Using `limma` to analyze microarray and RNA-Seq data

Humberto Ortiz-Zuazaga

March 7, 2013

1 Introduction

Bioconductor [3] is a set of R packages for analysis of biological data, with an emphasis on microarray and other high-throughput datasets.

The bioconductor web site has instructions to install R and bioconductor. I also highly recommend installing R Studio, a graphical environment for running R, that groups together a source editor, help, figures, data browsers, and many other tools. R Studio is available on it's own website.

This example will use standard `affy` [2] and `limma` [4] commands to analyze example datasets. Bioconductor has extensive help, which you can access in many ways. One simple way is to type `?foo` where you want help on the object called "foo" (or `??foo` to search for relevant topics). You can open an interactive browser interface to the help system by typing `help.start()`. In the browser, you can look at the documentation for the installed packages to find help on `limma` and `affy`.

The `limma` package, in particular, has an extensive 120 page user's guide, with many examples, including around 8 full case studies of high-throughput data analysis. You can read the guide using the `limmaUsersGuide()` command.

```
> library(limma)
> library(affy)
```

2 Microarrays

2.1 Reading the data

Both `limma` and `affy` have many functions to simplify import of microarray data into the system. See the documentation for `readTargets` and `ReadAffy` for some examples.

In this document, we will instead use sample data that has already been read into the appropriate R objects.

Affydata is one such example data set.

```
> require(affydata)

     Package
[1,] "affydata"
     LibPath
[1,] "/Library/Frameworks/R.framework/Versions/2.15/Resources/library"
     Item       Title
[1,] "Dilution" "AffyBatch instance Dilution"

> data(Dilution)
> Dilution

AffyBatch object
size of arrays=640x640 features (35221 kb)
cdf=HG_U95Av2 (12625 affyids)
number of samples=4
number of genes=12625
annotation=hgu95av2
notes=
```

The Dilution affybatch contains four samples, two each of 20μg and 10μg liver tissue from human subjects, read on two different scanners (A and B).

```
> phenoData(Dilution)

An object of class 'AnnotatedDataFrame'
  sampleNames: 20A 20B 10A 10B
  varLabels: liver sn19 scanner
  varMetadata: labelDescription
```

Examine MA plot for raw data, comparing the 20μg samples on scanner A and B.

```
> plotMA(exprs(Dilution)[,c(1,2)])
```

`Dilution` will contain the AffyBatch, with the raw expression values for each probe in each sample, with additional information on the probes and samples.

2.2 Normalization and pre-processing

We can use the `rma` command to normalize and summarize the probes for each feature. Prior to the summarization, each feature is represented with four probes. After the normalization and summarization routine, we have a single expression value for each feature in each sample.

```
> probeNames(Dilution)[1:20]

 [1] "100_g_at" "100_g_at" "100_g_at" "100_g_at" "100_g_at" "100_g_at"
 [7] "100_g_at" "100_g_at" "100_g_at" "100_g_at" "100_g_at" "100_g_at"
[13] "100_g_at" "100_g_at" "100_g_at" "100_g_at" "1000_at"  "1000_at"
[19] "1000_at"  "1000_at"
```

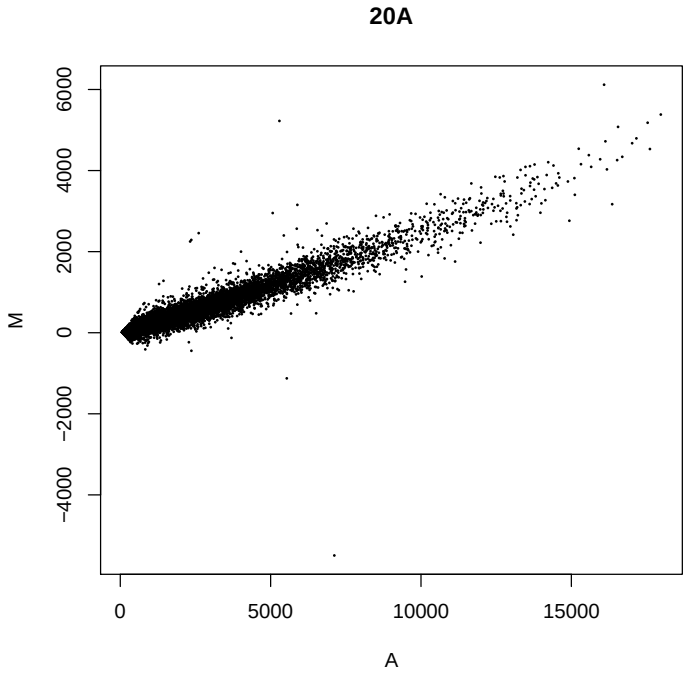


Figure 1: MA plot before normalization

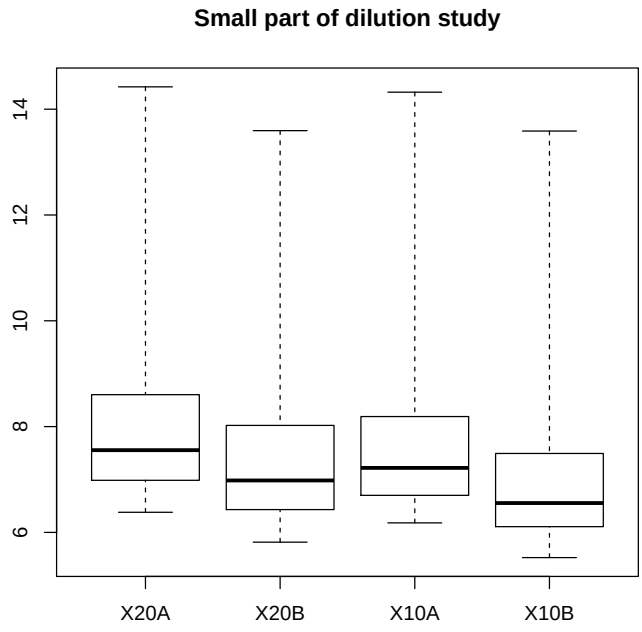


Figure 2: Box plot before normalization

```
> eset <- rma(Dilution)

Background correcting
Normalizing
Calculating Expression

> featureNames(eset)[1:10]

 [1] "100_g_at"  "1000_at"   "1001_at"   "1002_f_at" "1003_s_at" "1004_at"
 [7] "1005_at"   "1006_at"   "1007_s_at" "1008_f_at"
```

A boxplot shows the distribution of expression values before (Figure 2) and after (Figure 3) the normalization.

```
> boxplot(Dilution)


> boxplot(exprs(eset))
```

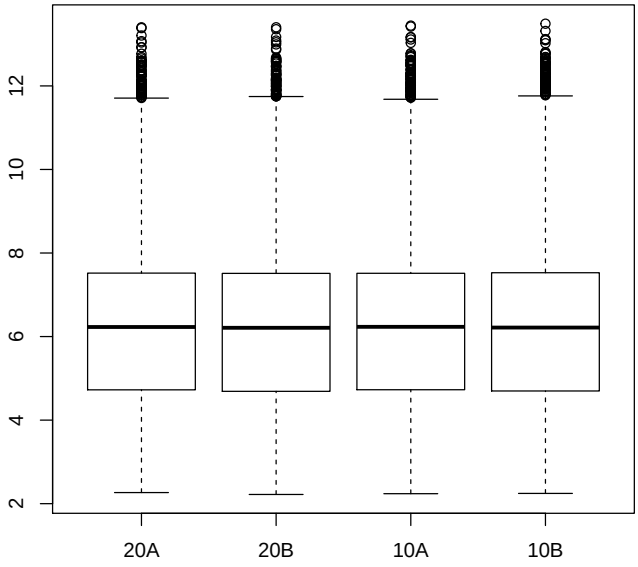


Figure 3: Box plot after normalization

2.3 Experimental design

The experiment has a simple design, each sample is labeled in the targets file with the target it was hybridized with. This information can be used to constuct a design matrix that identifies each group.

```
> f <- factor(c("C20","C20","C10","C10"))
> design <- model.matrix(~0+f)
> colnames(design) <- c("C20", "C10")
> design
```

```
  C20 C10
1   0   1
2   0   1
3   1   0
4   1   0
attr(,"assign")
[1] 1 1
attr(,"contrasts")
attr(,"contrasts")$f
[1] "contr.treatment"
```

We can fit a model that has a mean for each group, and test if the group means are different. The `eBayes` function computes an empirical Bayes factor, pooling the variances from all the genes to estimate significance.

```
> cont.matrix <- makeContrasts(conc=C20-C10, levels=design)
> cont.matrix
```

```
      Contrasts
Levels conc
   C20    1
   C10   -1
```

```
> fit <- lmFit(eset, design)
> fit2 <- contrasts.fit(fit, cont.matrix)
> fit.b <- eBayes(fit2)
```

2.4 Reporting the results

We now have a model fit that estimates the log ratios between the positive and negative samples. An MA plot (Figure 4) summarizes the fit. The y axis plots M, the log ratio of expression in the positive and negative coefficients. The x axis plots the A, or average log intensity of each gene.

```
> plotMA(fit.b)
```

The fit also has an estimate of the Bayes factor, the log odds of differential expression for each gene. A plot of the B vs log ratios is called a volcanoplot (see Figure 5).

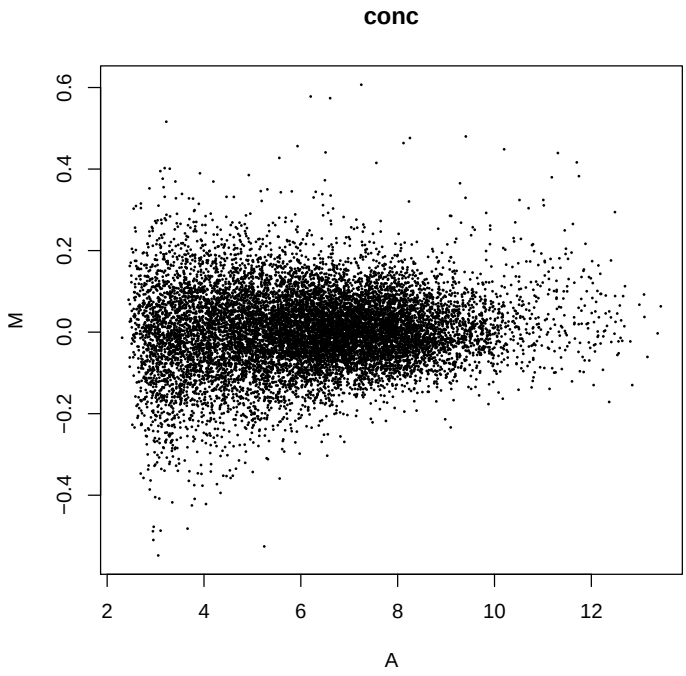


Figure 4: MA plot

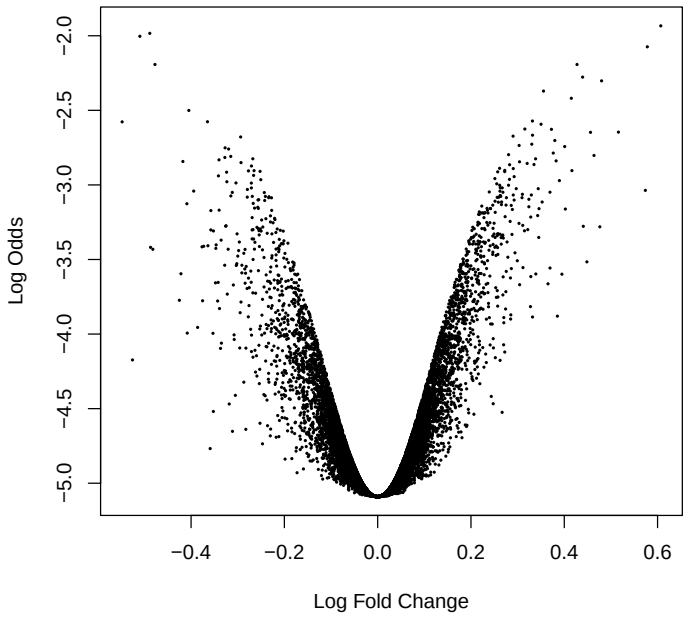


Figure 5: Volcano plot

```
> volcanoplot(fit.b)
```

Another way to report the results is exporting a table with the most significant features. Estimated p-values using a number of multiple testing corrections can be computed, in this case we use the Benjamini & Hochberg correction. [1]

```
> topTable(fit.b, adjust="BH")
```

	ID	logFC	AveExpr	t	P.Value
12579	AFFX-DapX-M_at	0.6068933	7.248605	8.046185	0.0001351863
4143	34103_at	-0.4884786	2.945512	-7.624196	0.0001858859
1657	31642_at	-0.5097732	2.954299	-7.472093	0.0002092465
12607	AFFX-M27830_5_at	0.5780179	6.203493	6.982726	0.0003104979
2962	32934_i_at	-0.4774220	2.961046	-6.301995	0.0005585254
12599	AFFX-HUMRGE/M10098_M_at	0.4273471	5.555700	6.301609	0.0005587188
10990	40886_at	0.4393364	11.308830	5.895455	0.0008118744
12561	AFFX-BioB-5_at	0.4799698	9.407338	5.785891	0.0009008883
8993	38907_at	0.3554734	3.165857	5.503947	0.0011852670
9483	39393_r_at	0.4151446	7.560304	5.320512	0.0014244158

```
      adj.P.Val         B
12579 0.8805792 -1.933454
4143  0.8805792 -1.983803
1657  0.8805792 -2.003599
12607 0.9800091 -2.074075
2962  0.9984512 -2.192581
12599 0.9984512 -2.192656
10990 0.9984512 -2.277300
12561 0.9984512 -2.302214
8993  0.9984512 -2.370834
9483  0.9984512 -2.419237
```

3 RNA-Seq

3.1 Introduction

Many gene expression studies are turning to RNA-Seq, a form of second-generation sequencing that sequences products derived from mRNA. RNA-Seq count data can be analyzed with the same tools we have learned to use for microarrays. Obtaining count data (also known as digital gene expression data) from raw RNA-Seq reads is beyond the scope of this guide, but tools such as galaxy [5], bowtie [6], and trinity [7] can be used to map reads to genes and tally the counts.

3.2 RNA-Seq packages

```
> library(Biobase)
> library(biomaRt)
> library(edgeR)
```

3.3 Load ReCount data

ReCount [8] is an online database of RNA-seq data from 18 experiments. These experiments have already been read into R and are published on the web. R can load prepared datasets directly from the web:

```
> gilad <- load(
+   url("http://bowtie-bio.sourceforge.net/recount/ExpressionSets/gilad_eset.RData"))
> gilad
```

```
[1] "gilad.eset"
```

The data is already an Expression Set, with raw counts for each probe, and information on the samples. This data comes from a published study of gene expression in liver in males and females of different species [9]. We will use data from 3 males and 3 females from the human species.

```
> gilad.eset
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 52580 features, 6 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: SRX014818and9 SRX014820and1 ... SRX014828and9 (6 total)
  varLabels: sample.id num.tech.reps gender
  varMetadata: labelDescription
featureData
  featureNames: ENSG00000000003 ENSG00000000005 ... LRG_99 (52580
    total)
  fvarLabels: gene
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

```
> phenoData(gilad.eset)$gender
```

```
[1] F F F M M M
Levels: F M
```

```
> exprs(gilad.eset)[1:5,]
```

```
                SRX014818and9 SRX014820and1 SRX014822and3 SRX014824and5
ENSG00000000003            60            60            16             9
ENSG00000000005             0             0             0             0
ENSG00000000419            25             9            15            15
ENSG00000000457            32            19            21            31
ENSG00000000460             1             3             0             5
                SRX014826and7 SRX014828and9
ENSG00000000003            56            37
ENSG00000000005             0             0
ENSG00000000419            26            11
ENSG00000000457            28            28
ENSG00000000460             1             1
```

3.4 Removing genes that are not expressed

Many genes are not present in any sample, it is simpler to remove these before continuing.

```
> isexpr <- rowSums(cpm(exprs(gilad.eset))>1) >= 3
> sum(isexpr)
```

```
[1] 8069
```

```
> gilad.isexpr <- gilad.eset[isexpr,]
```

3.5 Normalize the counts

Limma provides a routine, `voom`, designed to normalize digital gene expression data.

```
> nf <- calcNormFactors(gilad.isexpr)
> groups <- phenoData(gilad.isexpr)$gender
> design <- model.matrix(~ groups)
> y <- voom(exprs(gilad.isexpr),design,
+           lib.size=colSums(exprs(gilad.isexpr))*nf,
+           normalize.method="quantile")
```

3.6 Build a linear model

Once we have log-normalized counts, we can proceed to construct a linear model, using the same tools we used for one-color (Affymetrix) arrays.

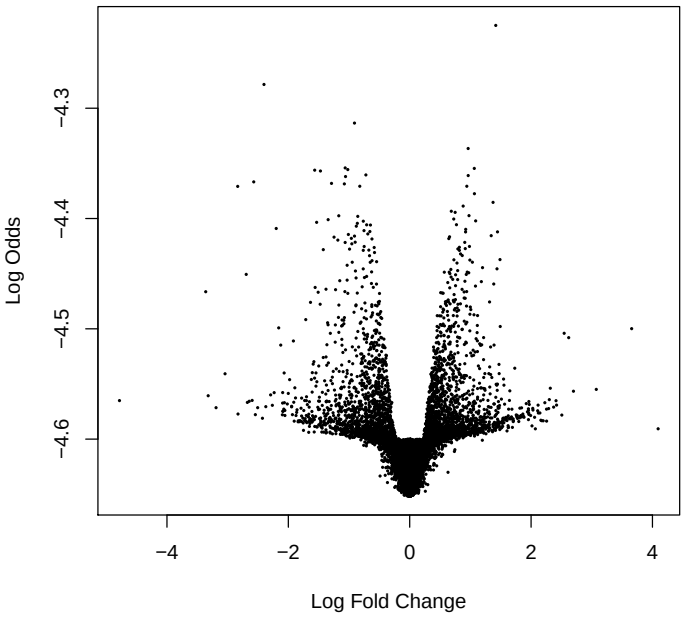
```
> fit <- lmFit(y,design)
> fit <- eBayes(fit)
> topTable(fit, coef=2)
```

```
                  ID      logFC  AveExpr         t      P.Value adj.P.Val
286  ENSG00000049239  1.4179955 8.353328  4.932432 0.0006353586 0.9932438
1774 ENSG00000110244 -2.4011726 6.693654 -4.493731 0.0012221106 0.9932438
5957 ENSG00000174718 -0.9099697 7.477112 -3.614733 0.0049090586 0.9932438
191  ENSG00000023330  0.9631780 7.737762  3.342743 0.0076923282 0.9932438
6993 ENSG00000187837 -1.0631848 6.102709 -3.413789 0.0068359089 0.9932438
7743 ENSG00000214456  1.0644053 6.614080  3.275959 0.0085985800 0.9932438
4913 ENSG00000164626 -1.0201108 6.587511 -3.254500 0.0089125858 0.9932438
5654 ENSG00000171051 -1.5657336 5.564722 -3.758463 0.0038844280 0.9932438
3045 ENSG00000133392 -1.4710554 7.721775 -3.134114 0.0109064995 0.9932438
4827 ENSG00000163513 -0.7220998 7.084751 -3.117506 0.0112154685 0.9932438
            B
286  -4.224938
1774 -4.278461
5957 -4.313473
191  -4.336530
6993 -4.354120
7743 -4.354584
4913 -4.355630
5654 -4.356174
3045 -4.356915
4827 -4.360486
```

3.7 Volcanoplot model fit

The fit can be summarized using the same reporting tools as before.

```
> volcanoplot(fit,coef=2)
```



References

[1] Benjamini, Y., and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society Series B,* 57, 289–300.

[2] Gautier, L., Cope, L., Bolstad, B. M., and Irizarry, R. A. (2004). affy—analysis of Affymetrix GeneChip data at the probe level. *Bioinformatics* 20, 3 (Feb. 2004), 307-315.

[3] R. Gentleman, V. J. Carey, D. M. Bates, B.Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, and others Bioconductor: Open software development for computational biology and bioinformatics (2004). *Genome Biology*, Vol. 5, R80

[4] Smyth, G. K. (2005). Limma: linear models for microarray data. In: 'Bioinformatics and Computational Biology Solutions using R and Bioconductor'. R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds), Springer, New York, pages 397–420.

[5] `http://galaxyproject.org/`

[6] http://bowtie-bio.sourceforge.net/index.shtml

[7] http://trinityrnaseq.sourceforge.net/

[8] Frazee AC, Langmead B, Leek JT. ReCount: a multi-experiment resource of analysis-ready RNA-seq gene count datasets. BMC Bioinformatics 12:449.

[9] Marioni JC, Mason CE, Mane SM, Stephens M, Gilad Y. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. Genome Res. 2008 Sep;18(9):1509-17.