BCGES short courses, session 7
RNA-Seq

Vincent Plagnol

1 GTF format to store information about genes

1.1 Ensembl data

If one works with genes and exons, it is important to have a format that captures this information. The file format that does this is the GTF format. A good place to download GTF file is the http://www.ensembl.org/info/data/ftp/index.html. One can start by using the `curl` function (which is a combination of `cat` and `url`) to obtain the first few lines of an example GTF file.

```
curl --silent ftp://ftp.ensembl.org/pub/release-76/gtf/homo_sapiens/Homo_sapiens.GRCh38.76.gtf.gz | \
   zcat | head -100 > results/human_gtf_example.gtf
```

Exercise: Go over the GTF format and understand what the fields mean, and how the data are organised.

You can now download the full ensembl file to get an idea of the size of the file. We will use the wget function that was used before in these practicals (note that the code below is not executed, because too long to go through).

```
wget -O results/ensembl_human_GRCh38.gtf.gz \
   ftp://ftp.ensembl.org/pub/release-76/gtf/homo_sapiens/Homo_sapiens.GRCh38.76.gtf.gz
```

1.2 UCSC data

UCSC is the other obvious place to obtain genome-scale data. The webpage you want to become familiar with is this one.

Exercise: Look for a human GTF file generally equivalent to the one you just downloaded from UCSC. Compare the sizes of both files, look for differences and similarities.

```
#You want to set the group option \textt{mRNA and EST}.
#Use the Human mRNA track
#In output format select \texttt{GTF - gene transfer format}
#Specify a name in the output file
#Maybe request a compressed file to limit transfer time
#Compressed the UCSC GTF is ... and the compressed Ensembl one is 16M.
```

2 Aligning RNA-Seq data

3 Library normalization choices

4 Differential expression analysis

We start by loading the DESeq package as well as an example dataset from a mouse brain RNA-Seq experiment.

```
library(DESeq)

## Loading required package:  BiocGenerics
## Loading required package:  methods
## 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,
##    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
##
## Loading required package:  Biobase
## Welcome to Bioconductor
##
##    Vignettes contain introductory material; view with
##    'browseVignettes()'.  To cite Bioconductor, see
##    'citation("Biobase")', and for packages 'citation("pkgname")'.
##
## Loading required package:  locfit
## locfit 1.5-9.1 2013-03-22
## Loading required package:  lattice
##    Welcome to 'DESeq'.  For improved performance, usability and
##    functionality, please consider migrating to 'DESeq2'.
```

```
load("../data/RNASeq/deseq_counts_TDP43.RData")
head(genes.counts)
```

```
##                    control_rep1_dexseq_counts.txt
## ENSMUSG00000000001                            208
## ENSMUSG00000000003                              0
## ENSMUSG00000000028                             15
## ENSMUSG00000000037                              9
## ENSMUSG00000000049                              4
## ENSMUSG00000000056                            233
##                    control_rep2_dexseq_counts.txt
## ENSMUSG00000000001                            295
## ENSMUSG00000000003                              0
## ENSMUSG00000000028                             26
## ENSMUSG00000000037                             20
## ENSMUSG00000000049                              1
## ENSMUSG00000000056                            390
##                    control_rep3_dexseq_counts.txt
## ENSMUSG00000000001                            239
## ENSMUSG00000000003                              0
## ENSMUSG00000000028                             13
## ENSMUSG00000000037                             13
## ENSMUSG00000000049                              3
## ENSMUSG00000000056                            346
##                    control_rep4_dexseq_counts.txt KD_rep1_dexseq_counts.txt
## ENSMUSG00000000001                            292                       326
## ENSMUSG00000000003                              0                         1
```

```
## ENSMUSG00000000028                            13                          21
## ENSMUSG00000000037                            21                          11
## ENSMUSG00000000049                             2                           2
## ENSMUSG00000000056                           381                         339
##                    KD_rep2_dexseq_counts.txt KD_rep3_dexseq_counts.txt
## ENSMUSG00000000001                       371                       316
## ENSMUSG00000000003                         0                         0
## ENSMUSG00000000028                        22                        18
## ENSMUSG00000000037                        12                        18
## ENSMUSG00000000049                         1                         1
## ENSMUSG00000000056                       359                       317
##                    KD_rep4_dexseq_counts.txt
## ENSMUSG00000000001                       339
## ENSMUSG00000000003                         0
## ENSMUSG00000000028                        18
## ENSMUSG00000000037                        30
## ENSMUSG00000000049                         2
## ENSMUSG00000000056                       379
```

We can now define the model for the differential expression analysis:

```
formula1 <- count ~ condition
formula0 <- count ~ 1
design.deseq <- c('control', 'control', 'control', 'control', 'KD', 'KD', 'KD', 'KD')
```

And now the computations can properly start. Note that these steps are very long, and therefore the code is not executed as part of this file (to be more precise, it is executed once, and the output is saved).

```
CDS <- newCountDataSet(genes.counts, condition = design.deseq)

CDS <- estimateSizeFactors(CDS)
CDS <- estimateDispersions(CDS, method = 'pooled')

fit0 <- fitNbinomGLMs( CDS, formula0 )

## ..................

fit1 <- fitNbinomGLMs( CDS, formula1 )

## ..................

deseq.pval <- fit1
deseq.pval$EnsemblID <- row.names( deseq.pval)
deseq.pval$basic.pval <- signif(nbinomGLMTest( fit1, fit0 ), 4)
save(list = 'deseq.pval', file = 'results/DE_pvalues_ranked.RData')
```

See below some polishing: a multiple testing/false discovery rate Bonferroni-Hochberg analysis, and the ordering of the results by significance of P-values.

```
load('results/DE_pvalues_ranked.RData')
deseq.pval$adj.pval <- signif(p.adjust( deseq.pval$basic.pval, method="BH" ), 4)

deseq.pval <- deseq.pval[ order(deseq.pval$basic.pval, decreasing = FALSE), ]
head(deseq.pval)

##                    (Intercept) conditionKD deviance converged
## ENSMUSG00000023224       5.788       1.733    4.110      TRUE
## ENSMUSG00000023826       6.559      -2.002    7.912      TRUE
## ENSMUSG00000026547       6.032       1.688    3.296      TRUE
## ENSMUSG00000039419       9.698      -1.185   12.093      TRUE
## ENSMUSG00000040424       8.450      -1.373    6.263      TRUE
```

```
## ENSMUSG00000041459      10.080      -1.691    5.526      TRUE
##                            EnsemblID basic.pval adj.pval
## ENSMUSG00000023224 ENSMUSG00000023224          0        0
## ENSMUSG00000023826 ENSMUSG00000023826          0        0
## ENSMUSG00000026547 ENSMUSG00000026547          0        0
## ENSMUSG00000039419 ENSMUSG00000039419          0        0
## ENSMUSG00000040424 ENSMUSG00000040424          0        0
## ENSMUSG00000041459 ENSMUSG00000041459          0        0
```