

Lab 3 - Differential Expression and Linear Modeling using limma

James Wettenhall. Created: June 18, 2004. Last modified: July 2, 2004.

Contents:

1. [Software and data required for this lab](#)
 1. [Required R packages](#)
 2. [Recommended R packages](#)
 3. [Required data](#)
2. [ApoAI data set](#)
 1. [Loading and normalizing the data](#)
 2. [Defining a design matrix](#)
 3. [Fitting a linear model](#)
 4. [Empirical Bayes statistics](#)
 5. [Displaying table\(s\) of differentially expressed genes](#)
 6. [An M A plot using coefficients fitted from the linear model](#)
3. [Estrogen data set](#)
 1. [Loading the \(normalized\) exprSet object](#)
 2. [Defining a design matrix](#)
 3. [Fitting a linear model](#)
 4. [Defining a contrasts matrix](#)
 5. [Fitting a linear model for the contrasts](#)
 6. [Empirical Bayes statistics](#)
 7. [Displaying table\(s\) of differentially expressed genes](#)
 8. [Linking the gene list to annotation information on the Internet](#)
 9. [M A plots using contrasts from the linear model](#)
 10. [Venn diagrams using contrasts from the linear model](#)
4. [Acknowledgements](#)
5. [References](#)
6. [Glossary](#)

1. Software required for this lab.

You will need R 1.9.0 (<http://www.R-Project.org>) for this lab exercise. This section lists all of the R packages you need to have installed and also lists some additional R packages which are recommended. Note that most of the R packages can be installed from the Bioconductor site automatically using:

```
source("http://www.bioconductor.org/getBioC.R")
getBioC()
```

but you are expected to use a more recent version of the `limma` package for this workshop, which is the reason for the alternate (non-Bioconductor) URL for the `limma` package below.

1.1 Required R packages

It is highly desirable to have these R packages in a directory in which you have write permission, especially for the `estrogen` package. You can use `.libPaths("C:/Custom/R/library/directory")` or `.libPaths("C:\\Custom\\R\\library\\directory")` before you run `install.packages()` (or equivalent) to install the package(s) in a customized directory location.

Package	URL
Biobase	http://www.bioconductor.org/repository/release1.4/package/html/Biobase.html
estrogen	http://www.bioconductor.org/data/experimental.html
hgu95av2cdf	http://www.bioconductor.org/data/metaData.html
hgu95av2	http://www.bioconductor.org/data/metaData.html
limma	http://bioinf.wehi.edu.au/limma
statmod	http://www.statsci.org/r/
xtable	http://cran.at.r-project.org/src/contrib/Descriptions/xtable.html

1.2 Recommended R packages

Package	URL
limmaGUI	http://bioinf.wehi.edu.au/limmaGUI
affyImGUI	http://bioinf.wehi.edu.au/affyImGUI
affy	http://www.bioconductor.org/repository/release1.4/package/html/affy.html
tkrplot	http://cran.at.r-project.org/src/contrib/Descriptions/tkrplot.html
R2HTML	http://cran.at.r-project.org/src/contrib/Descriptions/R2HTML.html
sma	http://cran.at.r-project.org/src/contrib/Descriptions/sma.html

1.3 Required data

As in Lab 2, the Estrogen data set is required, but this is available as an R package, so it is listed in [Section 1.1](#). An additional data set, ApoAI is required for this lab. It can be obtained from:

<http://bioinf.wehi.edu.au/marray/ibc2004/apoai.zip>.

2. ApoAI data set

In this section we consider a case study where two RNA sources are compared through a common reference RNA. The analysis of the log-ratios involves a two-sample comparison of means for each gene.

In this example we assume that the data is available as an RG list in the data file `ApoAI.RData`.

Background. The data is from a study of lipid metabolism by Callow et al (2000) ([\[1\]](#)). The apolipoprotein AI (ApoAI) gene is known to play a pivotal role in high density lipoprotein (HDL) metabolism. Mice which have the ApoAI gene knocked out have very low HDL cholesterol levels. The purpose of this experiment is to determine how ApoAI deficiency affects the action of other genes in the liver, with the idea that this will help determine the molecular pathways through which ApoAI operates.

Hybridizations. The experiment compared 8 ApoAI knockout mice with 8 wild type (normal) C57BL/6 ("black six") mice, the control mice. For each of these 16 mice, target mRNA was obtained from liver tissue and labelled using a Cy5 dye. The RNA from each mouse was hybridized to a separate microarray. Common reference RNA was labelled with Cy3 dye and used for all the arrays. The reference RNA was obtained by pooling RNA extracted from the 8 control mice.

Number of arrays	Red (Cy5)	Green (Cy3)
8	Wild Type "black six" mice (WT)	Pooled Reference (Ref)
8	ApoAI Knockout (KO)	Pooled Reference (Ref)

The experimental design could also be described in a diagram, such as:



This experiment is slightly old. Today it would be quite unusual to use this many replicate arrays without including any dye-swaps. In two color arrays, dye bias is still a significant problem. This is the main reason why normalization is always necessary.

This is an example of a single comparison experiment using a common reference. The fact that the comparison is made by way of a common reference rather than directly as for the swirl experiment makes this, for each gene, a two-sample rather than a single-sample setup.

2.1 Loading and normalizing the data

```

library(limma)
load("ApoAI.RData") # from:
http://bioinf.wehi.edu.au/marray/ibc2004/apoai.zip
objects()
names(RG)

MA <- normalizeWithinArrays(RG) # uses print-tip loess by default

```

2.2 Defining a design matrix

In order to construct a design matrix, let us remind ourselves of the linear model which we are fitting for each gene:

$$E(\mathbf{y}_g) = X\boldsymbol{\alpha}_g$$

where \mathbf{y}_g is the vector of normalized log ratios from the sixteen arrays, $E(\mathbf{y}_g)$ is the Expected Value of \mathbf{y}_g , X is the design matrix and $\boldsymbol{\alpha}_g$ is the vector of log ratios to estimate, corresponding to the "M" (fold change) column in the final list of differentially expressed genes given by `topTable`). The estimated log ratios are also known as "coefficients", "parameters" and "log fold changes".

This experiment has three types of RNA: Reference (**Ref**), Wild Type (**WT**), and Knockout (**KO**), so it is sufficient to estimate two log ratios in the linear model for each gene, i.e. we will estimate two parameters, so our design matrix should have two columns. In our case, the two parameters in the $\boldsymbol{\alpha}_g$ vector are the log ratios which compare gene expression levels in **WT vs Ref** and **KO vs WT**. (There are other possible parameterizations which could have been chosen instead. We are using one which allows us to estimate the contrast of interest (**KO vs WT**) directly from the linear model fit, rather than estimating it later as a contrast (i.e. a linear combination of parameters estimated from the linear model).

The design matrix we will use is:

$$X = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

where the first column is for the "**WT vs Ref**" parameter and the second column is for the "**KO vs WT**" parameter. The first 8 arrays hybridize **WT** RNA with **Ref** RNA so it makes sense that they each have a '1' in the **WT vs Ref** column. The last 8 arrays hybridize **KO** RNA with **Ref** RNA which corresponds to the sum of the two parameters, "**WT vs Ref**" and "**KO vs WT**" which is clear if you replace "**vs**" with a minus sign (remembering that everything has been log2 transformed so that subtraction here actually represents a log ratio).

This design matrix can be defined in R as follows:

```
design <- matrix(c(rep(1,16),rep(0,8),rep(1,8)),ncol=2)
colnames(design) <- c("WT-Ref","KO-WT")
design
```

2.3 Fitting a linear model

```
fit <- lmFit(MA,design=design)
names(fit)
```

2.4 Empirical Bayes statistics

```
fit <- eBayes(fit)
names(fit)
```

2.5 Displaying table(s) of differentially expressed genes

We now use the function `topTable` to obtain a list the genes with the most evidence of differential expression between the Knockout and Wild-Type RNA samples. The knockout gene (ApoAI) should theoretically have a log fold change of minus infinity, but microarrays cannot measure extremely large fold changes. While the M value of the ApoAI gene in the `topTable` may not have much biological meaning, the high ranking shows that this gene is consistently down-regulated across the replicate arrays.

```
topTable(fit,coef="KO-WT")

numGenes <- nrow(RG$genes)
completeTableKOvsWT <- topTable(fit,coef="KO-WT",number=numGenes)
write.table(completeTableKOvsWT,file="KOvsWTgenes.xls",sep="\t",quote=F
ALSE,col.names=NA)
```

The ".xls" extension is used so that the (tab-delimited text) file can be opened in Excel by double-clicking its icon.

The arguments of `topTable` can be studied in more detail with `?topTable` or `args(topTable)` or just `topTable`. Currently in `limma`, `topTable` is a high-level wrapper function which calls a lower level function "`toptable`". The default method for ranking genes is the B statistic (log odds of differential expression, Lonnstedt and Speed 2003 [2]), but the moderated t statistic and p-value can also be used. Using the average fold-change (the M column) is not recommended because this ignores the variability between replicate arrays.

2.6 An M A plot using coefficients fitted from the linear model

Using an M A plot, we can see which genes are selected as being differentially expressed by the B statistic (log odds of differential expression), which is the default ranking statistic for the `topTable`. Of course, the differentially expressed genes selected by the B statistic may not have the most extreme fold changes (M values), because some of the genes with extreme average fold changes may vary significantly between replicate arrays so they will be down-weighted by the empirical Bayes statistics.

```
M <- fit$coefficients[,"KO-WT"]
A <- fit$Amean
ord <- order(fit$lods[,"KO-WT"],decreasing=TRUE)
top10 <- ord[1:10]
plot(A,M,pch=16,cex=0.1)
text(A[top10],M[top10],labels=substring(fit$genes[top10,"NAME"],1,5),ce
x=0.8,col="blue")
```

3. Estrogen data set

This experiment is described in Lab 2. Only the RNA Targets file will be reviewed here.

3.1 Loading the (normalized) `exprSet` object

You should have an `.RData` file saved from Lab 2 including an `exprSet` object for the normalized Estrogen data. Change to the correct directory, using the R function `setwd("[directory]")`, e.g. `setwd(system.file("data", package="estrogen"))` and then load the `.RData` file with:

```
load("estrogen.RData")
```

Now check the data objects in your workspace. Hopefully you will find an `exprSet` object called `"eset"` which you created in Lab 2 using `rma`.

```
objects()
```

In order to use the `exprSet` methods such as `pData`, you will need to load the `Biobase` package. The main focus of this lab is on differential expression and linear models for microarrays, so you will also need to load the `limma` package.

```
library(Biobase)
library(limma)
```

3.2 Defining a design matrix

Defining a design matrix is quite simple for Affymetrix data because each microarray chip has only a single channel, as distinct from the two-color cDNA microarrays slides. Recall that we used an RNA Targets file (stored in tab-delimited text) to describe the Estrogen experiment:

Name	FileName	Target
Abs10.1	low10-1.cel	EstAbsent10
Abs10.2	low10-2.cel	EstAbsent10
Pres10.1	high10-1.cel	EstPresent10
Pres10.2	high10-2.cel	EstPresent10
Abs48.1	low48-1.cel	EstAbsent48
Abs48.2	low48-2.cel	EstAbsent48
Pres48.1	high48-1.cel	EstPresent48
Pres48.2	high48-2.cel	EstPresent48

After reading this Targets file into R with `read.phenoData`, we then included it in our `rawAffyData` object (of class `AffyBatch`), by using the `phenoData` argument of the `ReadAffy` function. Then when we created an object of class `exprSet` with `rma`, the phenotype data (i.e. the RNA Targets table) was automatically included in our `eset`

object (of class `exprSet`), and it can be recovered using either the `pData` method of the `exprSet` class.

```
targets <- pData(eset)
```

We have four pairs of replicate arrays (each of which is a single-channel array), so we should estimate four parameters in the linear model. The design matrix should have four columns and the rows should describe the four pairs of replicate arrays.

$$X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where the four columns of the matrix correspond to `EstAbsent10`, `EstPresent10`, `EstAbsent48` and `EstPresent48`.

This matrix can be defined in R as follows:

```
design <- model.matrix(~ -1 +  
factor(targets$Target, levels=unique(targets$Target)))  
colnames(design) <- unique(targets$Target)  
numParameters <- ncol(design)  
parameterNames <- colnames(design)  
design
```

3.3 Fitting a linear model

Now that we have defined our design matrix, fitting a linear model is as simple as:

```
fit <- lmFit(eset, design=design)
```

`fit` is an object of class `MArrayLM`. Its components can be accessed with the `$` operator just a standard list in R.

```
names(fit)
```

3.4 Defining a contrasts matrix

Contrasts are linear combinations of parameters from the linear model fit.

$$\beta_g = C^T \alpha_g$$

where β_g is a vector of contrasts for gene g , C is the contrasts matrix, and α_g is a vector of coefficients (estimated log fold changes), obtained from a linear model fit.

We will estimate three contrasts (so our contrasts matrix will have three columns). The first contrast is an estrogen effect at time 10 hours, the second as an estrogen effect at time 48 hours and the third is a time effect in the absence of estrogen.

$$C = \begin{pmatrix} -1 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

```
contrastNames <- c(paste(parameterNames[2],parameterNames[1],sep="-"),
                  paste(parameterNames[4],parameterNames[3],sep="-"),
                  paste(parameterNames[3],parameterNames[1],sep="-"))
contrastsMatrix <- matrix(c(-1,1,0,0,0,0,-1,1,-
1,0,1,0),nrow=ncol(design))
rownames(contrastsMatrix) <- parameterNames
colnames(contrastsMatrix) <- contrastNames
contrastsMatrix
```

3.5 Fitting a linear model for the contrasts

```
fit2 <- contrasts.fit(fit,contrasts=contrastsMatrix)
names(fit2)
```

3.6 Empirical Bayes statistics

```
fit2 <- eBayes(fit2)
names(fit2)
```

3.7 Displaying table(s) of differentially expressed genes

We now use the function `topTable` to obtain a list genes differentially expressed between Estrogen-Present and Estrogen-Absent at time 10 hours, followed by a list of genes differentially expressed between Estrogen-Present and Estrogen-Absent at time 48 hours.

```
topTable(fit2,coef="EstPresent10-EstAbsent10")
topTable(fit2,coef="EstPresent48-EstAbsent48")
```

```

numGenes <- nrow(eset@exprs)

completeTableEst10 <- topTable(fit2,coef="EstPresent10-
EstAbsent10",number=numGenes)
write.table(completeTableEst10,file="Est10genes.xls",sep="\t",quote=FAL
SE,col.names=NA)

completeTableEst48 <- topTable(fit2,coef="EstPresent48-
EstAbsent48",number=numGenes)
write.table(completeTableEst48,file="Est48genes.xls",sep="\t",quote=FAL
SE,col.names=NA)

```

3.8 Linking the gene list to annotation information on the Internet

If the genes in the topTable have standard IDs (e.g. UniGene or GenBank), then they can be linked with external annotation information on the Internet. Load the annotation package `hgu95av2`, which can be obtained from <http://www.bioconductor.org/data/metaData.html>.

```

library(hgu95av2cdf)
library(hgu95av2)

```

Now we obtain:

- the gene (probe-set) IDs (from the `AffyBatch` object, `ab`),
- the gene symbols (from the `hgu95av2SYMBOL` environment in the `hgu95av2` annotation package),
- the gene names (from the `hgu95av2GENENAME` environment in the `hgu95av2` annotation package), and
- the UniGene IDs (from the `hgu95av2UNIGENE` environment in the `hgu95av2` annotation package).

```

geneIDs <- ls(hgu95av2cdf)
geneSymbols <- unlist(as.list(hgu95av2SYMBOL))
geneNames <- unlist(as.list(hgu95av2GENENAME))
geneNames <- substring(geneNames,1,40)
unigene <- unlist(as.list(hgu95av2UNIGENE))
unigene <- gsub("Hs\\.","",unigene)
genelist <-
data.frame(GeneID=geneIDs, GeneSymbol=geneSymbols, GeneName=geneNames,
  UniGeneHsID=paste("<a
href=http://www.ncbi.nlm.nih.gov/UniGene/clust.cgi?ORG=Hs&CID=",
  unigene,">",unigene,"</a>",sep=""))

```

Now we recreate the toptable for the two contrasts considered earlier, "EstPresent10-EstAbsent10" and "EstPresent48-EstAbsent48" this time providing a hyperlink to the UniGene website for each gene in the toptable.

```

unigeneTopTableEst10 <- topTable(coef="EstPresent10-
EstAbsent10",n=20,fit=fit2,genelist=genelist)
unigeneTopTableEst48 <- topTable(coef="EstPresent48-
EstAbsent48",n=20,fit=fit2,genelist=genelist)

library(xtable)
xtableUnigeneEst10 <-
xtable(unigeneTopTableEst10,display=c("d","s","s","s","s","g","g","g","
e","g"))
xtableUnigeneEst48 <-
xtable(unigeneTopTableEst48,display=c("d","s","s","s","s","g","g","g","
e","g"))

cat(file="estrogenUniGeneE10.html", "<html>\n<body>")
print.xtable(xtableUnigeneEst10,type="html",file="estrogenUniGeneE10.ht
ml",append=TRUE)
cat(file="estrogenUniGeneE10.html", "</body>\n</html>",append=TRUE)

cat(file="estrogenUniGeneE48.html", "<html>\n<body>")
print.xtable(xtableUnigeneEst48,type="html",file="estrogenUniGeneE48.ht
ml",append=TRUE)
cat(file="estrogenUniGeneE48.html", "</body>\n</html>",append=TRUE)

```

The display argument to the `xtable` function is used to specify the format of text or numbers displayed in cells of the HTML table:

Format code	Meaning
d	Decimal (base ten) integer, e.g. 48
s	Character string, e.g. "Block"
g	General real floating-point number, e.g. 8.25
e	Floating point number in exponent format, e.g. 1.02E-05</td>

3.9 M A plots using contrasts from the linear model

Now we plot an M vs A plot for each contrast. Recall that M is a log fold change in base 2 and A is a log intensity in base 2 (in this case for each gene, we will use the average of all of the A values). The `geneSymbols` vector should be available from [Section 3.8](#). If you could not complete Section 3.8 because you don't have the `hgu95av2` annotation package installed, then you can just use the `geneIDs` vector also described in [Section 3.8](#), which does not require the annotation package.

```

M.Est10 <- fit2$coefficients[,"EstPresent10-EstAbsent10"]
A <- fit2$Amean
ord.Est10 <- order(fit2$lods[,"EstPresent10-
EstAbsent10"],decreasing=TRUE)
top30.Est10 <- ord.Est10[1:30]
plot(A,M.Est10,pch=16,cex=0.1)
text(A[top30.Est10],M.Est10[top30.Est10],labels=geneSymbols[top30.Est10
],cex=0.8,col="blue")

```

```

M.Est48 <- fit2$coefficients[,"EstPresent48-EstAbsent48"]
ord.Est48 <- order(fit2$lods[,"EstPresent48-
EstAbsent48"],decreasing=TRUE)
top30.Est48 <- ord.Est48[1:30]
plot(A,M.Est48,pch=16,cex=0.1)
text(A[top30.Est48],M.Est48[top30.Est48],labels=geneSymbols[top30.Est48
],cex=0.8,col="blue")

M.EstAbs48vs10 <- fit2$coefficients[,"EstAbsent48-EstAbsent10"]
ord.EstAbs48vs10 <- order(fit2$lods[,"EstAbsent48-
EstAbsent10"],decreasing=TRUE)
top30.EstAbs48vs10 <- ord.EstAbs48vs10[1:30]
plot(A,M.EstAbs48vs10,pch=16,cex=0.1)
text(A[top30.EstAbs48vs10],M.EstAbs48vs10[top30.EstAbs48vs10],labels=ge
neSymbols[top30.EstAbs48vs10],cex=0.8,col="blue")

```

3.10 Venn diagrams using contrasts from the linear model

We can now examine which genes respond to Estrogen at either time (10 hours or 48 hours) by examining moderated F-statistics on 2 degrees of freedom and p-values calculated from these F-statistics:

```

F.stat <- fit2$F
p.value <- fit2$F.p.value

```

What p-value cutoff should be used? One guide is to examine control probe-clusters which are known not to be differentially expressed.

```

i <- grep("AFFX",geneNames(eset))
summary(p.value[i])

```

We will choose a p-value of $1.0E-5$ which is well below the average p.value for the control probe-clusters. Consider those genes with moderated F-statistics with p-values below $1.0E-5$, and classify these according to whether they are significantly up or down regulated at the early or late times:

```

results <- classifyTestsF(fit2, p.value=1.0E-5)
vennDiagram(results,names=c("Est10PresAbs","Est48PresAbs","EstAbs48vs10
"))

```

Use the `table` command (below) to determine how many genes were up-regulated at 10 hours, and how many of those genes were still up-regulated at 48 hours. The same question can be answered for down-regulated genes, using the same table.

```

table(E10=results[,1],E48=results[,2])

```

4. Acknowledgements

Thanks to Scholtens et al. [3] for providing the estrogen data on the Bioconductor site (<http://www.bioconductor.org/data/experimental.html>). Thanks to Yee Hwa Yang and Sandrine Dudoit for the ApoAI data.

Thanks also to Gordon Smyth for the `limma` usersguide which this lab was based on.

5. References

1. Callow, M. J., Dudoit, S., Gong, E. L., Speed, T. P., and Rubin, E. M. (2000). Microarray expression profiling identifies genes with altered expression in HDL deficient mice. *Genome Research* **10**, 2022-2029. (<http://www.genome.org/cgi/content/full/10/12/2022>)
2. Lonnstedt I, Speed T. Replicated microarray data. STAT SINICA. Volume 12. Pages 31-46. Jan 2002
3. Scholtens D, Miron A, Merchant FM, Miller A, Miron PL, Iglehart JD, Gentleman R. Analyzing Factorial Designed Microarray Experiments. *Journal of Multivariate Analysis*. To appear.
4. Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology* **3**, No. 1, Article 3. (<http://www.bepress.com/sagmb/vol3/iss1/art3/>)

6. Glossary

Knockout RNA	RNA extracted from a biological specimen which has had one gene artificially knocked out (removed) from it in a laboratory.
Wild Type RNA	RNA extracted from a biological specimen whose genes are in their natural form (as found in the wild).