# Dirichlet-multinomial framework for differential splicing and sQTL analysis in RNA-seq.

Malgorzata Nowicka*, Mark Robinson

October 13, 2015

This vignette describes version 0.3.1 of the *DRIMSeq* package.

## Contents



# 1 Overview of Dirichlet-multinomial model

# 2 Differential splicing analysis work-flow

## 2.1 Example data

FASTQ files can be downloaded from ... (See Charlotte's paper) Reference genome files from ...

## 2.2 RNA-seq data alignment and counting

Step depend on which counts you want to use:

Exonic bin counts from HTSeq/DEXSeq, featureCounts

Transcript counts from kallisto, RSEM, BitSeq

*gosia.nowicka@uzh.ch

(Reference to "Count-based differential expression...", pasilla package)

## 2.3 Differential splicing analysis with DM package

Assuming that you have a table with feature counts...

Load *DRIMSeq* package.

```
library(DRIMSeq)
```

```
## Loading required package:  BiocGenerics
## Loading required package:  parallel
##
## Attaching package:  'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
##    clusterMap, parApply, parCapply, parLapply, parLapplyLB, parRapply,
##    parSapply, parSapplyLB
##
## The following object is masked from 'package:stats':
##
##    xtabs
##
## The following objects are masked from 'package:base':
##
##    anyDuplicated, append, as.data.frame, as.vector, cbind, colnames, do.call,
##    duplicated, eval, evalq, Filter, Find, get, intersect, is.unsorted, lapply,
##    Map, mapply, match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##    Position, rank, rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
##    table, tapply, union, unique, unlist, unsplit
##
## Creating a generic function for 'nchar' from package 'base' in package 'S4Vectors'
```

Create a `dmDSdata` object that contains counts.

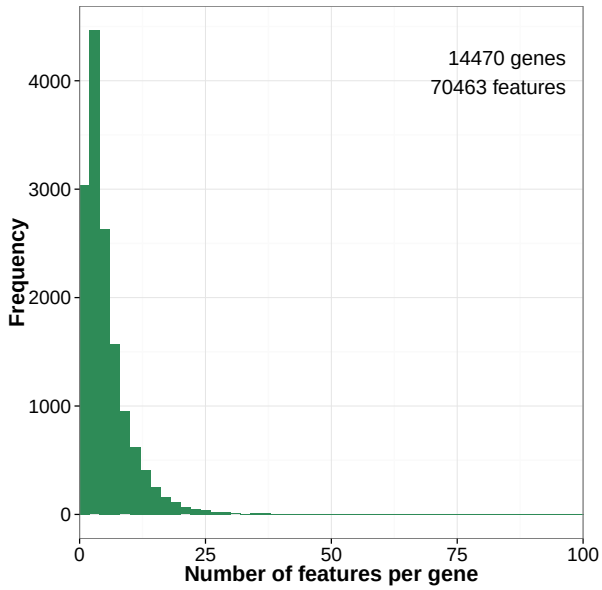
```
library(pasilla)
data_dir <- system.file("extdata", package = "pasilla")
count_files <- list.files(data_dir, pattern = "fb.txt$", full.names = TRUE)
count_files
```

```
## [1] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/treated1fb.txt"
## [2] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/treated2fb.txt"
## [3] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/treated3fb.txt"
## [4] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/untreated1fb.txt"
## [5] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/untreated2fb.txt"
## [6] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/untreated3fb.txt"
## [7] "/Users/gosia/Library/R/3.2/library/pasilla/extdata/untreated4fb.txt"
```

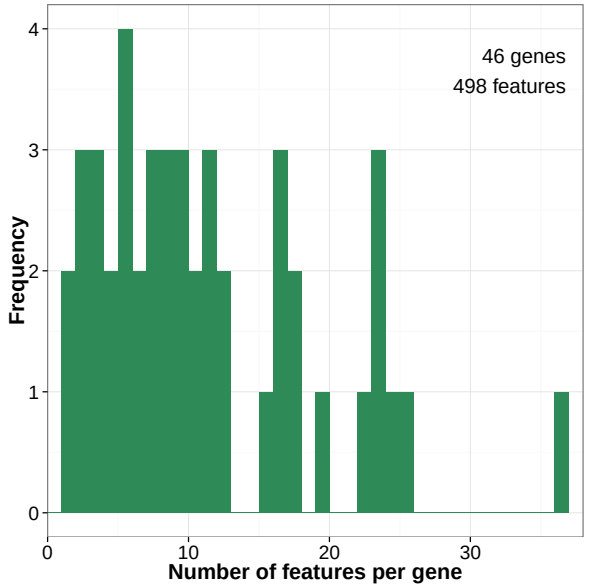
```
# Create a data frame with htseq counts
htseq_list <- lapply(1:length(count_files), function(i) {
    # i = 1
    htseq <- read.table(count_files[i], header = FALSE, as.is = TRUE)
    colnames(htseq) <- c("group_id", gsub("fb.txt", "", strsplit(count_files[i],
        "extdata/")[[1]][2]))
    return(htseq)
})
htseq_counts <- Reduce(function(...) merge(..., by = "group_id",
    all = TRUE, sort = FALSE), htseq_list)
tail(htseq_counts)
```

```
##              group_id treated1 treated2 treated3 untreated1 untreated2 untreated3
## 70462 FBgn0261575:001        2        0        0          0          1          0
## 70463 FBgn0261575:002       10        1        3          5          7          1
## 70464      _ambiguous     1317        0        0       1096       1144          0
## 70465          _empty 11173758 16686814 16057634    7756745   12150698   13990064
## 70466       _lowaqual 67756093        0        0   22017200   42760359          0
## 70467     _notaligned        0        0        0          0          0          0
##       untreated4
## 70462          0
## 70463          0
## 70464          0
## 70465   14248758
## 70466          0
## 70467          0
```

```
htseq_counts <- htseq_counts[!grepl(pattern = "_", htseq_counts$group_id),
    ]
group_split <- limma::strsplit2(htseq_counts[, 1], ":")
d <- dmDSdata(counts = htseq_counts[, -1], gene_id = group_split[,
    1], feature_id = group_split[, 2], sample_id = colnames(htseq_counts)[-1],
    group = gsub("[1-4]", "", colnames(htseq_counts)[-1]))
plotData(d)
```

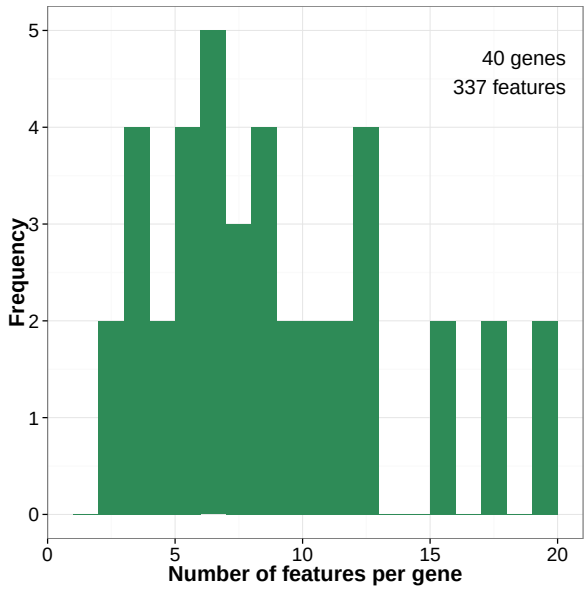



```
# Use a subset of genes, which is defined in the following
# file
genes_subset = readLines(file.path(data_dir, "geneIDsinsubset.txt"))
d <- d[names(d) %in% genes_subset, ]
plotData(d)
```




Filter genes and transcripts with low expression.

```
d <- dmFilter(d)
plotData(d)
```




Estimate dispersion.

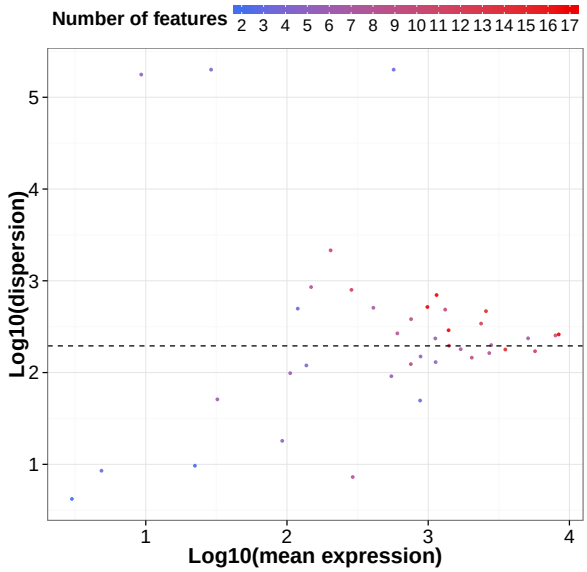
```
d <- dmDispersion(d, BPPARAM = BiocParallel::MulticoreParam(workers = 2))
```

```
## * Calculating mean gene expression..
## Took  0.519  seconds.
## * Estimating common dispersion..
## Took  29.42  seconds.
```

```
## !  Using common_dispersion = 195.27 as disp_init !
```

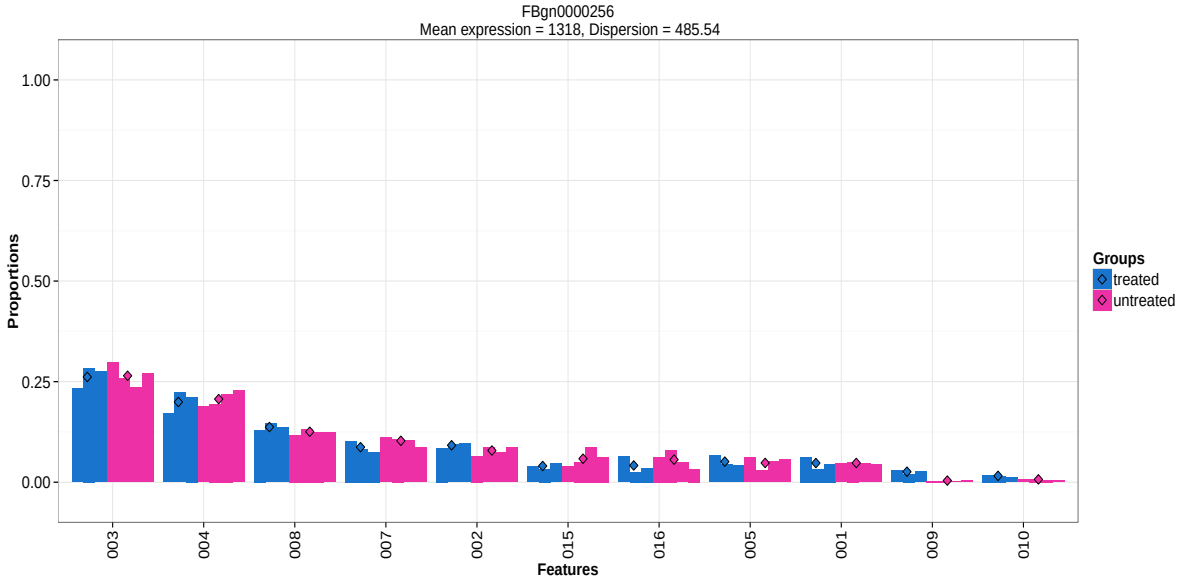
```
## * Estimating genewise dispersion..
## Took  4.431  seconds.
```

```
plotDispersion(d)
```



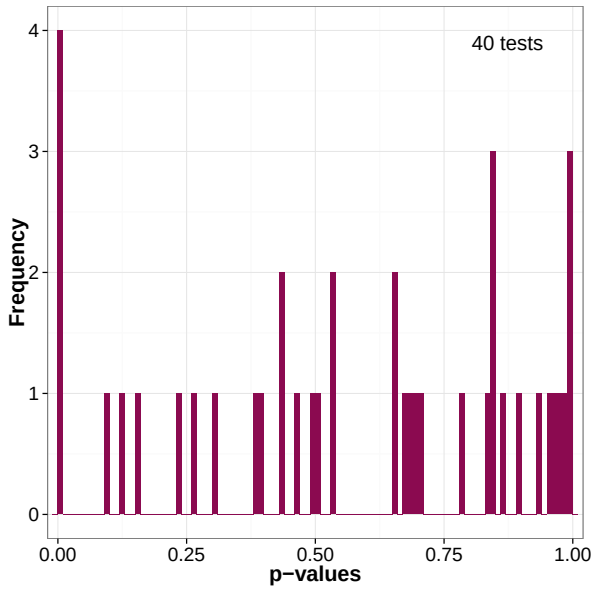
Estimate proportions.

```
d <- dmFit(d)

## * Fitting full model..
## Took  10.841  seconds.

gene_id <- names(d)[1]
plotFit(d, gene_id = gene_id, plot_type = "barplot")

## Plot gene 1: FBgn0000256
```




Use likelihood ratio to test for differential splicing.

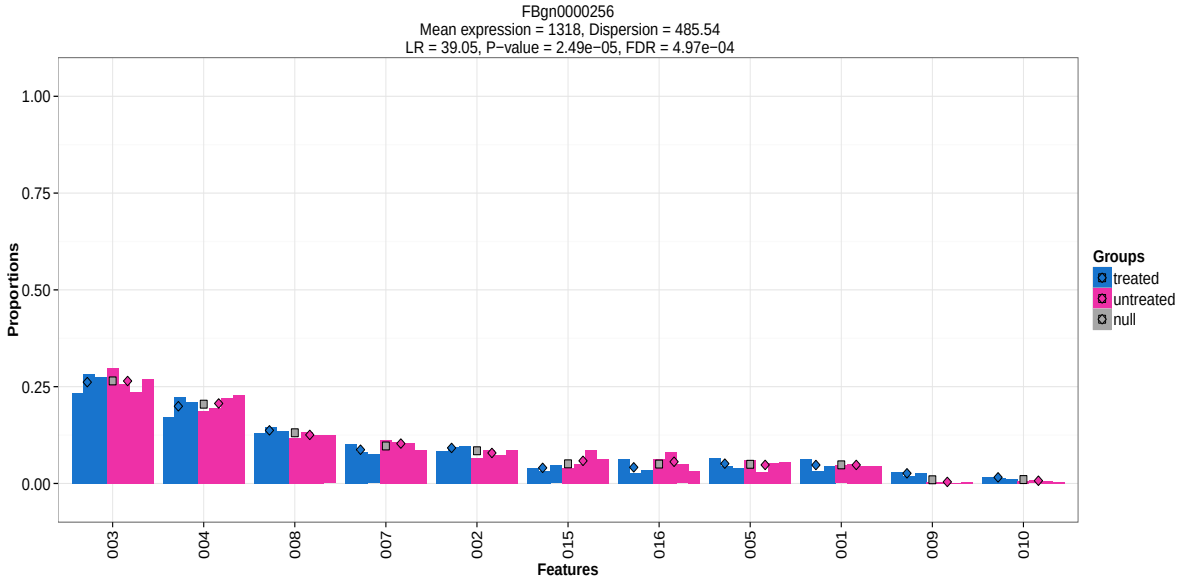
```
d <- dmTest(d)

## Running comparison  between groups:  treated, untreated

## * Fitting null model..
## Took  10.879  seconds.
## * Calculating likelihood ratio statistics..
## Took  0.003160954  seconds.

plotTest(d)
```




```
res <- results(d)
```

```
gene_id <- res$gene_id[1]
plotFit(d, gene_id = gene_id)
```

```
## Plot gene 1: FBgn0000256
```




# 3 sQTL analysis work-flow

## 3.1 Example data

Data downloaded from

## 3.2 Extracting the bi-allelic SNPs from CSV files

Preprocessing steps on CSV files with genotypes

## 3.3 sQTL analysis with DM package

Assuming you have counts and bi-allelic genotypes...