table(topTreat_myresults, "B236494_topTreat_results.txt", sep="\t", quote
  =FALSE)
5 top_genes_topTreat_by_adj_p_value <- topTreat_myresults[order(topTreat_
  myresults$adj.P.Val), ][1:10,]
6 print(top_genes_topTreat_by_adj_p_value)
7 ```

```

| | ID | Symbol | Name | logFC | AveExpr | t | PValue | adj.PVal | B |
|--------------|--------------|---------------|---|-----------|----------|-----------|--------------|-------------|----------|
| 1456428_at | 1456428_at | Cxcl15 | chemokine (C-X-C motif) ligand 15 | 2.198971 | 3.152199 | 20.33287 | 3.348572e-08 | 0.001510240 | 7.690656 |
| 1453801_at | 1453801_at | Them5 | thioesterase superfamily member 5 | 2.483172 | 3.258623 | 16.86420 | 1.460978e-07 | 0.002522325 | 6.917656 |
| 1432339_at | 1432339_at | 4933432I03Rik | RIKEN cDNA 4933432I03 gene | -1.835624 | 3.417871 | -16.56918 | 1.677785e-07 | 0.002522325 | 6.837358 |
| 1455186_a_at | 1455186_a_at | Urah | urate (5-hydroxyiso-) hydrolase | -1.301675 | 6.224570 | -15.15968 | 3.361917e-07 | 0.003790645 | 6.414126 |
| 1425510_at | 1425510_at | Mark1 | MAP/microtubule affinity regulating kinase 1 | 1.181600 | 4.905757 | 13.86973 | 6.712648e-07 | 0.005558508 | 5.960645 |
| 1448265_x_at | 1448265_x_at | Mpzl2 | myelin protein zero-like 2 | -1.749585 | 5.920743 | -13.69758 | 7.394747e-07 | 0.005558508 | 5.894653 |
| 1439568_at | 1439568_at | Greb1 | gene regulated by estrogen in breast cancer protein | 1.826744 | 3.217314 | 13.10424 | 1.041416e-06 | 0.005576738 | 5.656307 |
| 1423844_s_at | 1423844_s_at | Cbs | cystathionine beta-synthase | 1.093131 | 4.304038 | 12.71355 | 1.315100e-06 | 0.005576738 | 5.489600 |
| 1441662_at | 1441662_at | Cyp4x1 | cytochrome P450, family 4, subfamily x, polypeptide 1 | -1.741845 | 3.198246 | -12.69951 | 1.326336e-06 | 0.005576738 | 5.483456 |
| 1453442_at | 1453442_at | 2310043M15Rik | RIKEN cDNA 2310043M15 gene | -1.510514 | 4.949019 | -12.63547 | 1.378966e-06 | 0.005576738 | 5.455301 |

Figure 16: Top 10 genes from topTreat sorted by adjusted p value

From this list, the two gene symbols which were present as an intersection between the earlier top 10 genes produced by logFC and adjusted p value are present in this as well. This is strong evidence that genes with this gene symbol are highly significant and should be explored in depth for their biological importance. They are "Cxcl15" and "Them5".

References

- [1] Ryan M. Marquardt, Tae Hoon Kim, Jung-Yoon Yoo, Hanna E. Teasley, Asgerally T. Fazleabas, Steven L. Young, Bruce A. Lessey, Ripa Arora, and Jae-Wook Jeong. Endometrial epithelial ARID1A is critical for uterine gland function in early pregnancy establishment. *The FASEB Journal*, 35(2):e21209, 2021.