R包edgeR差异基因分析

本篇简介R语言edgeR包的差异基因分析流程。edgeR使用经验贝叶斯估计和基于负二项模型的精确检验来确定差异基因，通过在基因之间来调节跨基因的过度离散程度，使用类似于Fisher精确检验但适应过度分散数据的精确检验用于评估每个基因的差异表达。

下文中使用的示例数据及相关R代码的百度盘链接：

<https://pan.baidu.com/s/1LWIH5dNUJMxWD0MkIh7mZQ>

## 前期准备

edgeR可直接使用Bioconductor安装。

#Bioconductor 安装 edgeR

BiocManager::install('edgeR')

#加载 edgeR

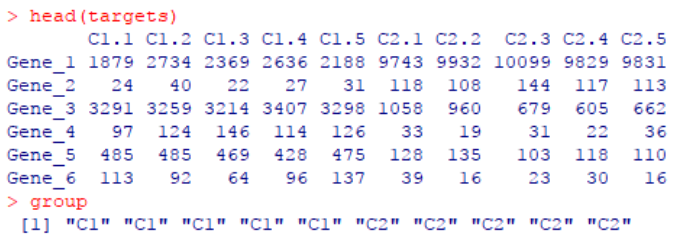
library(edgeR)

网盘示例数据“gene.txt”，是一个基因表达量数据矩阵，包含10列样本（可分为C1和C2两组，每组各5个样本）共计1000行基因。读取数据并设置分组，要保证样本名称和分组名称的顺序是一一对应的。

#读取数据，这里样本不多，直接手动定义分组了

targets <- as.matrix(read.delim('gene.txt', sep = '\t', row.names = 1))

group <- rep(c('C1', 'C2'), each = 5)

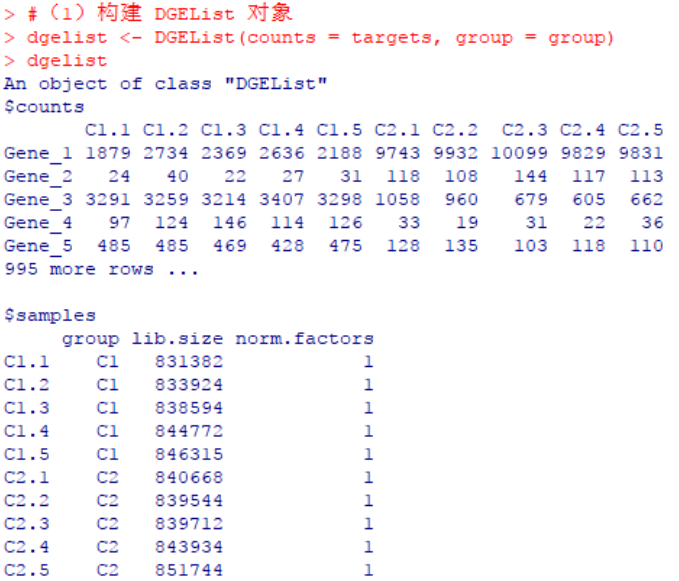


**第一步，构建DGEList对象**

首先根据基因表达量矩阵以及样本分组信息，构建DGEList对象。

#（1）构建 DGEList 对象，详见 ?DGEList

dgelist <- DGEList(counts = targets, group = group)



**第二步，过滤低表达的基因**

输入的基因表达量矩阵文件中，可能会含有一些低表达量的基因（甚至还有一些被忽略的全部为0值的行），需要在执行差异分析前将它们剔除。原因在于，这些基因未表达到具有生物学意义的程度（从生物学角度，有生物学意义的基因的表达量必须高于某一个阈值）；并且低表达量的基因受到随机因素影响比较大，故其统计结果也不可靠，还会影响p值校正过程；此外，过滤后的数据量降低，也能加快运行速率。

至于为什么要手动执行这一步，因为edgeR不会自动处理它，所以这一步不能省略。这点和DESeq2不同，DESeq2是能够自动识别这些低表达量的基因的，所以使用DESeq2时无需手动过滤。

过滤的方法有很多种。例如，我们可以指定一个阈值，将表达量总和低于该阈值的基因删除。相比之下，edgeR推荐根据CPM（count-per-million，每百万碱基中目标基因的read count值）值进行过滤，cpm()用于计算CPM值，具体详见?cpm帮助说明。例如下示例中，使用CPM值为1作为标准，即当某个基因在read count最低的样本（文库）中的count值大于(read count最低的样品count总数/1000000)，则保留。

实际的数据分析中，还需综合考虑选择一个合适的过滤条件。本示例使用了根据CPM值进行过滤的方法，并继续。

#（2）过滤 low count 数据

#例如，直接根据选定的 count 值过滤

keep <- rowSums(dgelist$counts) >= 50

dgelist <- dgelist[keep, ,keep.lib.sizes = FALSE]

#再例如，CPM 标准化（推荐）

keep <- rowSums(cpm(dgelist) > 1 ) >= 2

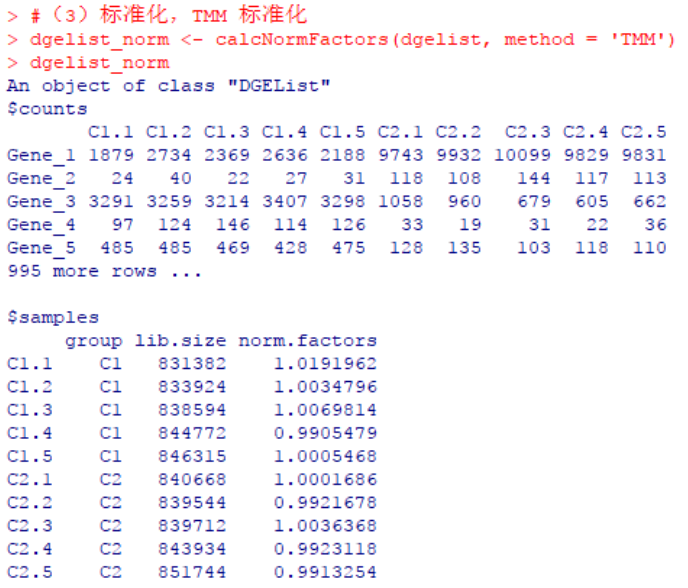
dgelist <- dgelist[keep, ,keep.lib.sizes = FALSE]

**第三步，标准化数据**

使用edgeR中的calcNormFactors()函数对数据标准化，以消除由于样品制备或建库测序过程中带来的影响。标准化的方法有多种可供选择，根据实际情况选择合适的标准化方法。这里以TMM标准化方法为例，计算得到的归一化系数被用作样本（文库）大小的缩放系数。标准化后的DGEList对象中，增添了标准化因子信息

#（3）标准化，详见?calcNormFactors，这里使用 TMM 标准化

dgelist\_norm <- calcNormFactors(dgelist, method = 'TMM')



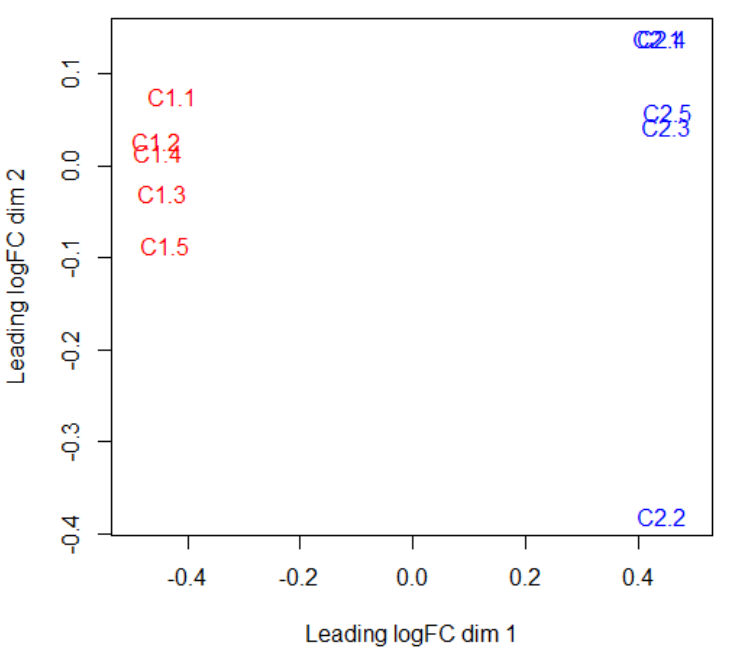
plotMDS()是limma包中的方法，绘制MDS图，使用无监督聚类方法展示出了样品间的相似性（或差异）。可据此查看各样本是否能够很好地按照分组聚类，评估试验效果，判别离群点，追踪误差的来源等。

#样本无监督聚类

#limma 包在 edgeR 加载时已经自动加载了

plotMDS(dgelist\_norm, col = rep(c('red', 'blue'), each = 5), dim = c(1, 2))

这里我们观察了前两个轴，总体来讲本示例的样本聚类效果良好。除了样本C2.2产生较大的偏离，这时可以不妨去思考下该样本重复偏差的原因。



**第四步，估算****离散值**

负二项分布（negative binomial，NB）模型需要均值和离散值两个参数。首先需要根据样本分组，或者说根据实验设计、目的，构建一个分组矩阵。然后使用该分组矩阵，通过加权似然经验贝叶斯（weighted likelihood empirical Bayes）模型，估算每个基因的负二项离散Tagwise（即经验贝叶斯稳健离散值）、Common（即经验贝叶斯稳健离散值的均值）、Trended（即经验贝叶斯稳健离散值的拟合值）。

edgeR中，分组矩阵使用model.matrix()获得，并可以通过estimateDisp()估算离散值。

#（4）估算离散值，详见 ?estimateDisp

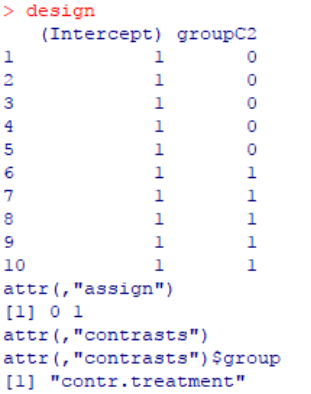
design <- model.matrix(~group) #构建分组矩阵

dge <- estimateDisp(dgelist\_norm, design, robust = TRUE) #估算离散值

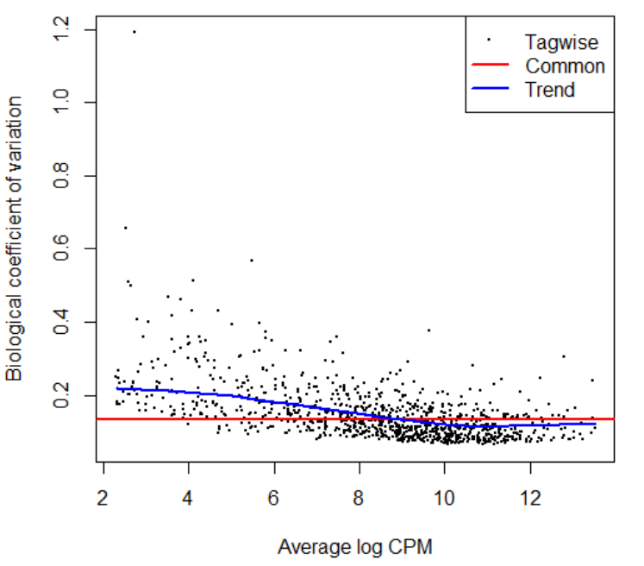
plotBCV(dge) #作图查看

对于分组矩阵design，没什么特别说明的了，就是根据之前设置好的样本分组信息group得到的。对于本示例的两个分组，分别标识为0和1，比方说下图中，对于各样本是否属于“C2分组”（groupC2），前5个样本（C1.1-C1.5）标识为0意为不属于“C2分组”，后5个（C2.1-C2.5）标识为1意为属于“C2分组”。

需要注意，标识好0、1类型后，后面的差异分析将以“分组1”的基因表达量相较于“分组0”是上调还是下调为准进行统计。因此在本示例中，后续分析获得的基因表达量上调/下调均为“分组C2”相较于“分组C1”而言的。实际的分析中，切记不要搞反了。



使用plotBCV()展示估算的离散值，结果如下。



实际上，在edgeR中还有很多函数可以单独估算离散值，如estimateCommonDisp()、estimateTagwiseDisp()、estimateGLMTagwiseDisp()、estimateGLMCommonDisp()、estimateGLMTrendedDisp()等，具体可以参阅帮助文档。当然不难看出，我们使用的estimateDisp()实际上是个组合函数，可以一步得到多个计算结果，例如在上文我们使用分组矩阵design通过estimateDisp()估算了3个值，其实就是estimateGLMTagwiseDisp()、estimateGLMCommonDisp()和estimateGLMTrendedDisp()这3个结果的组合。如果不指定分组矩阵，则将得到estimateCommonDisp()和estimateTagwiseDisp()的结果组合。

## 差异基因分析

第1-4步都是前面的准备环节，第5步终于可以正式地分析差异基因了。

edgeR包中提供了多种计算差异基因的方法，建立在不同模型的基础上，以下为主要方法的简介。

**负二项式广义对数线性模型**

首先拟合负二项式广义对数线性模型（negative binomial generalized log-linear model），获取差异基因。这种方法大致可以这样理解，如果某个基因的表达值偏离这个分布模型，那么该基因即为差异表达基因。

使用edgeR包中的函数glmFit()和glmLRT()实现，其中glmFit()用于将每个基因的read count值拟合到模型中，glmLRT()用于对给定系数进行统计检验。

#negative binomial generalized log-linear model 拟合

fit <- glmFit(dge, design, robust = TRUE) #拟合模型

lrt <- glmLRT(fit) #统计检验

topTags(lrt)

write.csv(topTags(lrt, n = nrow(dgelist$counts)), 'glmLRT.csv', quote = FALSE) #输出主要结果

dge\_de <- decideTestsDGE(lrt, adjust.method = 'fdr', p.value = 0.05) #查看默认方法获得的差异基因

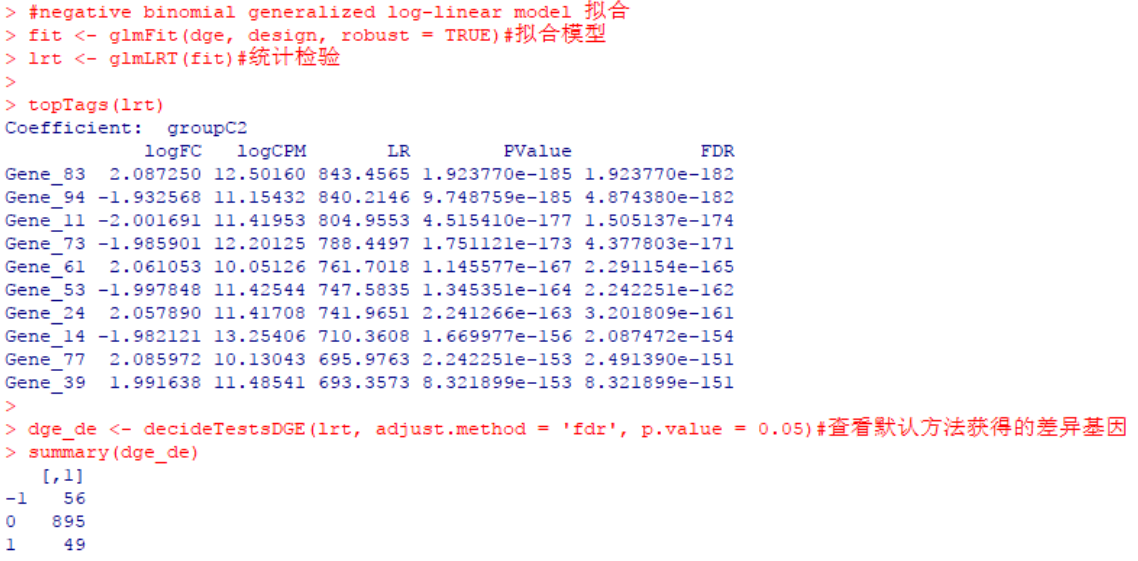
summary(dge\_de)

plotMD(lrt, status = dge\_de, values = c(1, -1), col = c('blue', 'red')) #作图观测

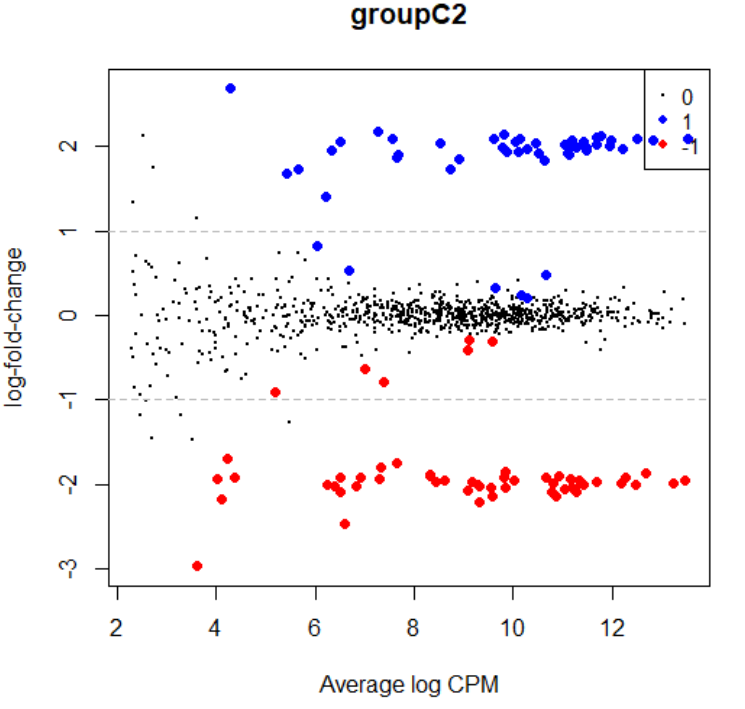
abline(h = c(-1, 1), col = 'gray', lty = 2)

topTags(lrt)简要展示了分析结果，详情参考?topTags。其中：logFC即log2转化后的 Fold Change值，但是要注意，这里不是简单的将基因的read count值直接对比，而是分别计算了基因在两组中的CPM值（每百万碱基中目标基因的read count值）然后据此计算的logFC；logCPM是log2转化后的CPM值；LR，似然比统计；PValue，差异表达的p值；FDR，FDR校正后的p值。topTags()默认展示前10行，可通过参数n指定展示的行数，这里我们指定了所有的行后输出至本地文件“glmLRT.csv”，详情自行浏览，以及使用该统计结果文件进行后续分析等（例如我们可以后续再自行根据Fold Change值以及p值等手动筛选差异基因），不再多说。

decideTestsDGE()可用于统计差异基因数量，详情参考?decideTestsDGE。屏幕输出了其默认值（供参考，大多数情况下我们还是优先根据Fold Change值以及p值等手动去筛选，而不会在意这个程序自己判断的数值），-1表示下调基因数量，1表示上调基因数量，0表示无差异基因数量。注意，对于这里的示例数据，基因表达量上调/下调均为“分组C2”相较于“分组C1”而言的。



plotMD()是limma包中的方法，可以初步绘制火山图观测差异基因分析结果，详见?plotMD说明。下图为程序默认的差异分析结果，对应了decideTestsDGE()统计的差异基因数量。纵轴为log2 Fold Change值；横轴为log2 CPM值，反映了基因表达量信息；蓝色的点表示上调基因，红色的点表示下调基因，黑色的点为无差异基因。如果你想根据导出的数据，选择使用ggplot2自定义绘制火山图，可以[查看该文](http://blog.sciencenet.cn/blog-3406804-1188483.html)。



**类似然负二项式广义对数线性模型**

我们更换其它模型，获取差异基因。对于类似然负二项式广义对数线性模型（quasi-likelihood negative binomial generalized log-linear model），使用edgeR包中的函数glmQLFit()和glmQLFTest()实现，同样地，glmQLFit()用于将每个基因的read count值拟合到模型中，glmQLFTest()用于对给定系数进行统计检验，如果某个基因的表达值偏离这个分布模型，那么该基因即为差异表达基因。

备注：相较于上一个模型，作者提到这个方法更严格一些。当然实际分析中还得视情况考虑了。

#quasi-likelihood negative binomial generalized log-linear model 拟合

fit <- glmQLFit(dge, design, robust = TRUE) #拟合模型

lrt <- glmQLFTest(fit) #统计检验

topTags(lrt)

write.csv(topTags(lrt, n = nrow(dgelist$counts)), 'glmQLFTest.csv', quote = FALSE) #输出主要结果

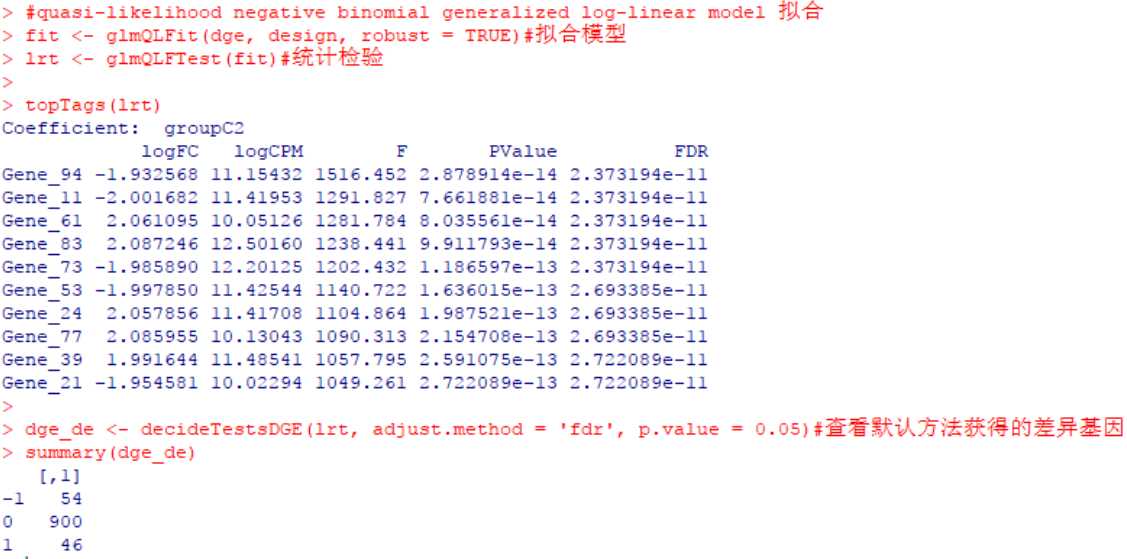
dge\_de <- decideTestsDGE(lrt, adjust.method = 'fdr', p.value = 0.05) #查看默认方法获得的差异基因

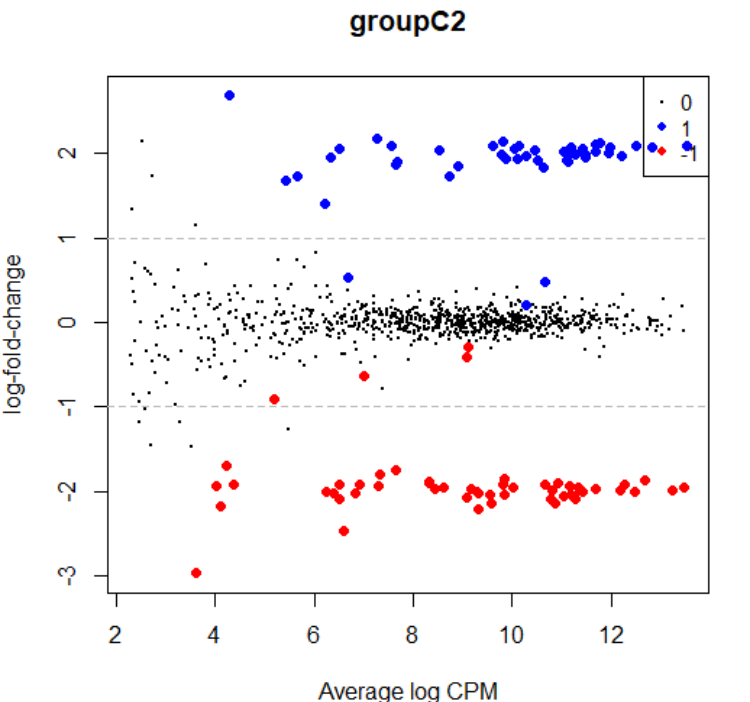
summary(dge\_de)

plotMD(lrt, status = dge\_de, values = c(1, -1), col = c('blue', 'red')) #作图观测

abline(h = c(-1, 1), col = 'gray', lty = 2)

结果内容说明可参见上文。





**配对检验**

除了拟合模型的方法外，在edgeR中还可使用exactTest()直接执行两组负二项分布count之间基因均值差异的精确检验。不知道怎么翻译，就称它为配对检验好了。

#配对检验，详情参考 ?exactTest

dge\_et <- exactTest(dge) #检验

topTags(dge\_et)

write.csv(topTags(dge\_et, n = nrow(dgelist$counts)), 'exactTest.csv', quote = FALSE) #输出主要结果

dge\_de <- decideTestsDGE(dge\_et, adjust.method = 'fdr', p.value = 0.05) #查看默认方法获得的差异基因

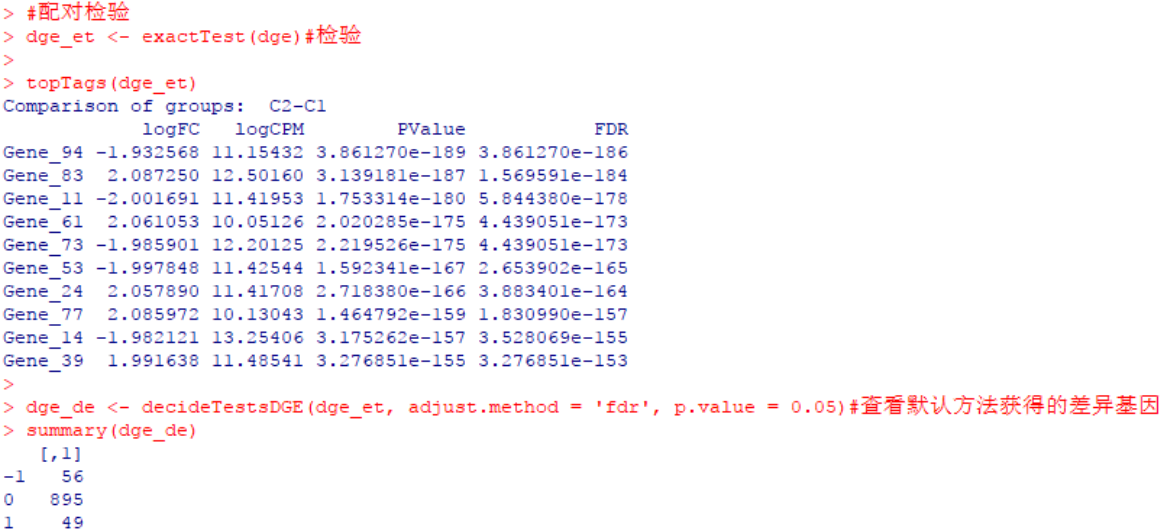
summary(dge\_de)

detags <- rownames(dge)[as.logical(dge\_de)]

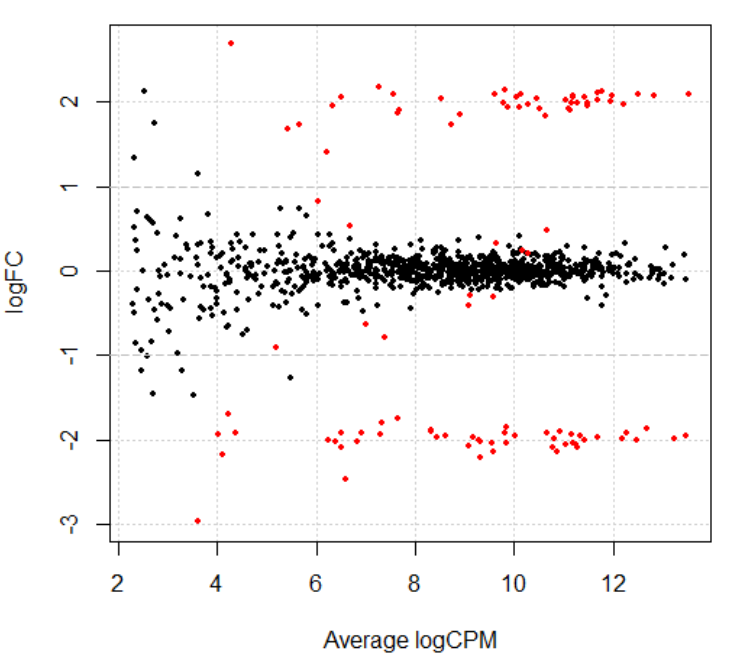
plotSmear(dge\_et, de.tags = detags, cex = 0.5) #作图观测

abline(h = c(-1, 1), col = 'gray', lty = 2)

同样地，结果内容说明可参见上文。



因limma包的plotMD()函数无法在此处适用，这里使用的作图函数plotSmear()是edgeR包中的方法，详见?plotSmear说明。同样地，下图中纵轴为log2 Fold Change值；横轴为log2 CPM值，反映了基因表达量信息；红色的点表示差异基因（未使用颜色进一步区分上调/下调），黑色的点为无差异基因。



**Voom标准化及线性建模（limma）**

上文多次使用到limma包中的一些命令。毫无疑问，limma包作为处理RNA-seq数据的最流行的R包之一，同样提供了多种差异基因分析的方法，其中最常用的就是“voom”。由于limma和edgeR二者的很多功能相辅相成，尽管limma也有自己的一套分析思路（本文不再详细说明了，有兴趣的同学可以自行研究），不过我们仍可以基于上文前四步获得的预处理结果（DGEList对象、原始的非标准化数据、估算的离散值等），继续使用limma包“voom方法”来完成后续的差异基因分析。

将read count数据转换为log2-counts per million（logCPM），通过估计均值-方差（mean-variance）关系并使用它来计算合适的observation-level weights，然后，数据就可以进行线性建模。

#voom 线性建模（limma）

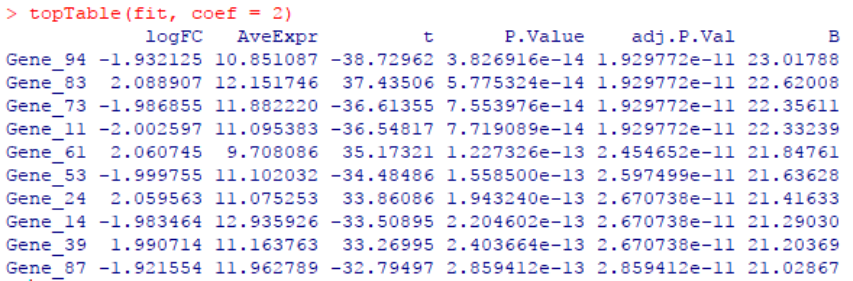
limma\_voom <- voom(dgelist, design, plot = TRUE)

fit <- lmFit(limma\_voom, design) #拟合

fit <- eBayes(fit)

topTable(fit, coef = 2)

write.csv(topTable(fit, coef = 2, number = nrow(dgelist$counts)), 'limma\_voom.csv', quote = FALSE) #输出主要结果



**小结**

总之，无论数据过滤、标准化、或者差异分析等，可供选择的方法很多。实际分析中具体选择哪些，确实无法给出绝对的标准，还需结合所关注的生物学问题、预期的结论等，多加尝试了。或者，分别尝试不同的方法，并对多种方法的结果取交集等，也是一个不错的选择。