Likelihood ratio test and quasi-likelihood F-test for detecting changed genes

Aleksandra Grudziąż

19 kwietnia 2018

Data

To prepare the following tables with counts, I used the Araport database, which is a more accurate version of the TAIR10 database, which I used in my previous analyzes. This database better assigns parts to the corresponding genes. In addition, counts take into account the distance between 300b and the longer 3'UTR for each gene we have more than 5 counts. If so, these counts are also counted as readings for the gene. In the analysis below, we look for differences between the wild subtype and the brm mutant.

```
load("countsAll_new.rda")
```

```
library(directRNAExplorer)
library(dplyr)
```

Testing diffrences in genes

As before, before testing we have to choose the "appropriate" genes, that is, those for whom the quality of sequencing satisfies us. We assumed that for wt a good sequencing for the gene is where at least 2 out of 3 samples we have more than 9 readings, and for mutations brm if at least 3 out of 4 samples we have more than 9 readings. In addition, we are looking for genes that have many counts in one of the subtypes and hardly any at all.

```
genesTest <-c()
countsWithCondition <- countsAll_new


for(i in 2:ncol(countsAll)){
  countsWithCondition[,i] <- ifelse(countsAll[,i]>=9, 1, 0)
}

allCounts_2 <- countsWithCondition[,c(1:8)]
genes <- allCounts_2[,1]

allCountsTransposed <- t(allCounts_2[,-1])
allCountsTransposed <- data.frame(allCountsTransposed)
colnames(allCountsTransposed)<- genes

allCountsTransposed$type <- c(rep("wt", 3), rep("brm", 4))
allCountsTransposed <- allCountsTransposed[,c(ncol(allCountsTransposed), 1:(ncol(allCountsTransposed)-1

sums <- aggregate(. ~ type, data=allCountsTransposed, FUN=sum)

for(i in 2:ncol(allCountsTransposed)){
  if(((sums[1,i]>=3) & (sums[2,i]>=2)) || ((sums[1,i]>=3) & (sums[2,i]==0))||((sums[1,i]==0) & (sums[2,
```

```
    genesTest[i] <- colnames(allCountsTransposed)[i]
  }else{
    genesTest[i] <- NA
  }
}

genesTest2<- na.omit(genesTest)

countsTest <- countsAll_new[,c(1:8)]
countsTest <- countsTest[which(countsTest$name %in% genesTest2),]

countsTest2 <- data.frame(t(countsTest[,-1]))
colnames(countsTest2) <- countsTest$name
```

Likelihood ratio test

For selected genes we perform a likelihood ratio test, in order to find genes for which the number of readings has changed significantly divided into two groups - wt and brm. The `testEdger()` is based on `edgeR` R package. We choose the `type` "lrt"" to perform likelihood ratio test.

```
cond  <- c(rep("wt", 3), rep("brm", 4))
test<-edgerTest(countsTest2, condition = cond, type="lrt")
```

To find the most changed genes, we will focus only on those that have a small pvalue and fold greater than 1.6.

```
library(dplyr)
testFiltered <- filter(test, pvalue<0.05)
testFiltered <- filter(testFiltered, abs(log2FoldChange)>0.32)
testFiltered <- arrange(testFiltered, pvalue)
testFiltered <- arrange(testFiltered, desc(log2FoldChange))

head(testFiltered, 8)
```

```
##   log2FoldChange   logCPM        LR       pvalue        id
## 1       5.450000 8.894925 11.661442 6.380886e-04 AT3G27700
## 2       5.106717 9.030065 12.863734 3.350115e-04 AT4G28740
## 3       4.345750 7.210490 12.027947 5.240878e-04 AT4G16710
## 4       4.226212 5.912666 91.647379 1.035830e-21 AT3G01345
## 5       3.976154 7.673624  9.694307 1.848398e-03 AT4G33000
## 6       3.707366 5.058775 12.833371 3.404918e-04 AT2G44850
## 7       3.697730 8.318383 10.154933 1.439156e-03 AT2G21320
## 8       3.247221 7.173796  6.242492 1.247209e-02 AT5G66180
```

```
nrow(testFiltered)
```

```
## [1] 977
```

Quasi-likelihood F-test

As before, we use the `testEdger()` function to find differences in genes, but in this case we use the "qlt" value for the `type` argument.

```
cond  <- c(rep("wt", 3), rep("brm", 4))
test<-edgerTest(countsTest2, condition = cond, type="qlf")
```

To find the most changed genes, we will focus only on those that have a small pvalue and fold greater than 1.6.

```
library(dplyr)
testFiltered <- filter(test, pvalue<0.05)
testFiltered <- filter(testFiltered, abs(log2FoldChange)>0.32)
testFiltered <- arrange(testFiltered, pvalue)
testFiltered <- arrange(testFiltered, desc(log2FoldChange))

head(testFiltered, 8)
```

```
##   log2FoldChange   logCPM          F       pvalue        id
## 1       5.438035 8.894925   9.837701 1.499784e-02 AT3G27700
## 2       5.102942 9.030065  11.256455 1.093759e-02 AT4G28740
## 3       4.328831 7.210490  11.000285 1.155626e-02 AT4G16710
## 4       4.268851 5.912666 121.084082 6.628767e-06 AT3G01345
## 5       3.999903 7.673624   8.792226 1.928688e-02 AT4G33000
## 6       3.709251 8.318383   9.577522 1.594091e-02 AT2G21320
## 7       3.691858 5.058775  12.812294 7.961881e-03 AT2G44850
## 8       3.249631 7.173796   5.707854 4.581513e-02 AT5G66180
```

```
nrow(testFiltered)
```

```
## [1] 1045
```