Laboratório de Imunobiologia

RNA-Seq

Aula 3:
Normalização e análise diferencial

Edgar Kozlova
Gabriela Luiz

O Curso

Pré requisitos obrigatorio:

Notebook, WiFi, Notepad++, R e Rstudio

Programação das aulas:

1. Banco de dados:

NCBI/SRA
NCBI/GEO

2. RStudio e Instalação de pacotes

edgeR, limma, pheatmap, gplots, ROTS

3. Normalização e Análise Diferencial

voom, RPKM, FPKM, TPM, CPM, counts

4. Análise Diferencial e Visualização

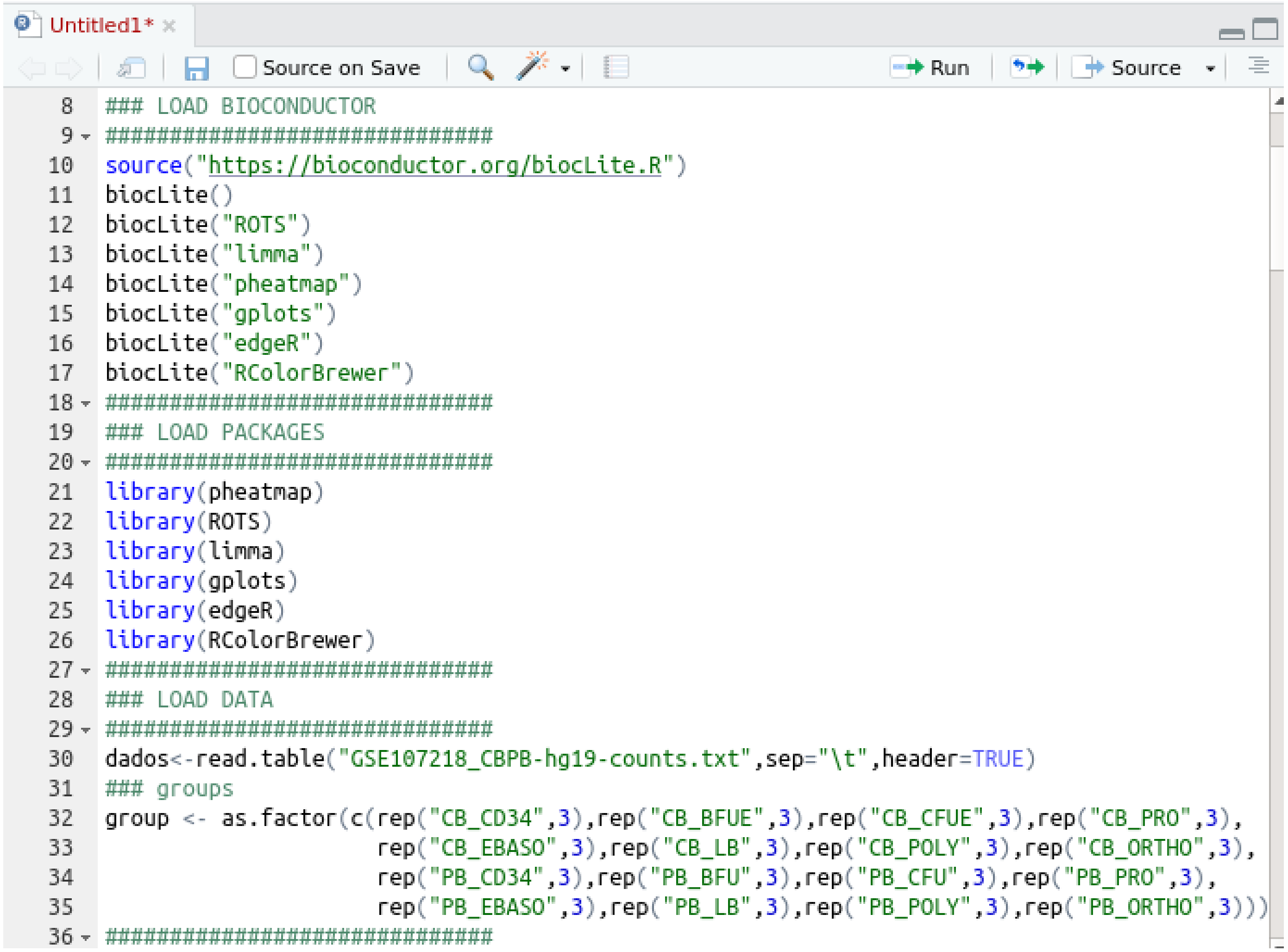
Script, MAplot, VolcanoPlot, Heatmap, Venn

Objetivo

Introduzir os principais conceitos de normalizacao de contagems.

Aplicar estes metodos sobre dados reais.

Carregar bioconductor / dados e definir os grupos da análise



```
### LOAD BIOCONDUCTOR
##############################
source("https://bioconductor.org/biocLite.R")
biocLite()
biocLite("ROTS")
biocLite("limma")
biocLite("pheatmap")
biocLite("gplots")
biocLite("edgeR")
biocLite("RColorBrewer")
##############################
### LOAD PACKAGES
##############################
library(pheatmap)
library(ROTS)
library(limma)
library(gplots)
library(edgeR)
library(RColorBrewer)
##############################
### LOAD DATA
##############################
dados<-read.table("GSE107218_CBPB-hg19-counts.txt",sep="\t",header=TRUE)
### groups
group <- as.factor(c(rep("CB_CD34",3),rep("CB_BFUE",3),rep("CB_CFUE",3),rep("CB_PRO",3),
                     rep("CB_EBASO",3),rep("CB_LB",3),rep("CB_POLY",3),rep("CB_ORTHO",3),
                     rep("PB_CD34",3),rep("PB_BFU",3),rep("PB_CFU",3),rep("PB_PRO",3),
                     rep("PB_EBASO",3),rep("PB_LB",3),rep("PB_POLY",3),rep("PB_ORTHO",3)))
##############################
```

Tratamento dos reads/counts prévios a normalização

$$\text{RPKM} = \frac{\text{Num. de reads mapeados a um gene} \times 10^3 \times 10^6}{\text{Total de reads mapeados} \times \text{tamanho do gene}}$$

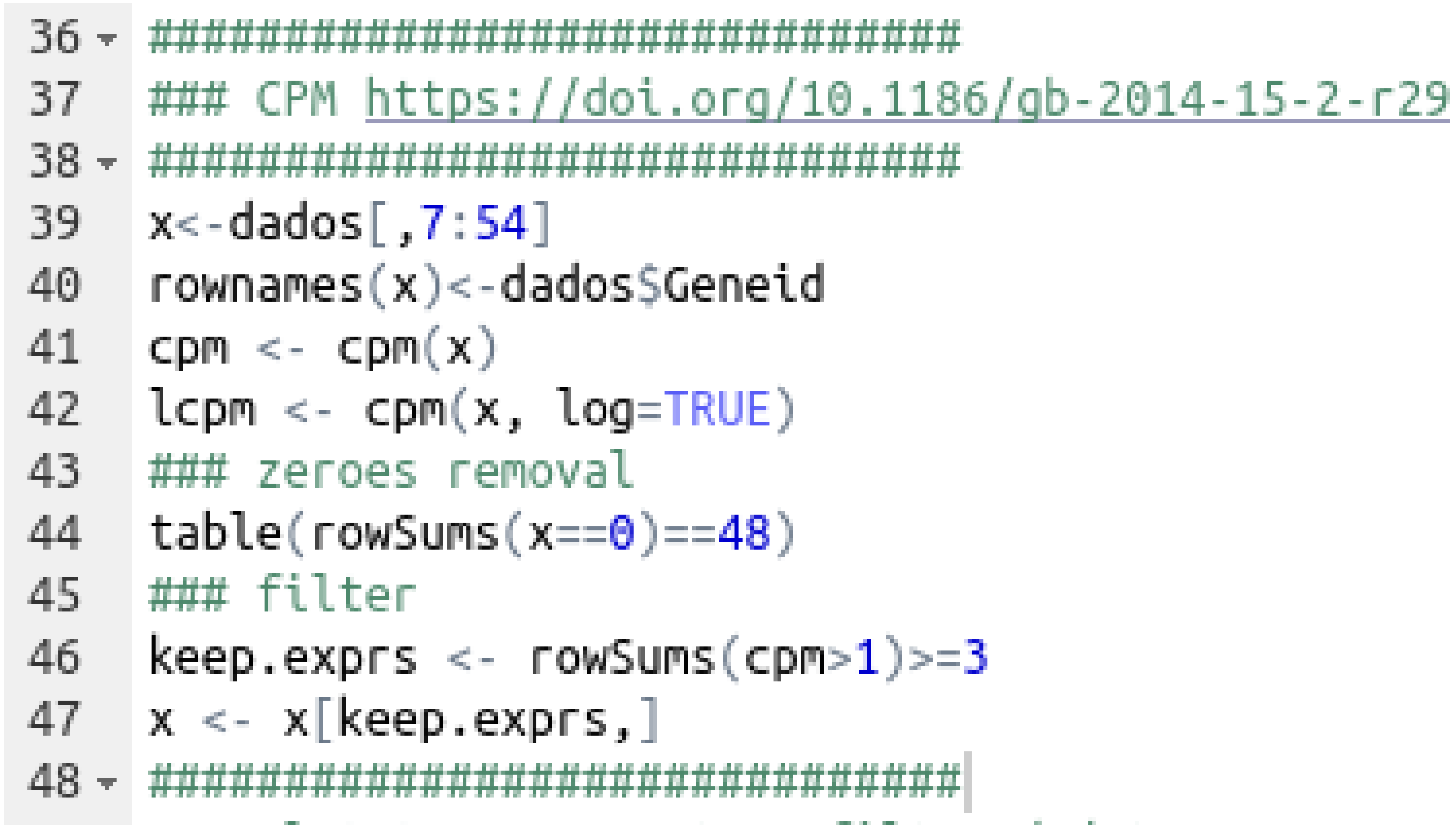
$$\text{RPM} = \frac{\text{Num. de reads mapeados a um gene} \times 10^6}{\text{Total de reads mapeados}}$$

$$\text{TPM} = \frac{\text{Num. de reads mapeados a um gene} \times \text{tamanho do read} \times 10^6}{\text{Total de reads mapeados} \times \text{tamanho do gene}}$$

$$\boxed{\text{CPM} = \frac{\text{Num. de reads mapeados a um gene}}{\text{Total de reads mapeados}} \times 10^6}$$

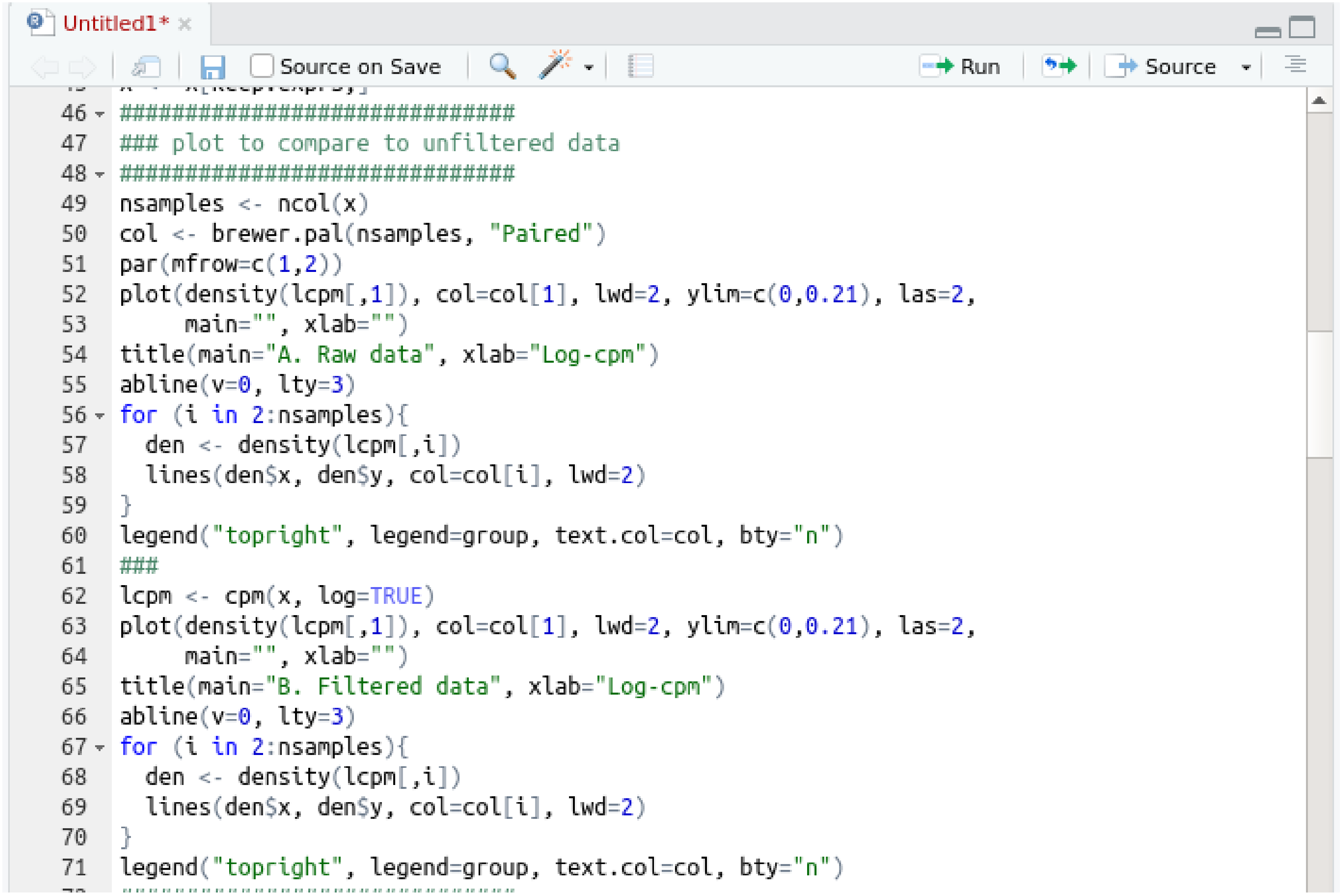
https://doi.org/10.1186/gb-2014-15-2-r29

Transformar os dados de reads/counts para CPM e filtrar genes com linhas vazias

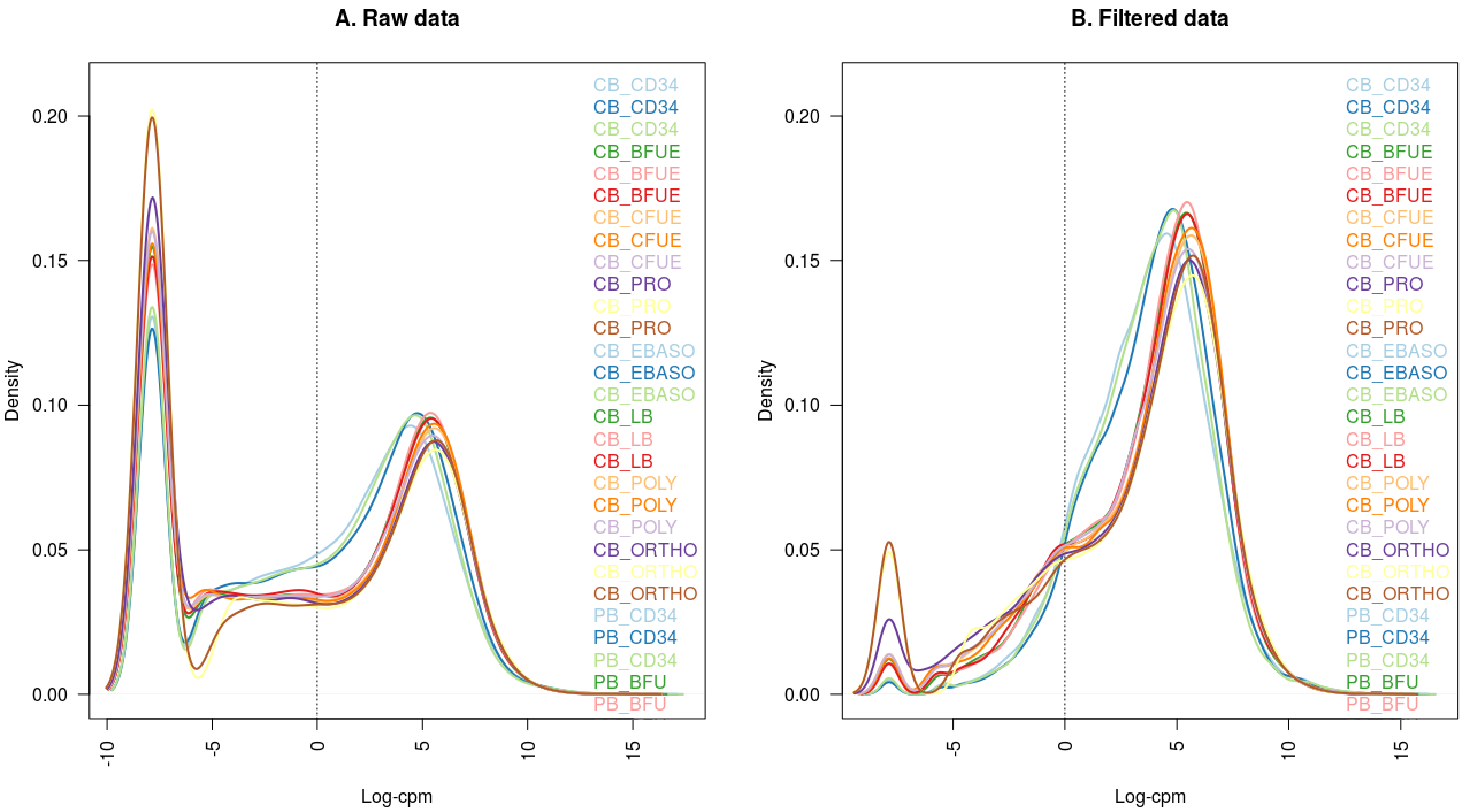


```
#################################
### CPM https://doi.org/10.1186/gb-2014-15-2-r29
#################################
x<-dados[,7:54]
rownames(x)<-dados$Geneid
cpm <- cpm(x)
lcpm <- cpm(x, log=TRUE)
### zeroes removal
table(rowSums(x==0)==48)
### filter
keep.exprs <- rowSums(cpm>1)>=3
x <- x[keep.exprs,]
#################################
```

Figura para verificar a filtragem dos genes com linhas vazias



```
#################################
### plot to compare to unfiltered data
#################################
nsamples <- ncol(x)
col <- brewer.pal(nsamples, "Paired")
par(mfrow=c(1,2))
plot(density(lcpm[,1]), col=col[1], lwd=2, ylim=c(0,0.21), las=2,
     main="", xlab="")
title(main="A. Raw data", xlab="Log-cpm")
abline(v=0, lty=3)
for (i in 2:nsamples){
  den <- density(lcpm[,i])
  lines(den$x, den$y, col=col[i], lwd=2)
}
legend("topright", legend=group, text.col=col, bty="n")
###
lcpm <- cpm(x, log=TRUE)
plot(density(lcpm[,1]), col=col[1], lwd=2, ylim=c(0,0.21), las=2,
     main="", xlab="")
title(main="B. Filtered data", xlab="Log-cpm")
abline(v=0, lty=3)
for (i in 2:nsamples){
  den <- density(lcpm[,i])
  lines(den$x, den$y, col=col[i], lwd=2)
}
legend("topright", legend=group, text.col=col, bty="n")
```


A. Raw data
B. Filtered data
Density
Log-cpm
0.00
0.05
0.10
0.15
0.20
-10
-5
0
5
10
15
CB_CD34
CB_BFUE
CB_CFUE
CB_PRO
CB_EBASO
CB_LB
CB_POLY
CB_ORTHO
PB_CD34
PB_BFU

Normalização
TMM – Trimmed Mean of M values

g = gene

k = libraria

Lg = tamanho do gene

Nk = total number of reads

Sk = total rna sample

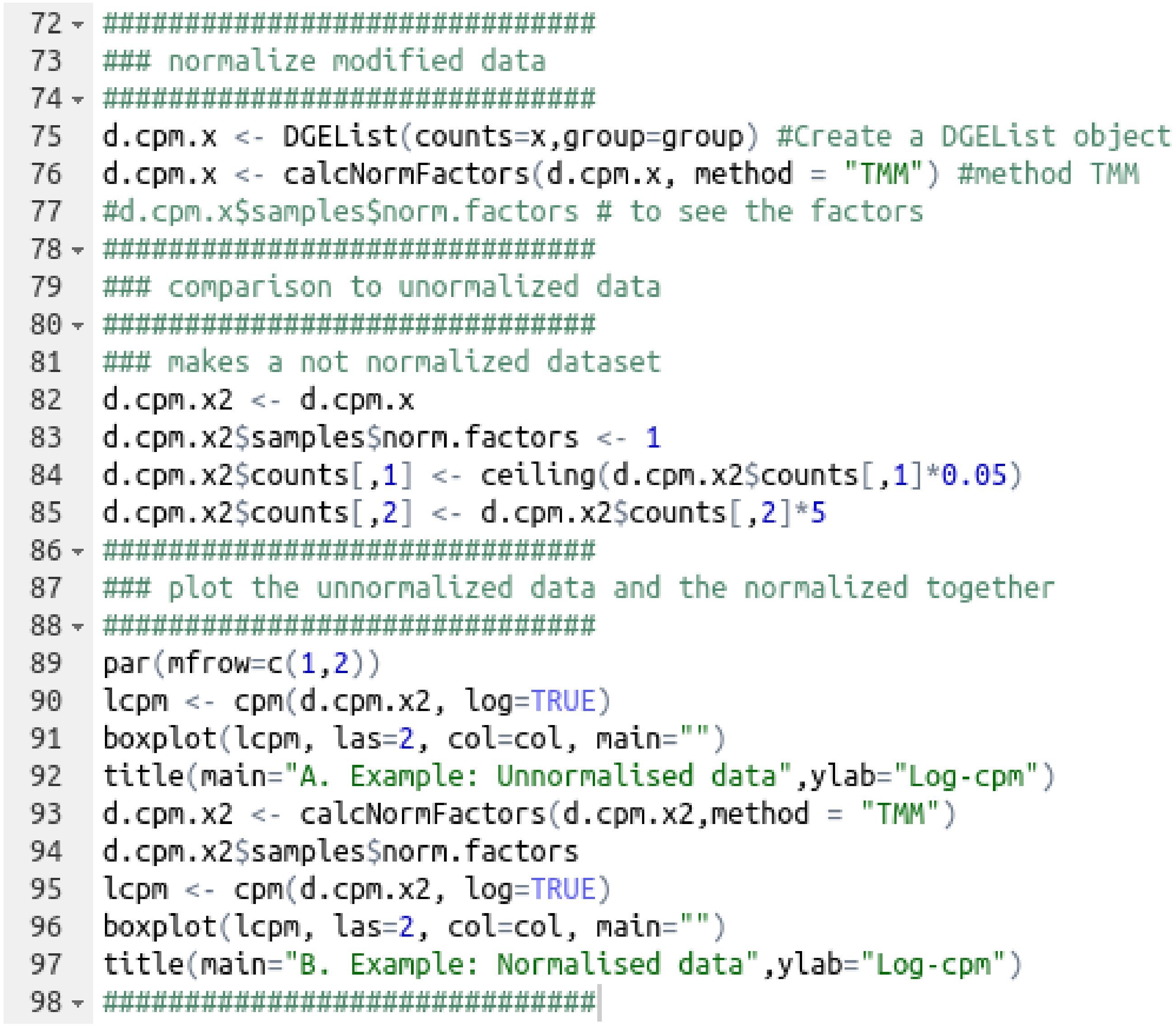
$$E[Y_{gk}] = \frac{\mu_{gk} L_g}{S_k} N_k$$

$$\text{where } S_k = \sum_{g=1}^{G} \mu_{gk} L_g;$$

$$\boxed{M_g = \log_2 \frac{Y_{gk}/N_k}{Y_{gk'}/N_{k'}}}$$

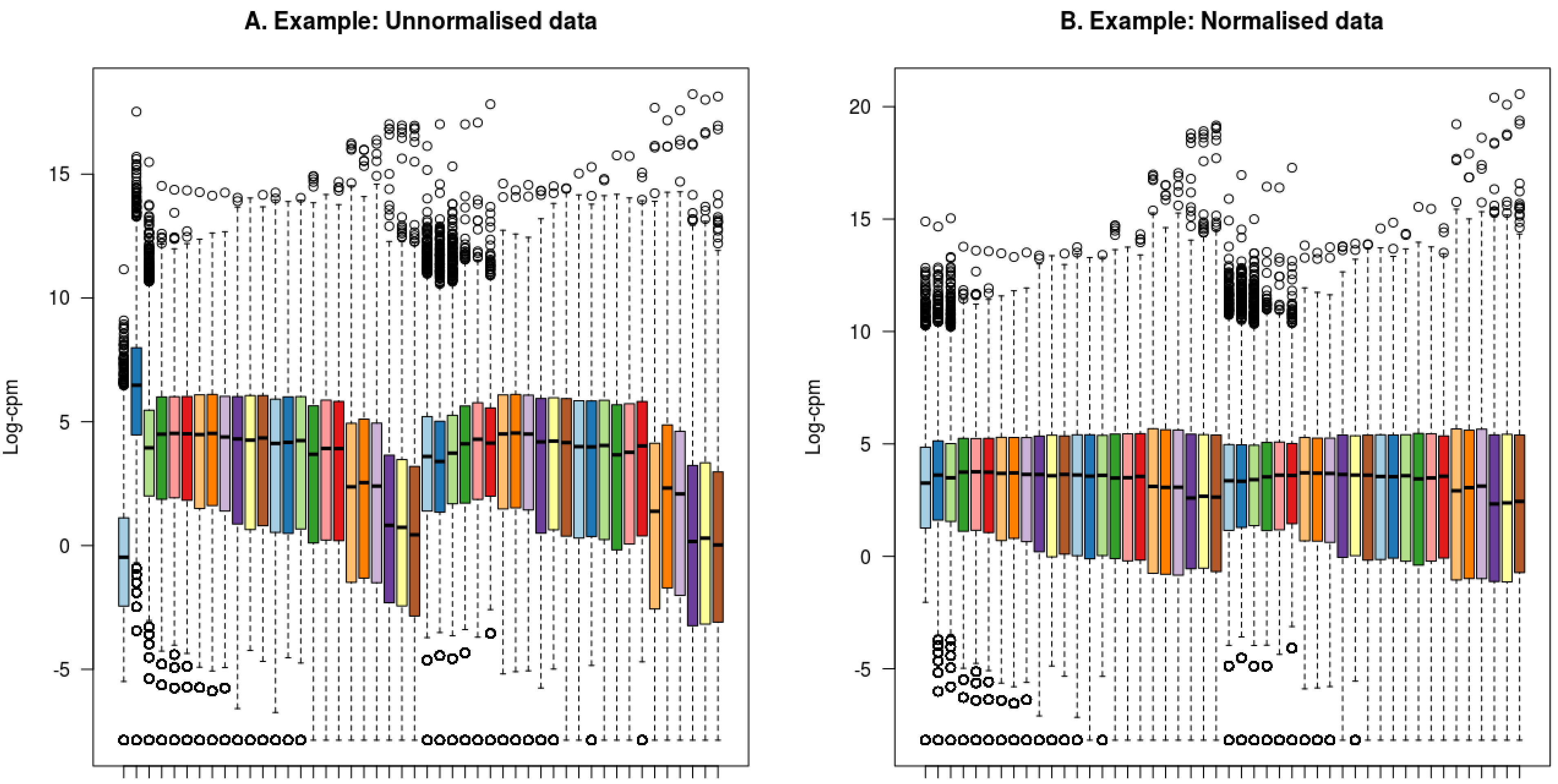
$$\boxed{A_g = \frac{1}{2}\log_2 \left(Y_{gk}/N_k \bullet Y_{gk'}/N_{k'} \right) \text{ for } Y_{g\cdot} \neq 0}$$

NORMALIZAÇÃO “TMM” method

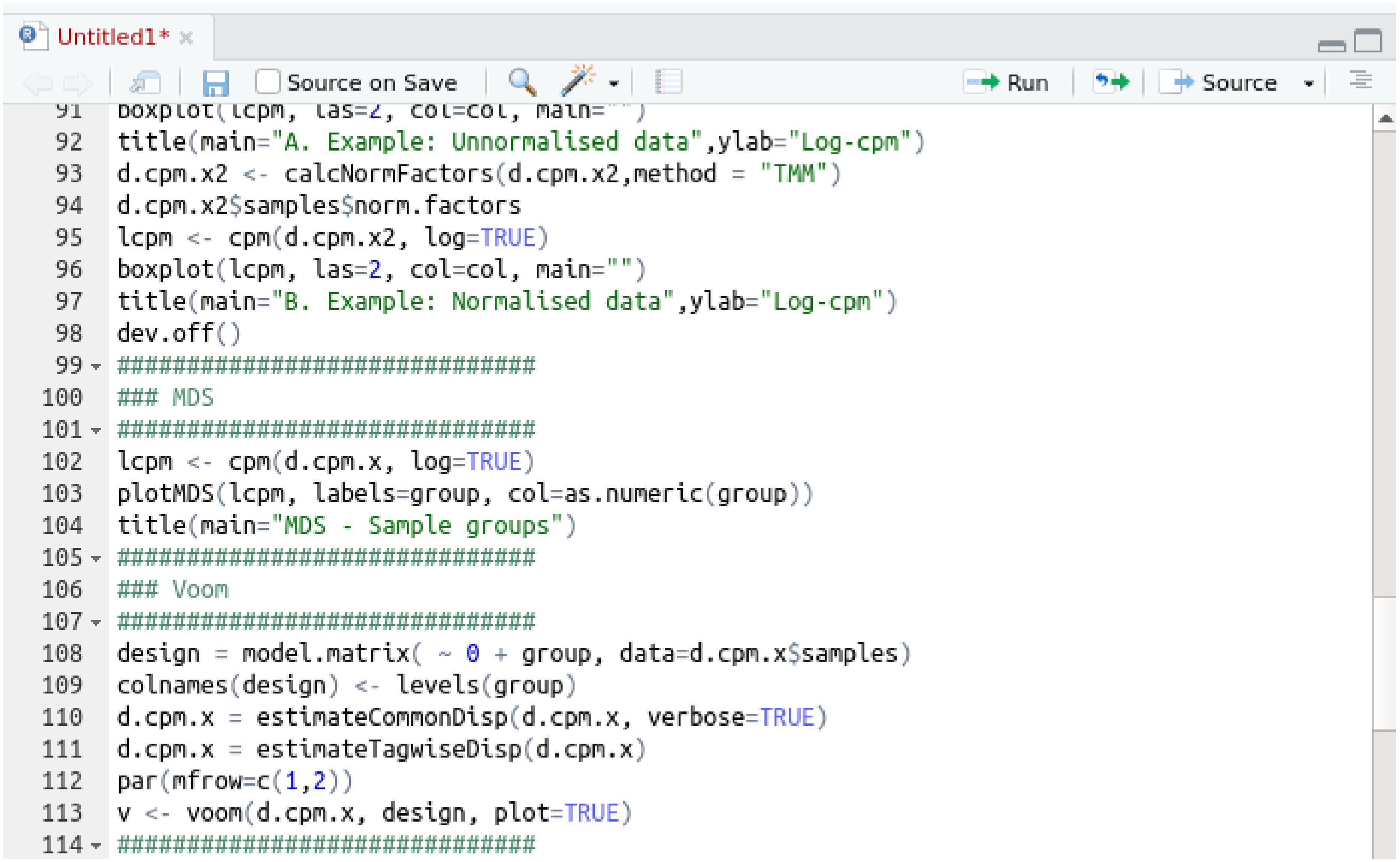


```r
################################
### normalize modified data
################################
d.cpm.x <- DGEList(counts=x,group=group) #Create a DGEList object
d.cpm.x <- calcNormFactors(d.cpm.x, method = "TMM") #method TMM
#d.cpm.x$samples$norm.factors # to see the factors
################################
### comparison to unormalized data
################################
### makes a not normalized dataset
d.cpm.x2 <- d.cpm.x
d.cpm.x2$samples$norm.factors <- 1
d.cpm.x2$counts[,1] <- ceiling(d.cpm.x2$counts[,1]*0.05)
d.cpm.x2$counts[,2] <- d.cpm.x2$counts[,2]*5
################################
### plot the unnormalized data and the normalized together
################################
par(mfrow=c(1,2))
lcpm <- cpm(d.cpm.x2, log=TRUE)
boxplot(lcpm, las=2, col=col, main="")
title(main="A. Example: Unnormalised data",ylab="Log-cpm")
d.cpm.x2 <- calcNormFactors(d.cpm.x2,method = "TMM")
d.cpm.x2$samples$norm.factors
lcpm <- cpm(d.cpm.x2, log=TRUE)
boxplot(lcpm, las=2, col=col, main="")
title(main="B. Example: Normalised data",ylab="Log-cpm")
################################
```

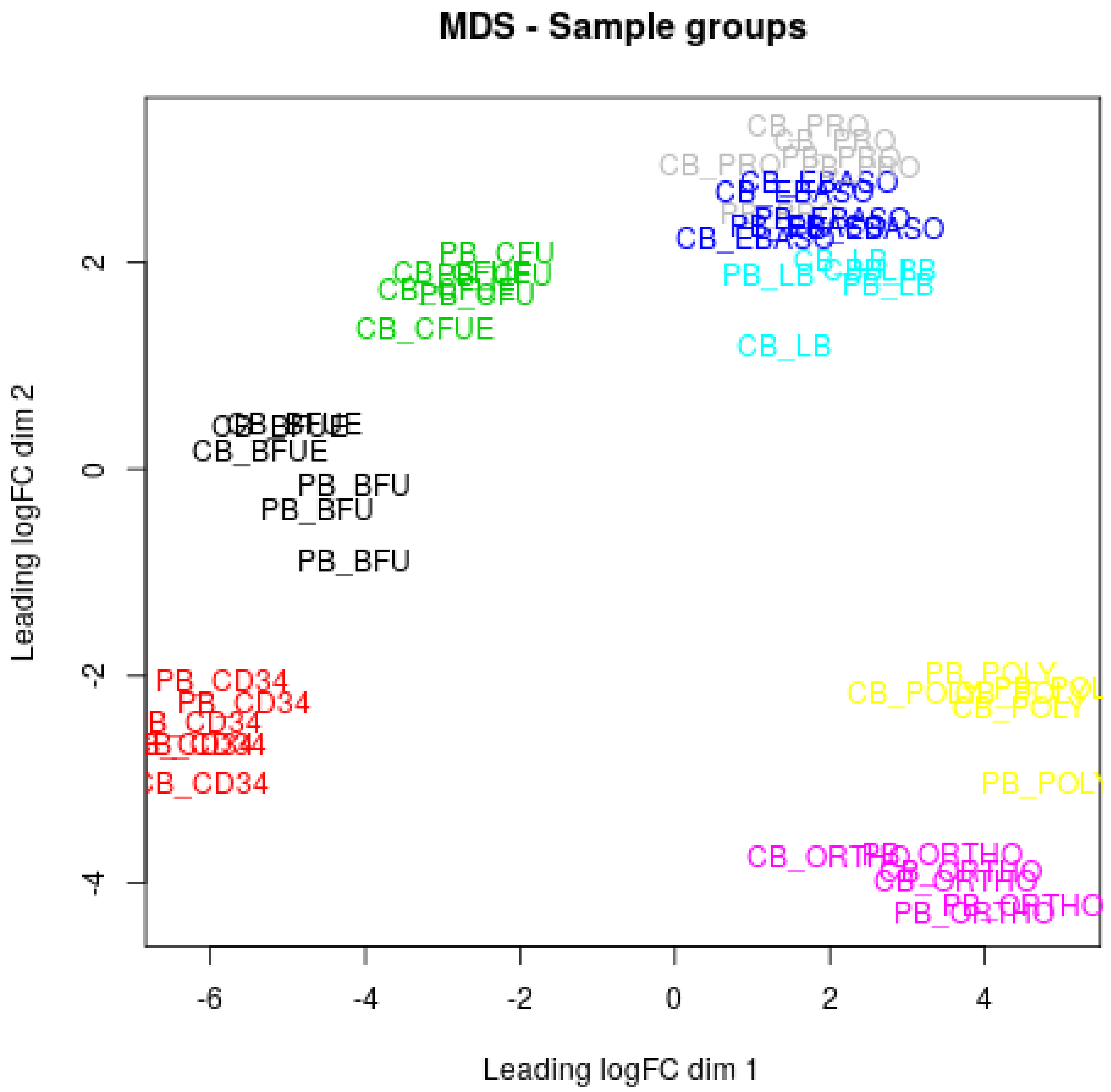
Normalização “TMM”



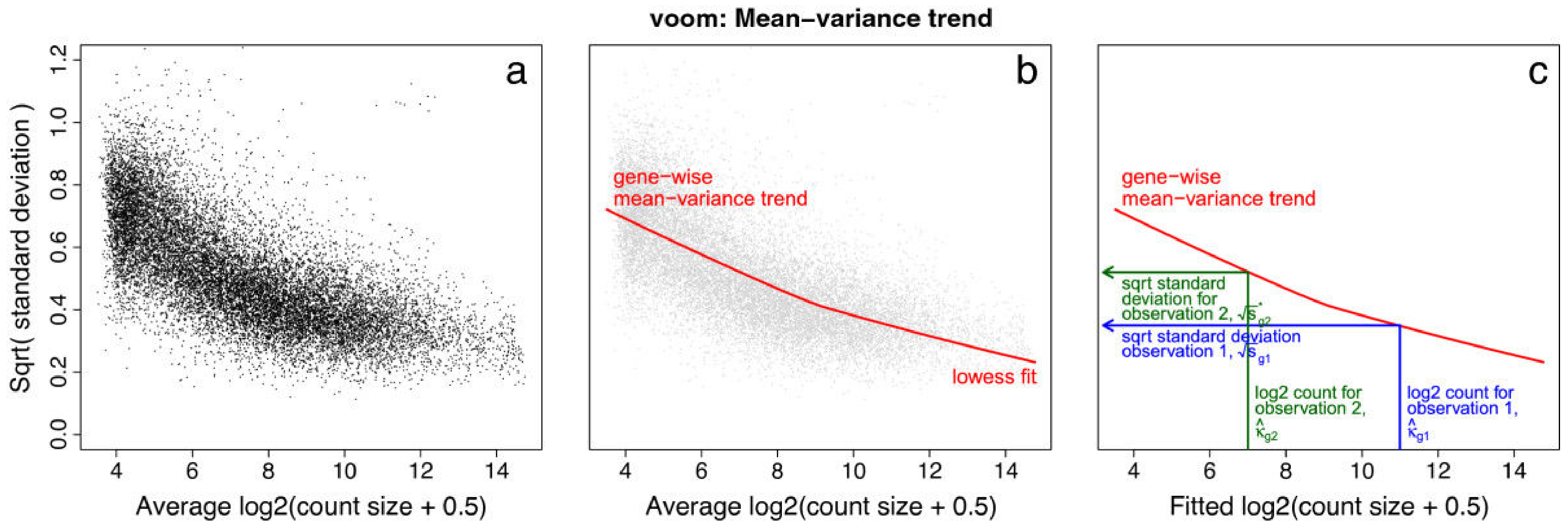
MultiDimensionalScaling

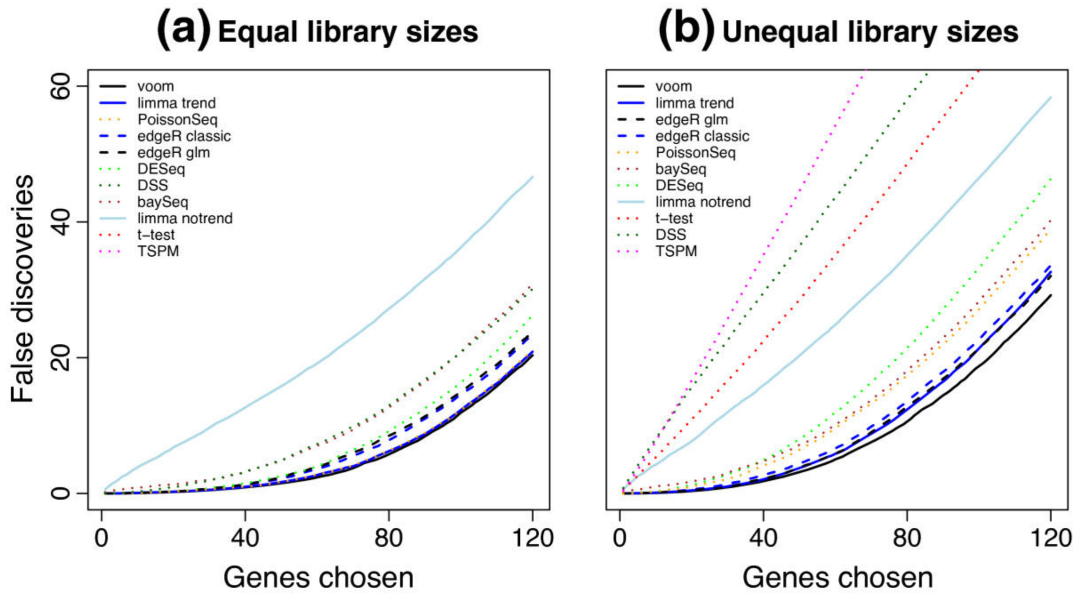


```r
boxplot(lcpm, las=2, col=col, main="")
title(main="A. Example: Unnormalised data",ylab="Log-cpm")
d.cpm.x2 <- calcNormFactors(d.cpm.x2,method = "TMM")
d.cpm.x2$samples$norm.factors
lcpm <- cpm(d.cpm.x2, log=TRUE)
boxplot(lcpm, las=2, col=col, main="")
title(main="B. Example: Normalised data",ylab="Log-cpm")
dev.off()
##############################
### MDS
##############################
lcpm <- cpm(d.cpm.x, log=TRUE)
plotMDS(lcpm, labels=group, col=as.numeric(group))
title(main="MDS - Sample groups")
##############################
### Voom
##############################
design = model.matrix( ~ 0 + group, data=d.cpm.x$samples)
colnames(design) <- levels(group)
d.cpm.x = estimateCommonDisp(d.cpm.x, verbose=TRUE)
d.cpm.x = estimateTagwiseDisp(d.cpm.x)
par(mfrow=c(1,2))
v <- voom(d.cpm.x, design, plot=TRUE)
##############################
```


MDS - Sample groups
Leading logFC dim 1
Leading logFC dim 2
CB_PRO
PB_PRO
CB_EBASO
PB_EBASO
PB_LB
CB_LB
PB_CFU
CB_CFUE
CB_BFUE
PB_BFU
PB_CD34
CB_CD34
PB_POLY
CB_POLY
CB_ORTHO
PB_ORTHO

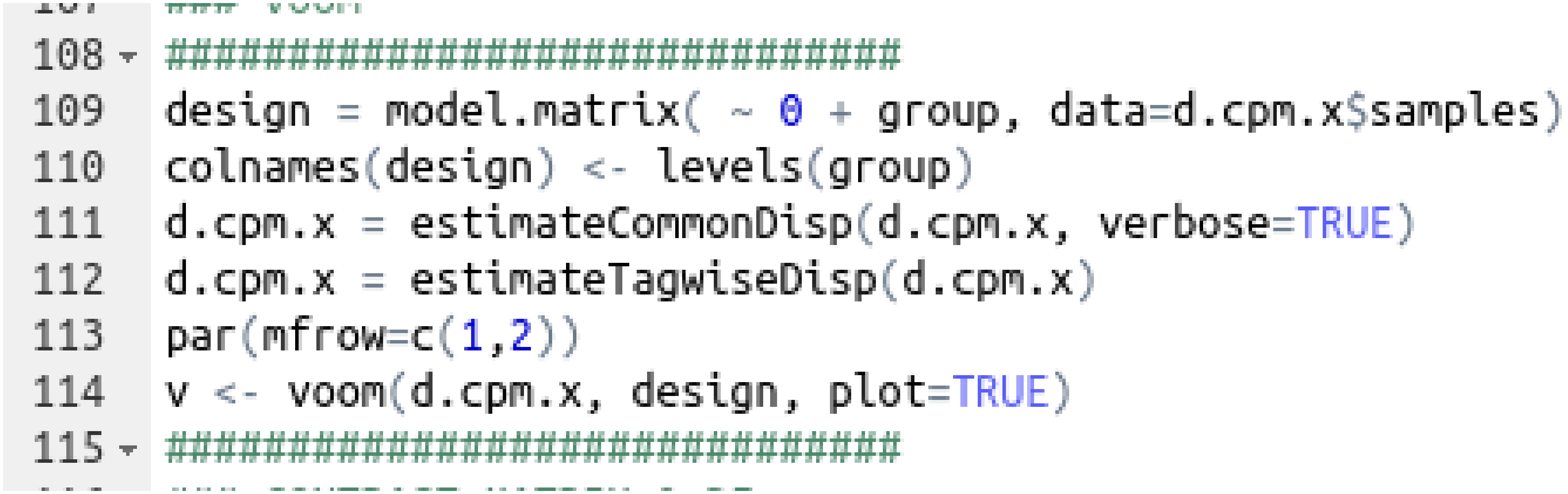
Limma – Voom !!!


voom: Mean−variance trend
a
b
c
Sqrt(standard deviation)
Average log2(count size + 0.5)
Fitted log2(count size + 0.5)
gene−wise mean−variance trend
lowess fit
sqrt standard deviation for observation 2, $\sqrt{s^*_{g2}}$
sqrt standard deviation observation 1, $\sqrt{s_{g1}}$
log2 count for observation 2, $\hat{\kappa}_{g2}$
log2 count for observation 1, $\hat{\kappa}_{g1}$

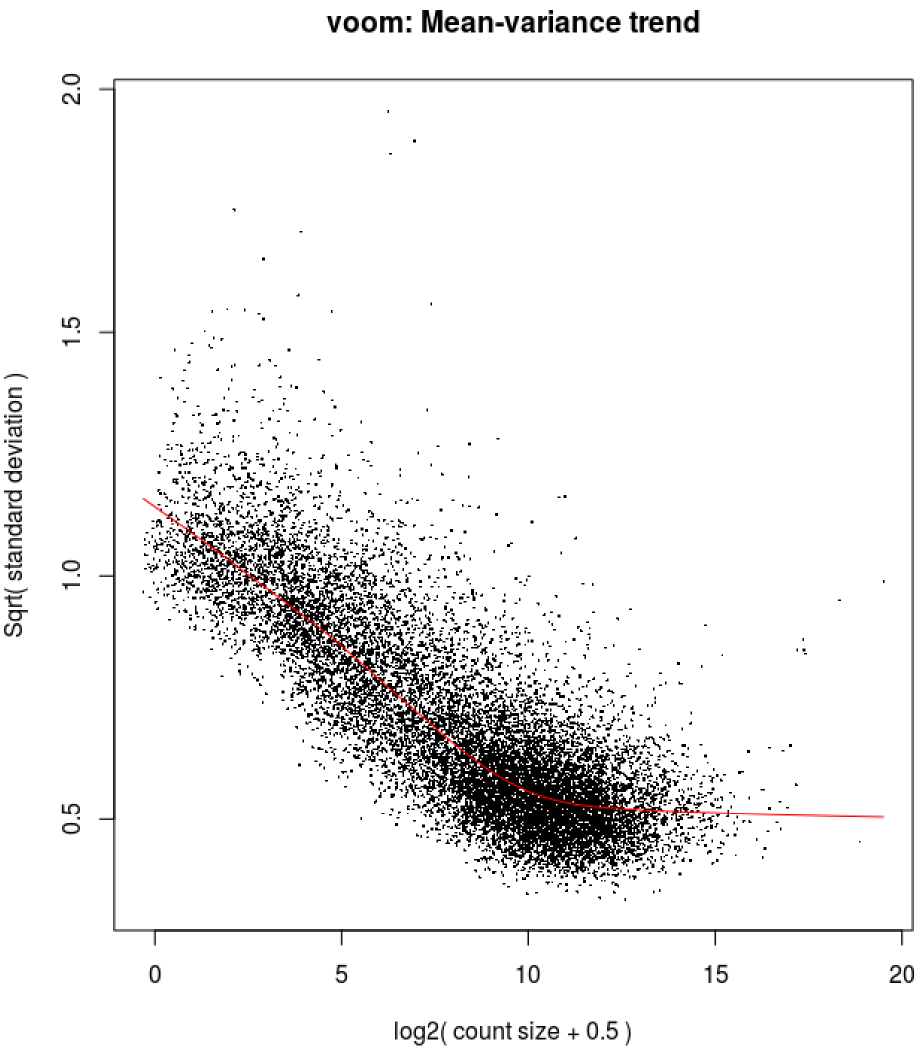

(a) Equal library sizes
(b) Unequal library sizes
False discoveries
Genes chosen
voom
limma trend
PoissonSeq
edgeR classic
edgeR glm
DESeq
DSS
baySeq
limma notrend
t−test
TSPM

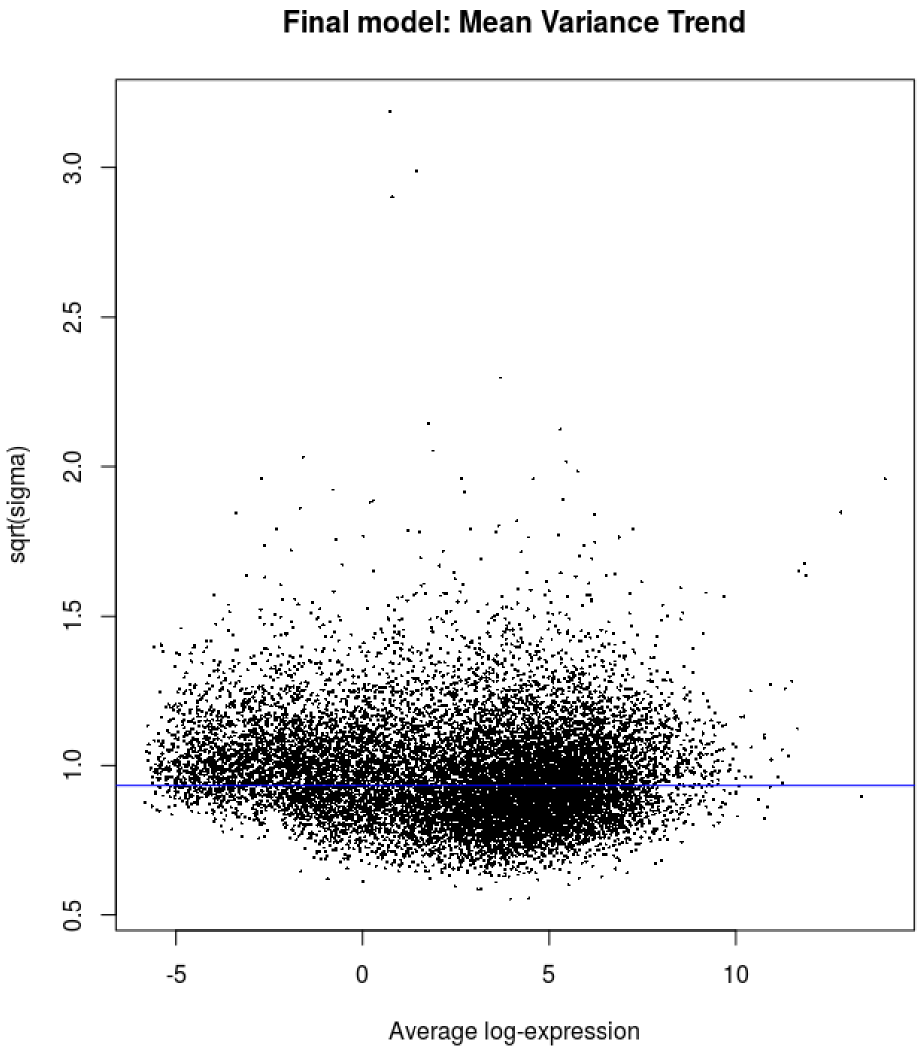
Limma – Voom !!!

Everything before limma is foreplay
Edgar 2018



```r
##############################
design = model.matrix( ~ 0 + group, data=d.cpm.x$samples)
colnames(design) <- levels(group)
d.cpm.x = estimateCommonDisp(d.cpm.x, verbose=TRUE)
d.cpm.x = estimateTagwiseDisp(d.cpm.x)
par(mfrow=c(1,2))
v <- voom(d.cpm.x, design, plot=TRUE)
##############################
```


voom: Mean-variance trend
Sqrt(standard deviation)
log2(count size + 0.5)
0.5
1.0
1.5
2.0
0
5
10
15
20


Final model: Mean Variance Trend
sqrt(sigma)
Average log-expression
0.5
1.0
1.5
2.0
2.5
3.0
-5
0
5
10