R and Bioconductor for the Analysis of Massive Genomic Data

Niccolò Bassani, Federico Ambrogi, Elia Biganzoli

Department of Clinical Sciences and Community, University of Milan

2nd MilanoR meeting, 27/09/2012
Outline

1. Introduction
2. Bioconductor
3. Practical applications
   - Illumina BeadArray
   - SAGE data
"Genomics is a discipline in genetics concerned with the study of the genomes of organisms. The field includes efforts to determine the entire DNA sequence of organisms and fine-scale genetic mapping. [...]"

For the United States Environmental Protection Agency, "the term "genomics" encompasses a broader scope of scientific inquiry associated technologies than when genomics was initially considered. A genome is the sum total of all an individual organism’s genes. Thus, genomics is the study of all the genes of a cell, or tissue, at the DNA (genotype), mRNA (transcriptome), or protein (proteome) levels.”

[Source: Wikipedia]
## Some types of genomic platforms

<table>
<thead>
<tr>
<th>Technology</th>
<th>What measures</th>
<th>Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT-qPCR (1992)</td>
<td>number of cycles at which the transcript exceeds a threshold (above background)</td>
<td>( p &gt; n )</td>
</tr>
<tr>
<td>Microarrays (1995)</td>
<td>relative intensity of fluorescence of hybridized transcripts</td>
<td>( p \gg n )</td>
</tr>
<tr>
<td>NGS (2008)</td>
<td>number of copies of a sequence of nucleotides (e.g. ATTCTTC) in a region of the genome</td>
<td>( p \gg n )</td>
</tr>
</tbody>
</table>

**Table:** Technologies for genomic experiments
(Some) issues in massive genomic analysis

- large (hundreds or thousands) number of variables
- small (tens, or even less) sample size
- noisy data
- discriminating between technical and biological effects
- difficulties in comparing results across experiments
- ...and many many others (platform specific)
Possible goals of genomic experiments

- **class discovery**: are there in the data groups of samples characterized by similar expression profiles? (e.g. cluster analysis, PCA)

- **class prediction**: according to some labelling of samples (e.g. clinical status: healthy/tumor), can we predict the status of subjects by exploiting information on the expression profiles? (e.g. Support Vector Machines)

- **class comparison**: are there any differences in the expression profiles between somehow labelled groups of samples? (e.g. *limma* approach)
What does R offer to analyze genomic data?
The Bioconductor project

Method

**Bioconductor: open software development for computational biology and bioinformatics**

Robert C Gentleman\(^1\), Vincent J Carey\(^2\), Douglas M Bates\(^3\), Ben Bolstad\(^4\), Marcel Dettling\(^5\), Sandrine Dudoit\(^4\), Byron Ellis\(^6\), Laurent Gautier\(^7\), Yongchao Ge\(^8\), Jeff Gentry\(^4\), Kurt Hornik\(^9\), Torsten Hothorn\(^10\), Wolfgang Huber\(^11\), Stefano Iacus\(^12\), Rafael Irizarry\(^13\), Friedrich Leisch\(^9\), Cheng Li\(^14\), Martin Maechler\(^5\), Anthony J Rossini\(^14\), Gunther Sawitzki\(^15\), Colin Smith\(^16\), Gordon Smyth\(^17\), Luke Tierney\(^18\), Jean YH Yang\(^19\) and Jianhua Zhang\(^1\)

Addresses: \(^1\)Department of Biostatistical Science, Dana-Farber Cancer Institute, 44 Binney St, Boston, MA 02115, USA. \(^2\)Channing Laboratory, Brigham and Women’s Hospital, 75 Francis Street, Boston, MA 02115, USA. \(^3\)Department of Statistics, University of Wisconsin-Madison, 1210 W Dayton St, Madison, WI 53706, USA. \(^4\)Division of Biostatistics, University of California, Berkeley, 140 Warren Hall, Berkeley, CA 94720-3880, USA. \(^5\)Computational Biology and Statistical Computing, ETH Zurich, Zurich, CH 8092, Switzerland. \(^6\)Department of Statistics, Howard University, USA

- based on the R programming language
- two releases each year (actual version: 2.10, fully compatible with R 2.15.0)
- 554 software packages (i.e. packages containing functions for computations and databases search) and more than 600 annotation packages (i.e. packages containing relevant biological information)
- ...of course, OPEN SOURCE!
Installation of Bioconductor

- install the latest R version
- in the console type the following (be sure to have internet connection):
  
  ```
  source("http://www.bioconductor.org/biocLite.R")
  biocLite()
  ```

  the following packages will be installed (basic installation):
  Biobase, IRanges, AnnotationDbi

- to install specific packages type
  
  ```
  biocLite("<name-of-package>")
  ```
Bioconductor resources...continued

- ML for users (https://stat.ethz.ch/mailman/listinfo/bioconductor) and developers (https://stat.ethz.ch/mailman/listinfo/bioc-devel), just like R Project

- package vignettes accessible by typing the following in the R console:
  
  browseVignettes(package = "<name-of-package>")

- HELP section of Bioconductor Homepage 😊
Two applications: microarray and SAGE data

- microarray analysis of the role of a peptide in fetal loss in mice
  - Illumina BeadArray MouseWG-6_V2 (microarray platform)
  - 12 samples and 4 groups
  - goal of the study: differential gene expression between treatment groups

- Serial Analysis of Gene Expression (SAGE) of Pulmonary Arterial Hypertension (PAH) samples
  - methodology similar to RNASEq
  - 23 samples (divided in 4 groups)
  - goal of the study: differential abundance between different types of PAH
Data import: the *lumibatch* class

```r
> setwd("/Users/Nico82/Documents/Lavoro/milanoR/dati/")
> dati <- lumiR("SampleProbeProfile2.txt",
                  columnNameGrepPattern = list(exprs = "AVG_signal",
                                             se.exprs = "BEAD_STERR", beadNum = NA, detection = "Detection Pval"))
> dati

ExpressionSet (storageMode: lockedEnvironment)
assayData: 45281 features, 12 samples
  element names: detection, exprs
protocolData: none
phenoData
  sampleNames: 4305961004_A 4305961004_B ... 4305961004_F (12 total)
  varLabels: sampleID
  varMetadata: labelDescription
featureData
  featureNames: 0_fG05_IoFC7_KUL14 T.kvUVfFyruaCeqZVs ... Zl14J4B0PsjTJ1N19IM (45281 total)
  fvarLabels: ProbeID TargetID ... DEFINITION (9 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

Bassani NP *et al.*  
R and Bioconductor in genomics
Accessing the data: using `exprs`

```r
> options(width = 80)
> head(exprs(dati)[,1:8])

   4305961004_A 4305961004_B 4305961057_A 4305961057_B
0_fGO5_IoFC7_KULl4  47.98294  41.73666  49.98639  50.62171
   T.kvUVfFyruaCeqZVs  61.28226  53.21529  54.74257  52.80311
fN_XSH016BXqmiAmUc  56.33011  57.84283  73.86475  85.00059
Q96MoiqM_z49FU37tE  40.49831  37.14137  47.59000  50.06340
91ekf9_d_rteewJTpQ  43.56656  56.21564  48.86168  56.90842
KqvaKDqYkgAXyjiImk  80.33805  79.86263  62.52376  66.17236

   4305961004_C 4305961057_C 4305961057_D 4305961004_D
0_fGO5_IoFC7_KULl4  47.11564  42.59488  52.18167  47.56797
   T.kvUVfFyruaCeqZVs  37.88368  47.46000  43.64196  59.96165
fN_XSH016BXqmiAmUc  48.63729  90.68812  75.77998  58.95243
Q96MoiqM_z49FU37tE  41.97672  43.56746  46.68839  39.78867
91ekf9_d_rteewJTpQ  43.80513  40.51325  41.54160  41.09210
KqvaKDqYkgAXyjiImk  496.37640 224.52330 263.98510 551.57820
```
Lumibatch objects created with the \texttt{lumiR} function are S4 objects which extend a well-known class named \texttt{ExpressionSet}, which is the basis for creating and managing various types of expression data within Bioconductor. Within these Lumibatch object one can access different slots, containing various information on the experiment at hand.

For further information:
\texttt{?lumiR}

\texttt{?ExpressionSet}

http://www.stat.auckland.ac.nz/S-Workshop/Gentleman/S4Objects.pdf for some deepening on S4 objects
> par(mfrow=c(2,2))
> plot(dati,what = "density")
> plot(dati,what = "boxplot")
> plot(dati,what = "sampleRelation")
> plot(dati,what = "outlier")
Pre-processing: data filtering

```r
> detection.exprs = detection(dati)
> detection.exprs[1:5,1:4]

    4305961004_A 4305961004_B 4305961057_A 4305961057_B
T.kvUVfFyruaCeqZVs  0.028846150  0.08547009  0.15384620  0.189102600
fN_XSH016BXqmiAmUc  0.066239320  0.04380342  0.01709402  0.002136752
KqvaKDqYkgAXyjiImk  0.000000000  0.00000000  0.00000000  0.000000000
WpaZ9x9f_hAnoR.VBE  0.000000000  0.00000000  0.00000000  0.000000000
ZhdXp75JftSF3iWLF4  0.001068376  0.00000000  0.00000000  0.000000000

> detect = detectionCall(dati,Th = 0.05)
> head(detect)

              0_fGO5_IoFC7_KULl4  T.kvUVfFyruaCeqZVs fN_XSH016BXqmiAmUc  Q96MoigM_z49FU37tE
0_fGO5_IoFC7_KULl4           0          2          9          0
91ekf9_d_rteewJTpQvKqvaKDqYkgAXyjiImk
0_fGO5_IoFC7_KULl4           0         12

> dati = dati[detect > ncol(dati)*0.15,]
> dim(dati)

Features  Samples
      22777       12
```
Introduction

Bioconductor

Practical applications

Illumina BeadArray

SAGE data

Array Normalization

Normalization is the (crucial) process by which we try to minimize variability due to technical artifacts in order to maximize the relevant biological information in the data. In package lumi 6 different methods are implemented in the function lumiN: quantile, loess, vsn, rank invariant, rsn and ssn.

```r
> dati.T = lumiT(dati, method = "log2") # transform data to log2

> dati.qnorm = lumiN(dati.T, method = "quantile",verbose = FALSE)
> dati.loess = lumiN(dati.T, method = "loess",verbose = FALSE)
> dati.vsn = lumiN(dati, method = "vsn",
               verbose = FALSE) # requires non-transformed raw data
> dati.rankinvariant = lumiN(dati.T, method = "rankinvariant",
               verbose = FALSE)
> dati.rsn = lumiN(dati.T, method = "rsn",verbose = FALSE)
> dati.ssn = lumiN(dati.T, method = "ssn",verbose = FALSE)
```

Bassani NP et al. R and Bioconductor in genomics
> par(mfrow = c(2,3))
> meanSdPlot(dati.qnorm, main = "quantile")
> meanSdPlot(dati.loess, main = "loess")
> meanSdPlot(dati.vsn, main = "vsn")
> meanSdPlot(dati.rankinvariant, main = "Rank Invariant")
> meanSdPlot(dati.rsn, main = "rsn")
> meanSdPlot(dati.ssn, main = "ssn")
Normalization check (2)

```r
> par(mfrow = c(2,3))
> plot(dati.qnorm,what = "boxplot", main = "quantile")
> plot(dati.loess,what = "boxplot",main = "loess")
> plot(dati.vsn,what = "boxplot",main = "vsn")
> plot(dati.rankinvariant,what = "boxplot",main = "Rank Invariant")
> plot(dati.rsn, what = "boxplot",main = "rsn")
> plot(dati.ssn, what = "boxplot",main = "ssn")
```
Annotating the dataset (1)

The gene IDs we have in our data are called nuIDs (nucleotide universal identifier) need to be converted to something more informative.

```r
> library(lumiMouseIDMapping)
> RefSeq = nuID2RefSeqID(row.names(dati.qnorm),
   lib.mapping='lumiMouseIDMapping',returnAllInfo = TRUE )
> head(RefSeq)

       Accession      EntrezID Symbol
TdGred.oCNpxxeuf14  NM_029887    77254    Yif1b
xxvA7xwYBA530hx.yk NM_173750  212772 2700007P21Rik
QS6jj1c7eisgjr7r0k NM_173750  212772 2700007P21Rik
ohd5KfZFEl.0Z6kE5M NM_173750  212772 2700007P21Rik
KddlZ7dxU4KdgMC0Fc NM_023132   19703      Renbp
64iTfwidAkRB5EdEpU NM_028777   22877    Sec1411
```

```r
> dim(RefSeq)
[1] 60385    3
```

```r
> dim(dati.qnorm)
Features  Samples
  22777     12
```
Annotating the dataset (2)

```r
> Annot = RefSeq[row.names(RefSeq) %in% row.names(exprs(dati.qnorm))],
> head(Annot)

<table>
<thead>
<tr>
<th>Accession</th>
<th>EntrezID</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>TdGred.oCNpxxeuf14</td>
<td>77254</td>
<td>Yif1b</td>
</tr>
<tr>
<td>QS6jj1c7eisgjr7r0k</td>
<td>212772</td>
<td>2700007P21Rik</td>
</tr>
<tr>
<td>ohd5KfZFE1.0Z6kE5M</td>
<td>212772</td>
<td>2700007P21Rik</td>
</tr>
<tr>
<td>KddlZ7dxU4KdgMC0Fc</td>
<td>19703</td>
<td>Renbp</td>
</tr>
<tr>
<td>64iTfwidAkRB5EdEpU</td>
<td></td>
<td>Sec14l1</td>
</tr>
<tr>
<td>cp7VWLX1cIk9JQkoAc</td>
<td>208624</td>
<td>Alg3</td>
</tr>
</tbody>
</table>
```

```r
> dim(Annot)
[1] 22777  3
> dim(dati.qnorm)
Features Samples
22777  12
```

**IMPORTANT:** The non-unique values of Accession, EntrezID and Symbol mean that there are more "probes" mapping to the same gene.
Differential expression: the *limma* approach

The *limma* package implements a gene-by-gene linear model fitting, using an Empirical Bayes approach to estimate a moderated variance by "borrowing" information from groups of genes with similar levels of expression, in order to avoid spurious results.

```r
> library(limma)
> treatment = as.factor(rep.int(LETTERS[1:4],c(4,3,4,1)))
> design = model.matrix(~ -1 + treatment)
> fit = lmFit(dati.vsn,design)
> names(fit)

[1] "coefficients"    "rank"          "assign"        "qr"
[5] "df.residual"     "sigma"        "cov.coefficients" "stdev.unscaled"
[9] "pivot"           "genes"        "Amean"         "method"
[13] "design"
```
Estimating the contrasts

> fit$coefficients
> fit$coefficients[1:4,]  # means of expression within treatment

A   B   C   D
T.kvUVfFyruaCeqZVs 4.865204 4.661610 4.791456 4.687051
fN_XSH016BXqmiAmUc 5.367983 5.994914 5.283411 4.622358
KqvaKDqYkgAXyjiImk 9.254445 8.663788 8.517429 8.945982
WpaZ9x9f_hAnoR.VBE 10.287848 10.638507 10.650699 9.671832

> contrast.matrix = makeContrasts('C - A','C - B','B - A', levels = design)
> fit2 = contrasts.fit(fit, contrast.matrix)
> fit2$coefficients[1:4,]  # mean differences of expression between treatments

Contrasts

<table>
<thead>
<tr>
<th></th>
<th>C - A</th>
<th>C - B</th>
<th>B - A</th>
</tr>
</thead>
<tbody>
<tr>
<td>T.kvUVfFyruaCeqZVs</td>
<td>-0.07374797</td>
<td>0.12984615</td>
<td>-0.20359413</td>
</tr>
<tr>
<td>fN_XSH016BXqmiAmUc</td>
<td>-0.08457131</td>
<td>-0.71150209</td>
<td>0.62693077</td>
</tr>
<tr>
<td>KqvaKDqYkgAXyjiImk</td>
<td>-0.73701592</td>
<td>-0.14635872</td>
<td>-0.59065719</td>
</tr>
<tr>
<td>WpaZ9x9f_hAnoR.VBE</td>
<td>0.36285140</td>
<td>0.01219187</td>
<td>0.35065954</td>
</tr>
</tbody>
</table>
Assessing significance of coefficients

```r
> fit2 = eBayes(fit2)
> fit2$p.value[1:4,]

Contrasts

<table>
<thead>
<tr>
<th></th>
<th>C - A</th>
<th>C - B</th>
<th>B - A</th>
</tr>
</thead>
<tbody>
<tr>
<td>T.kvUVfFyruaCeqZVs</td>
<td>0.724167155</td>
<td>0.56662493</td>
<td>0.37314770</td>
</tr>
<tr>
<td>fN_XSH016BXqmiAmUc</td>
<td>0.803088423</td>
<td>0.06816999</td>
<td>0.10343331</td>
</tr>
<tr>
<td>KqvaKDqYkgAXyjiImk</td>
<td>0.001670785</td>
<td>0.48538696</td>
<td>0.01202689</td>
</tr>
<tr>
<td>WpaZ9x9f_hAnoR.VBE</td>
<td>0.142607404</td>
<td>0.96209942</td>
<td>0.18616680</td>
</tr>
</tbody>
</table>

> topTable(fit2,coef = 3, number = 5, adjust = "BH")[,c(2,10:15)]

<table>
<thead>
<tr>
<th>TargetID</th>
<th>logFC</th>
<th>AveExpr</th>
<th>t</th>
<th>P.Value</th>
<th>adj.P.Val</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>9261</td>
<td>-1.700224</td>
<td>7.652034</td>
<td>-7.359561</td>
<td>4.121594e-06</td>
<td>0.06284554</td>
<td>4.020218</td>
</tr>
<tr>
<td>14491</td>
<td>-1.265346</td>
<td>4.835079</td>
<td>-7.167309</td>
<td>5.518333e-06</td>
<td>0.06284554</td>
<td>3.794893</td>
</tr>
<tr>
<td>8145</td>
<td>-1.441038</td>
<td>7.266179</td>
<td>-6.465275</td>
<td>1.669820e-05</td>
<td>0.08158869</td>
<td>2.919401</td>
</tr>
<tr>
<td>18499</td>
<td>-1.511175</td>
<td>5.911325</td>
<td>-6.317407</td>
<td>2.126439e-05</td>
<td>0.08158869</td>
<td>2.724178</td>
</tr>
<tr>
<td>13206</td>
<td>-1.647630</td>
<td>8.036668</td>
<td>-6.243341</td>
<td>2.402872e-05</td>
<td>0.08158869</td>
<td>2.624951</td>
</tr>
</tbody>
</table>
Visualizing results: the Volcano Plot

\[
\text{p.fdr} = \text{p.adjust(fit2$p.value[,3],method = "fdr")}
\]
\[
\text{plot(fit2$coefficients[,3],-log10(p.fdr),xlab = "Contrast estimate", ylab = ",log10(pvalue)",main = "",cex = 0.6, pch = 19)
}\]
\[
\text{points(fit2$coefficients[,3][p.fdr < 0.1],-log10(p.fdr)[p.fdr < 0.1], cex = 0.6, pch = 19, col = "red")
}\]
\[
\text{abline(h = -log10(0.1),col = "blue",lty =2,lwd = 1.5)
}\]
Visualization through the heatmap

```r
> topGenes = sort(p.fdr)
> top.vsn = dati.vsn[row.names(exprs(dati.vsn)) %in% names(topGenes)[1:100],
  treatment %in% LETTERS[1:2]]
> topSymbol = fit2$genes$TargetID[row.names(fit2$genes) %in% names(topGenes)]

> library(cluster)
> library(amap)
> library(gplots)
> library(heatmap.plus)

> hc <- hclust(Dist(exprs(top.vsn), method="correlation"),
  method="average")  # clustering of samples
> hr <- hclust(Dist(t(exprs(top.vsn)), method="correlation"),
  method="average")  # clustering of genes
> heatmap.plus(exprs(top.vsn), Rowv=as.dendrogram(hc), Colv=as.dendrogram(hr),
  col=greenred(75), cexRow=0.4, cexCol=0.7, labRow = topSymbol)
```
Finally...the heatmap!

Heatmap of top 100 genes differentially expressed in contrast 3 ('B - A')
Some references for this application

Anti-phospholipid induced murine fetal loss: Novel protective effect of a peptide targeting the β2 glycoprotein I phospholipid-binding site. Implications for human fetal loss

Yeny Martínez de la Torre¹,², Francesca Pregnolato³, Fabio D’Amelio⁴, Claudia Grossi⁵, Nicoletta Di Simone⁶, Fabio Pasqualini⁷, Manuela Nebuloni⁸, Pojen Chen⁹, Silvia Pierangeli⁶, Niccolò Bassani¹, Federico Ambrogi¹, Maria-Orietta Borghi¹⁰, Annunziata Vecchi¹.

Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments*

Gordon K. Smyth
Walter and Eliza Hall Institute of Medical Research
Vic 3050, Australia

**limma:**
Linear Models for Microarray Data
User’s Guide

Gordon K. Smyth, Matthew Ritchie, Natalie Thorne and James Wettenhall
The Walter and Eliza Hall Institute of Medical Research
Melbourne, Australia
### Differences between SAGE and microarrays

<table>
<thead>
<tr>
<th></th>
<th>Microarrays</th>
<th>SAGE/RNASEq</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of Data</strong></td>
<td>Relative intensities</td>
<td>Read counts</td>
</tr>
<tr>
<td><strong>Distribution</strong></td>
<td>$N(\mu, \sigma)$</td>
<td>$\text{Poisson}(\mu)$ or $\text{NB}(r, p)$</td>
</tr>
<tr>
<td><strong>Normalization</strong></td>
<td>Changes data intensities</td>
<td>adds an offset in modeling</td>
</tr>
<tr>
<td><strong>Packages</strong></td>
<td>limma, samr</td>
<td>edgeR, DESeq, PoissonSeq, BaySeq</td>
</tr>
</tbody>
</table>

**Table:** Differences between SAGE and Microarrays. The packages in the table are just a part of all packages available on Bioconductor for linear modeling of expression data.
Analysis using the edgeR package

```r
> library(edgeR)
> path = "/Users/Nico82/Documents/Lavoro/SAGE/IPAH_SANI.Giugno2012/
> targets = read.delim(paste(path,"IPAH_SANI.targets.txt",sep=""),
        stringsAsFactors = FALSE)
> d = readDGE(targets,path = path,header=FALSE)
> colnames(d$counts) = paste("Sample",1:ncol(d$counts),sep="")
> row.names(d$counts) = paste("Gene", 1:nrow(d$counts),sep="")
> head(d$counts)[,1:7]
```

<table>
<thead>
<tr>
<th></th>
<th>Sample1</th>
<th>Sample2</th>
<th>Sample3</th>
<th>Sample4</th>
<th>Sample5</th>
<th>Sample6</th>
<th>Sample7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene1</td>
<td>25</td>
<td>29</td>
<td>29</td>
<td>170</td>
<td>259</td>
<td>20</td>
<td>138</td>
</tr>
<tr>
<td>Gene2</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Gene3</td>
<td>80</td>
<td>192</td>
<td>90</td>
<td>5</td>
<td>0</td>
<td>33</td>
<td>17</td>
</tr>
<tr>
<td>Gene4</td>
<td>19</td>
<td>14</td>
<td>16</td>
<td>21</td>
<td>11</td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>Gene5</td>
<td>722</td>
<td>752</td>
<td>770</td>
<td>864</td>
<td>84</td>
<td>712</td>
<td>608</td>
</tr>
<tr>
<td>Gene6</td>
<td>145</td>
<td>294</td>
<td>224</td>
<td>78</td>
<td>2</td>
<td>314</td>
<td>85</td>
</tr>
</tbody>
</table>
Estimating the *sequencing depth*

**SEQUENCING DEPTH**: estimate of the relative measure of the number of counts produced by a sample/library/experiment.

↓

- **Total Count**: sums the read counts for each sample
- **TMM**: or "Trimmed Mean of M values", estimates the depth by computing a mean weight for the inverse of the variance of all M (gene-wise log-fold change) after excluding genes with an extreme M value (implemented in *edgeR*)
- **RLE**: or "Relative Log Expression", computes a ratio between a transcript and the geometric mean of all genes of a sample/library and then considers the median of these ratios (implemented in *edgeR* and *DESeq*)
- **quantile**: considers the third quartile of the count distribution within each sample (implemented in *edgeR*)
- **PoissonSeq**: extracts a subset of genes S not differentially expressed using a specific criterion, then estimates the ratio between the sum of the counts of genes $\in S$ for a sample and the sum of the counts of genes $\in S$ for all samples (implemented in *PoissonSeq*)
> library(DESeq)
> library(PoissonSeq)

> d$samples$lib.size = colSums(d$counts)
> d1 = calcNormFactors(d)$samples$norm.factors
> d2 = calcNormFactors(d, method = "RLE")$samples$norm.factors
> d3 = calcNormFactors(d, method = "upperquartile")$samples$norm.factors
> d4 = PS.Est.Depth(d$counts, ct.sum = 0, ct.mean = 0)
> dati.DE = newCountDataSet(d$counts, d$samples$group)
> dati.DE = estimateSizeFactors(dati.DE)
> d5 = sizeFactors(dati.DE)
> data.di = data.frame(TMM = d1, RLE1 = d2, RLE2 = d5, uppquart = d3, PoissonSeq = d4)
> const = ((d2*d$samples$lib.size)/d5)[1]
> d1 = (d1*d$samples$lib.size)/const
> d2 = (d2*d$samples$lib.size)/const
> d3 = (d3*d$samples$lib.size)/const
> data.di$TMM = d1
> data.di$RLE1 = d2
> data.di$uppquart = d3
Comparing estimates of depth

> library(epiR)

```r
> panel.cor <- function(x, y, digits=4,
>                        prefix="", cex.cor = 2, ...)
> {
>   usr <- par("usr"); on.exit(par(usr))
>   par(usr = c(0, 1, 0, 1))
>   r <- epi.ccc(x, y)$rho.c$est
>   txt <- format(c(r, 0.123456789),
>                  digits=digits)[1]
>   txt <- paste(prefix, txt, sep="")
>   text(0.5, 0.5, txt, cex = cex.cor)
> }
```

```r
> pairs(data.di,pch = 19, cex = 0.6,
>        lower.panel = panel.cor)
```

NO RELEVANT DIFFERENCES!

Bassani NP et al.  R and Bioconductor in genomics
### Filtering: the *genefilter* approach

```r
> row.Var.counts = rowVars(d$counts)
> row.Var.logcounts = rowVars(log(d$counts + 1))
> row.cpm = rowSums(cpm(d))
> row.counts = rowSums(d$counts)
> thr = seq(0, 0.8, length.out = 30)
> rej.sum.cpm = NULL
> rej.var.count = NULL
> rej.var.logcount = NULL
> rej.sum.raw = NULL
> for (i in 1:length(thr)){
>   print(thr[i])
>   d.sum.cpm = d[row.cpm >= quantile(row.cpm, probs = thr[i]),]
>   d.sum.raw = d[row.counts >= quantile(row.counts, probs = thr[i]),]
>   d.var.count = d[row.Var.counts >= quantile(row.Var.counts, probs = thr[i]),]
>   d.var.logcount = d[row.Var.logcounts >= quantile(row.Var.logcounts, probs = thr[i]),]
>   d.sum.cpm$samples$lib.size = colSums(d.sum.cpm$counts)
>   d.sum.cpm = calcNormFactors(d.sum.cpm)
>   d.sum.raw$samples$lib.size = colSums(d.sum.raw$counts)
>   d.sum.raw = calcNormFactors(d.sum.raw)
>   d.var.count$samples$lib.size = colSums(d.var.count$counts)
>   d.var.count = calcNormFactors(d.var.count)
>   d.var.logcount$samples$lib.size = colSums(d.var.logcount$counts)
>   d.var.logcount = calcNormFactors(d.var.logcount)
>   test.sum.cpm = exactTest(d.sum.cpm, dispersion = "tagwise")
>   test.sum.raw = exactTest(d.sum.raw, dispersion = "tagwise")
>   test.var.count = exactTest(d.var.count, dispersion = "tagwise")
>   test.var.logcount = exactTest(d.var.logcount, dispersion = "tagwise")
>   rej.sum.cpm = c(rej.sum.cpm, sum(p.adjust(test.sum.cpm$table$PValue, method="fdr") < 0.1))
>   rej.var.count = c(rej.var.count, sum(p.adjust(test.var.count$table$PValue, method="fdr") < 0.1))
>   rej.var.logcount = c(rej.var.logcount, sum(p.adjust(test.var.logcount$table$PValue, method="fdr") < 0.1))
>   rej.sum.raw = c(rej.sum.raw, sum(p.adjust(test.sum.raw$table$PValue, method="fdr") < 0.1))
> }
```
The filtering curves

```r
> plot(thr,rej.sum.cpm,type = "l",
> col="red",xlab = "Fraction filtered out",
> ylab = "Rejected null hypothesis",
> ylim = c(0,120))
> lines(thr,rej.sum.raw,col = "red",
> lty =2)
> lines(thr,rej.var.logcount,
> col = "blue")
> lines(thr,rej.var.count,col = "blue",
> lty =2)
> legend("topleft",lty = c(1,2,1,2),
> col = c("red","red",
> "blue","blue"),
> legend = c("Overall CPM sum",
> "Overall raw-count sum",
> "Overall log-count variance",
> "Overall raw-count variance"),
> cex = 0.8)
```
Estimating over-dispersion $\phi$

```r
> rejections = data.frame(Soglia = thr, CPM = rej.sum.cpm, counts = rej.sum.raw,
>                        logvariance = rej.var.logcount, variance = rej.var.count)
> row.names(rejections) = NULL
> d.var = d[row.Var.counts > quantile(row.Var.counts, 
>           probs = round(rejections$Soglia[apply(rejections[,-1],2,which.max)[4]],4)],]
> dim(d.var)

[1] 8999  21

> d.var$samples$group = rep.int(LETTERS[1:2],c(13,8))
> d.var$samples$lib.size = colSums(d.var$counts)
> d.var = calcNormFactors(d.var)
> d.var = estimateCommonDisp(d.var)
> d.var = estimateTrendedDisp(d.var)
> d.var = estimateTagwiseDisp(d.var, trend = "none")
```
The `plotBCV` function

```r
> plotBCV(d.var)
```
Identify technical/batch effects

```r
> plotMDS(d.var, top = 8999, labels = targets$Anno)
```
Differential expression using GLMs

```
> design = model.matrix(~ factor(targets$group) + targets$Anno)
> fit1.var = glmFit(d.var, design = design, dispersion =
>     d.var$tagwise.dispersion)
> lrt1.var = glmLRT(d.var, fit1.var, coef = 2)
> TT.var = topTags(lrt1.var, n = dim(d.var)[1])$table
> head(TT.var)
```

<table>
<thead>
<tr>
<th>logFC</th>
<th>logCPM</th>
<th>LR</th>
<th>PValue</th>
<th>FDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene6298</td>
<td>-3.357</td>
<td>2.906</td>
<td>14.2086</td>
<td>0.0001635980</td>
</tr>
<tr>
<td>Gene17019</td>
<td>8.017</td>
<td>-2.199</td>
<td>14.0089</td>
<td>0.0001819423</td>
</tr>
<tr>
<td>Gene16743</td>
<td>7.112</td>
<td>-2.907</td>
<td>13.2271</td>
<td>0.0002759215</td>
</tr>
<tr>
<td>Gene16490</td>
<td>7.236</td>
<td>-2.988</td>
<td>13.2181</td>
<td>0.0002772458</td>
</tr>
<tr>
<td>Gene15437</td>
<td>7.551</td>
<td>-2.583</td>
<td>13.1212</td>
<td>0.0002919763</td>
</tr>
<tr>
<td>Gene1097</td>
<td>7.228</td>
<td>-2.716</td>
<td>13.0146</td>
<td>0.0003090798</td>
</tr>
</tbody>
</table>
..what happens using DESeq (short)?

```r
> library(DESeq)
> DE = newCountDataSet(d$counts,targets)
> DE.var = DE[row.Var.counts > quantile(row.Var.counts,
>    probs = round(rejections$Soglia[apply(rejections[,-1],2,
>    which.max)[4]],4)),]
> DE.var = estimateSizeFactors(DE.var)
> DE.var = estimateDispersions(DE.var,method = "blind",
>    sharingMode = "fit-only", fitType = "local")
> fit1.NB.var = fitNbinomGLMs(DE.var, count ~ as.factor(targets$group) +
>    as.factor(targets$Anno))
> fit0.NB.var = fitNbinomGLMs(DE.var, count ~ as.factor(targets$Anno))
> LRT.var = nbinomGLMTest(fit1.NB.var,fit0.NB.var)
> LRT.var.FDR = p.adjust(LRT.var,method ="fdr")
> fit1.NB.var$PVal = LRT.var
> fit1.NB.var$Adj.PVal = LRT.var.FDR
> names(fit1.NB.var)[1:4] = c("Intercept","Group","Year2011","Year2012")
> fit1.NB.var[fit1.NB.var$Adj.PVal < 0.05,c(1:4,7:8)]
```

<table>
<thead>
<tr>
<th></th>
<th>Intercept</th>
<th>Group</th>
<th>Year2011</th>
<th>Year2012</th>
<th>PVal</th>
<th>Adj.PVal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene2782</td>
<td>9.119338</td>
<td>-2.244916</td>
<td>-2.9691223</td>
<td>-4.2740700</td>
<td>1.825910e-05</td>
<td>0.027385605</td>
</tr>
<tr>
<td>Gene2808</td>
<td>10.414905</td>
<td>-2.657267</td>
<td>-2.6645535</td>
<td>-0.9157067</td>
<td>4.748111e-07</td>
<td>0.002136412</td>
</tr>
<tr>
<td>Gene2926</td>
<td>-1.757687</td>
<td>7.585771</td>
<td>-0.9534059</td>
<td>-30.1339280</td>
<td>5.012337e-06</td>
<td>0.011276506</td>
</tr>
<tr>
<td>Gene4982</td>
<td>4.284014</td>
<td>2.671677</td>
<td>3.2438252</td>
<td>3.0701494</td>
<td>6.513934e-06</td>
<td>0.011723779</td>
</tr>
<tr>
<td>Gene6298</td>
<td>6.121526</td>
<td>-3.382693</td>
<td>-2.9695972</td>
<td>0.8306978</td>
<td>1.447139e-07</td>
<td>0.001302280</td>
</tr>
<tr>
<td>Gene10501</td>
<td>3.567870</td>
<td>3.320806</td>
<td>0.7742452</td>
<td>3.8447390</td>
<td>1.660480e-06</td>
<td>0.004980887</td>
</tr>
</tbody>
</table>
BIOINFORMATICS APPLICATIONS NOTE

Vol. 26 no. 1 2010, pages 139–140
doi:10.1093/bioinformatics/btp676

Gene expression

dgeR: a Bioconductor package for differential expression analysis of digital gene expression data
Mark D. Robinson1,2,*,†, Davis J. McCarthy2,† and Gordon K. Smyth2
1Cancer Program, Garvan Institute of Medical Research, 384 Victoria Street, Darlinghurst, NSW 2010 and 2Bioinformatics Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville,
Melbourne, Victoria 3052, Australia

Genome Biology

Differential expression analysis for sequence count data
Simon Anders*, Wolfgang Huber

Biostatistics (2011), 0, 0, pp. 1–16
doi:10.1093/biostatistics/kxt031

Normalization, testing, and false discovery rate estimation for RNA-sequencing data

JUN LI

Department of Statistics, Stanford University, Stanford, CA 94305, USA
junli07@stanford.edu

DANIELA M. WITTEN

Bassani NP et al. R and Bioconductor in genomics
Bioconductor is a powerful and flexible resource for exploring and analyzing a large variety of -omic data for the same issue (e.g. class comparison) many different packages, implementing slightly different approaches exist. Some knowledge of the (bio)statistic behind the software is needed. With the advent of NGS data the urge to build "suites" that incorporate all the phases of pre-processing (alignment, annotation, etc.) with those of statistical analysis (normalization, depth estimation, differential expression) is increasing everyday and Bioconductor offers some solutions in this sense.