Hands on

Kasper Daniel Hansen

March 31, 2012

1 Mapping using Bowtie

The `yeastRNASeq` package contains a number of FASTQ files from an experiment in *S. Cerevisiae*. We will use `wt_1_f.fastq`.

1. Look at the FASTQ file and get a feel for the file format. How many reads does it contains, how long are they?
2. Use Bowtie to map these reads against the yeast genome. Look at the output. For example `bowtie s_cerevisiae wt_1_f.fastq > wt_1_f.bowtie`. You can get a list of command line arguments just by typing `bowtie`.
3. Use the R package `ShortRead` to read the bowtie ouput into R. For example

```
> library(ShortRead)
> wt.bwt <- readAligned("wt_1_f.bowtie", type = "Bowtie")
> wt.bwt

class: AlignedRead
length: 410349 reads; width: 26 cycles
chromosome: Scchr11 Scchr04 ... Scchr05 Scchr10
position: 283246 961713 ... 195595 218358
strand: + - ... + +
alignQuality: NumericQuality
alignData varLabels: similar mismatch
```

2 Differential Expression

We will do a simple differential expression analysis using a small number of biological replicates. This is based on the Montgomery dataset, and we have arbitrarily divided the samples into two groups.

```
> library(cqn)
> library(edgeR)
> data(montgomery.subset)
> data(uCovar)
```

The number of reads generated for each sample is

```
> data(sizeFactors.subset)
> round(sizeFactors.subset / 10^6, 2)
```

```
NA06985 NA06994 NA07037 NA10847 NA11920 NA11918 NA11931 NA12003 NA12006 NA12287
   3.11    2.39    3.09    2.85    4.77    1.67    5.53    4.22    7.01    4.15
```

First we import this data into a `DGE-list` which is a special `edgeR` object type

```
> d.mont <- DGEList(counts = montgomery.subset, lib.size = sizeFactors.subset,
+                   group = rep(c("grp1", "grp2"), each = 5))
```

You can try different scaling normalizations using `calcNormFactors` (default is TMM).

The first step in an `edgeR` analysis is to estimate the dispersion parameter. This can be done in a variety of ways.

```
> design <- model.matrix(~ d.mont$sample$group)
> d.mont.std <- estimateGLMCommonDisp(d.mont, design = design)
```

With a dispersion parameter we can do a likelihood ratio test and find significant genes

```
> efit.std <- glmFit(d.mont.std, design = design)
> elrt.std <- glmLRT(d.mont.std, efit.std, coef = 2)
> topTags(elrt.std)
```

```
Coefficient:  d.mont$sample$groupgrp2
                        logFC    logCPM        LR       PValue          FDR
ENSG00000211642 -1.183042e+01  6.354651 126.70309 2.157532e-29 5.081420e-25
ENSG00000211660 -1.145797e+01  5.987328 120.11380 5.973379e-28 7.034251e-24
ENSG00000211890 -7.611397e+00 10.064744  99.84754 1.645917e-23 1.292154e-19
ENSG00000211937 -7.178811e+00  5.876130  86.28142 1.560680e-20 9.189281e-17
ENSG00000211638  7.821798e+00  4.879239  81.97274 1.379769e-19 6.499266e-16
ENSG00000243063 -7.321814e+00  5.650004  80.75488 2.555257e-19 1.003023e-15
ENSG00000211651  7.690969e+00  3.676676  68.94079 1.014645e-16 3.413847e-13
ENSG00000238649  6.754660e+00  4.093400  66.22103 4.030884e-16 1.186692e-12
ENSG00000224373 -1.442695e+08  2.070904  60.43882 7.590126e-15 1.986252e-11
ENSG00000253822  1.442695e+08  2.477550  59.99290 9.520032e-15 2.242158e-11
```

Take a look at the raw data for the top hit.

We now try to normalize using `cqn`.

```
> cqn.subset <- cqn(montgomery.subset, lengths = uCovar$length,
+                   x = uCovar$gccontent, sizeFactors = sizeFactors.subset,
+                   verbose = TRUE)

RQ fit ..........
SQN .
```

The return object has an `offset` component we can just pass into the `edgeR` functions

```
> d.mont.cqn <- estimateGLMCommonDisp(d.mont, design = design, offset = cqn.subset$off
> efit.cqn <- glmFit(d.mont.cqn, design = design, offset = cqn.subset$offset)
> elrt.cqn <- glmLRT(d.mont.cqn, efit.cqn, coef = 2)
> topTags(elrt.cqn)

Coefficient:  d.mont$sample$groupgrp2
                       logFC   logCPM       LR       PValue          FDR
ENSG00000211890 -1.090978e+01 34.81142 66.02394 4.454782e-16 6.720475e-12
ENSG00000211660 -1.292733e+01 27.98303 65.53575 5.706925e-16 6.720475e-12
ENSG00000211642 -1.249294e+01 27.27726 61.95841 3.507886e-15 2.753924e-11
ENSG00000243063 -8.751474e+00 27.48189 47.26079 6.214294e-12 3.658976e-08
ENSG00000253822  1.442695e+08 23.84259 43.10741 5.181566e-11 2.223974e-07
ENSG00000211937 -7.941604e+00 27.37098 42.93266 5.665695e-11 2.223974e-07
ENSG00000238649  8.560910e+00 22.37745 41.36862 1.260646e-10 4.241535e-07
ENSG00000211651  8.498505e+00 25.95761 37.39140 9.664744e-10 2.759233e-06
ENSG00000211892 -8.174582e+00 25.01262 37.22160 1.054395e-09 2.759233e-06
ENSG00000211958  7.998552e+00 24.35146 34.06416 5.332435e-09 1.255895e-05
```

3 SessionInfo

- R version 2.14.1 Patched (2012-02-09 r58307), `x86_64-apple-darwin10.8.0`
- Locale: `en_US.utf-8/en_US.utf-8/en_US.utf-8/C/en_US.utf-8/en_US.utf-8`
- Base packages: base, datasets, graphics, grDevices, methods, splines, stats, utils
- Other packages: Biostrings 2.22.0, cqn 1.0.1, edgeR 2.4.6, GenomicRanges 1.6.7, IRanges 1.12.6, lattice 0.20-6, latticeExtra 0.6-19, limma 3.10.3, mclust 3.4.11, nor1mix 1.1-3, preprocessCore 1.16.0, quantreg 4.77, RColorBrewer 1.0-5, Rsamtools 1.6.3, ShortRead 1.12.4, SparseM 0.96

- Loaded via a namespace (and not attached): Biobase 2.14.0, bitops 1.0-4.1, BSgenome 1.22.0, grid 2.14.1, hwriter 1.3, RCurl 1.91-1, rtracklayer 1.14.4, tools 2.14.1, XML 3.9-4, zlibbioc 1.0.1