segmentSeq: methods for detecting methylation loci and differential methylation

Thomas J. Hardcastle

August 4, 2015

1 Introduction

This vignette introduces analysis methods for data from high-throughput sequencing of bisulphite treated DNA to detect cytosine methylation. The segmentSeq package was originally designed to detect siRNA loci [1] and many of the methods developed for this can be used to detect loci of cytosine methylation from replicated (or unreplicated) sequencing data.

2 Preparation

Preparation of the segmentSeq package proceeds as in siRNA analysis. We begin by loading the segmentSeq package.

```
>   library(segmentSeq)
```

Note that because the experiments that segmentSeq is designed to analyse are usually massive, we should use (if possible) parallel processing as implemented by the parallel package. If using this approach, we need to begin by define a *cluster*. The following command will use eight processors on a single machine; see the help page for 'makeCluster' for more information. If we don't want to parallelise, we can proceed anyway with a NULL cluster. Results may be slightly different depending on whether or not a cluster is used owing to the non-deterministic elements of the method.

```
> if(require("parallel"))
+ {
+     numCores <- min(8, detectCores())
+     cl <- makeCluster(numCores)
+ } else {
+     cl <- NULL
+ }
```

The segmentSeq package is designed to read in output from the YAMA (Yet Another Methylome Aligner) program. This is a perl-based package using either bowtie or bowtie2 to align bisulphite treated reads (in an unbiased manner) to a reference and identify the number of times each cytosine is identified as methylated or unmethylated. Unlike most other aligners, YAMA does not require that reads that map to more than one location are discarded, instead it reports the number of alternate matches to the reference for each cytosine. This is then used by segmentSeq to weight the observed number of methylated/un-methylated cytosines at a location.

```
> datadir <- system.file("extdata", package = "segmentSeq")
> files <- c("short_18B_C24_C24_trim.fastq_CG_methCalls",
+ "short_Sample_17A_trimmed.fastq_CG_methCalls",
+ "short_13_C24_col_trim.fastq_CG_methCalls",
+ "short_Sample_28_trimmed.fastq_CG_methCalls")
> mD <- readMeths(files = files, dir = datadir,
+ libnames = c("A1", "A2", "B1", "B2"), replicates = c("A","A","B","B"),
+ nonconversion = c(0.004777, 0.005903, 0.016514, 0.006134))
```

We can begin by plotting the distribution of methylation for these samples. The distribution can be plotted for each sample individually, or as an average across multiple samples. We can also subtract one distribution from another to visualise patterns of differential methylation on the genome.

```
> par(mfrow = c(2,1))
> dists <- plotMethDistribution(mD, main = "Distributions of methylation", chr = "Chr1")
> plotMethDistribution(mD, subtract = rowMeans(sapply(dists, function(x) x[,2])), main = "Differences b
```

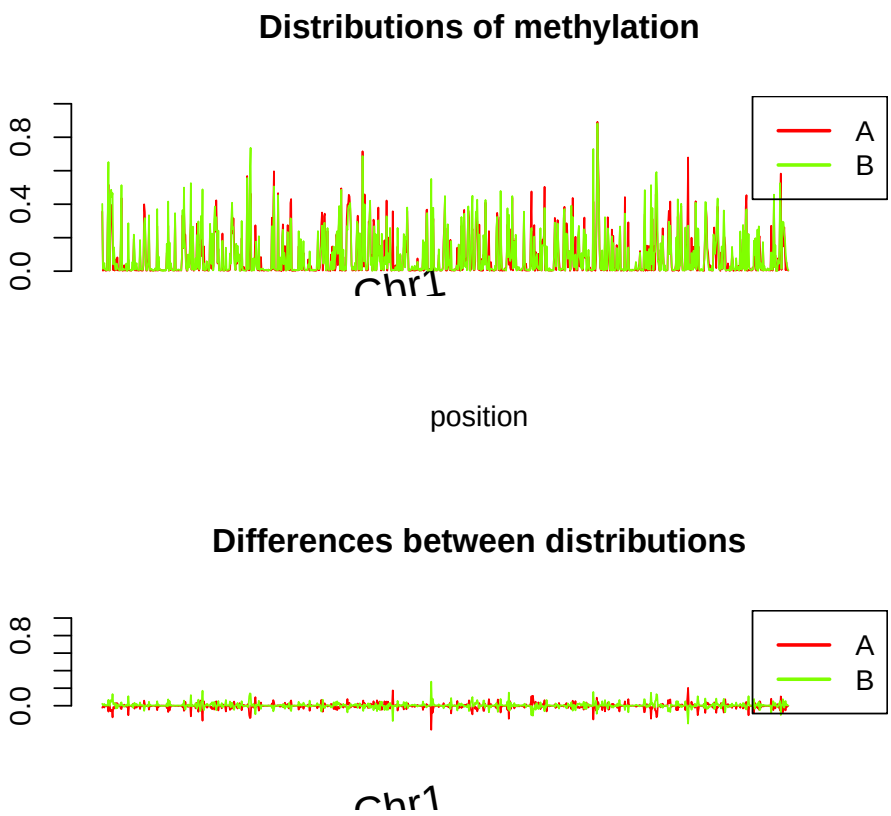

Figure 1: Distributions of methylation on the genome (first two million bases of chromosome 1.

Next, we process this `alignmentData` object to produce a `segData` object. This `segData` object contains a set of potential segments on the genome defined by the start and end points of regions of overlapping alignments in the `alignmentData` object. It then evaluates the number of tags that hit in each of these segments.

```
> sD <- processAD(mD, gap = 300, squeeze = 10, filterProp = 0.05, verbose = TRUE, strandSplit = TRUE, c
```

We can now construct a segment map from these potential segments.

Segmentation by heuristic Bayesian methods

A fast method of segmentation can be achieved by assuming a binomial distribution on the data with an uninformative beta prior, and identifying those potential segments which have a sufficiently large posterior likelihood that the proportion of methylation exceeds some critical value.

```
> hS <- heuristicSeg(sD = sD, aD = mD, prop = 0.2, cl = cl, gap = 100, getLikes = FALSE)
> hS
```

```
Slot "coordinates"
GRanges object with 2317 ranges and 0 metadata columns:
         seqnames             ranges strand
            <Rle>          <IRanges>  <Rle>
     [1]     Chr1       [ 108,  948]      +
     [2]     Chr1       [ 150,  837]      -
     [3]     Chr1       [5226, 5465]      +
     [4]     Chr1       [6452, 6452]      +
     [5]     Chr1       [7021, 7021]      -
     ...      ...                ...    ...
  [2313]     Chr1 [1990298, 1990298]      +
  [2314]     Chr1 [1993118, 1993149]      +
  [2315]     Chr1 [1993335, 1993335]      +
  [2316]     Chr1 [1994611, 1994611]      +
  [2317]     Chr1 [1994857, 1994886]      +
  -------
  seqinfo: 1 sequence from an unspecified genome; no seqlengths
An object of class "methData"
2317 rows and 4 columns

Slot "replicates"
A A B B
Slot "groups":
list()

Slot "data":
     A.1   A.2   B.1   B.2
[1,] NA:NA NA:NA NA:NA NA:NA
[2,] NA:NA NA:NA NA:NA NA:NA
[3,] NA:NA NA:NA NA:NA NA:NA
[4,] NA:NA NA:NA NA:NA NA:NA
[5,] NA:NA NA:NA NA:NA NA:NA
2312 more rows...

Slot "annotation":
data frame with 0 columns and 2317 rows

Slot "locLikelihoods" (stored on log scale):
Matrix with  2317  rows.
       A   B
1      1   1
2      1   1
3      0   1
4      1   0
5      0   1
...  ... ...
2313   1   1
2314   1   1
2315   1   0
2316   1   0
2317   0   1
```

Within a methylation locus, it is not uncommon to find completely unmethylated cytosines. If the coverage of these cytosines is too high, it is possible that these will cause the locus to be split into two or more fragments. The mergeMethSegs function can be used to overcome this splitting by merging loci with identical patterns of expression that are not separated by too great a gap. Merging in this manner is optional, but recommended.

```
> hS <- mergeMethSegs(hS, mD, gap = 5000, cl = cl)
```

We can then estimate posterior likelihoods on the defined loci by applying empirical Bayesian methods. These will not change the locus definition, but will assign likelihoods that the identified loci represent a true methylation locus in each replicate group.

```
> hSL <- lociLikelihoods(hS, mD, cl = cl)
```

```
......
```

Visualising loci

By one of these methods, we finally acquire an annotated `methData` object, with the annotations describing the co-ordinates of each segment.

We can use this `methData` object, in combination with the `alignmentMeth` object, to plot the segmented genome.

```
> plotMeth(mD, hSL, chr = "Chr1", limits = c(1, 50000), cap = 10)
```

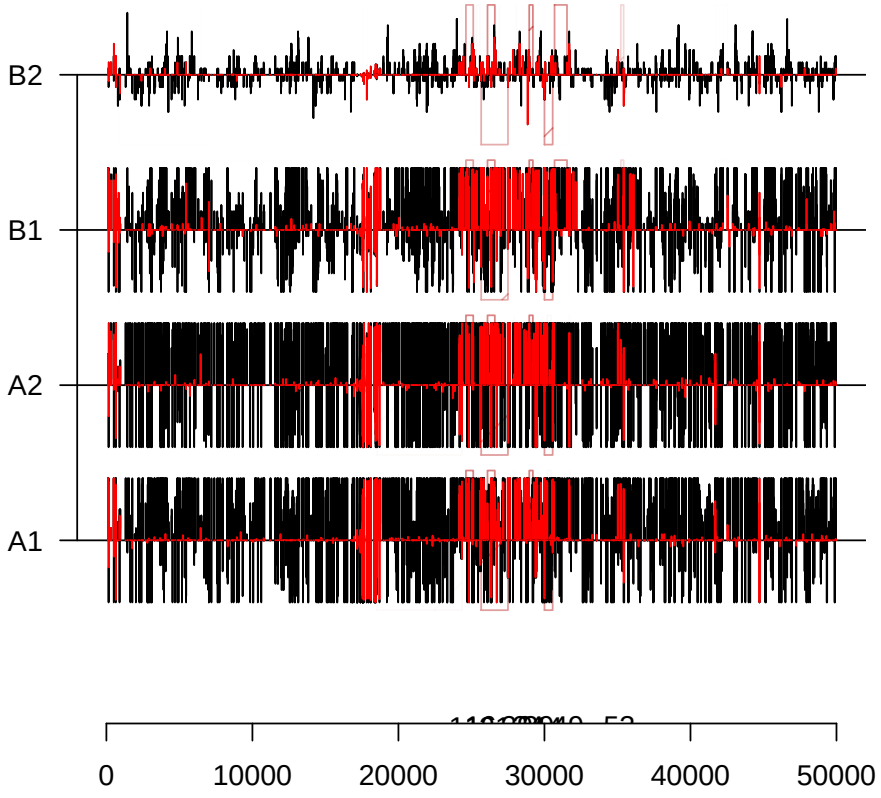

Figure 2: Methylation and identified loci on the first ten thousand bases of chromosome 1.

Differential Methylation analysis

We can also examine the `methData` object for differentially methylated regions using the beta-binomial methods [2] implemented in `baySeq`. We first define a group structure on the data.

```
> groups(hSL) <- list(NDE = c(1,1,1,1), DE = c("A", "A", "B", "B"))
```

The methObservables function pre-calculates a set of data to improve the speed of prior and posterior estimation (at some minor memory cost).

```
> hSL <- methObservables(hSL)
```

The density function used here is a composite of the beta-binomial and a binomial distribution that accounts for the reported non-conversion rates.

```
> densityFunction(hSL) <- bbNCDist
```

We can then determine a prior distribution on the parameters of the model for the data.

```
> hSL <- getPriors(hSL, cl = cl)
```

We can then find the posterior likelihoods of the models defined in the groups structure.

```
> hSL <- getLikelihoods(hSL, cl = cl)
```

.

We can then retrieve the data for the top differentially methylated regions.

```
> topCounts(hSL, "DE")
```

```
   seqnames   start     end width strand     A.1     A.2     B.1   B.2 Likelihood
1      Chr1  934849  935898  1050      +   0:314   0:410 121:324 17:51  0.9985317
2      Chr1 1359786 1360368   583      +   2:530   1:307  90:204  4:12  0.9982719
3      Chr1 1054379 1054858   480      +   49:62  82:124   3:163  0:27  0.9982339
4      Chr1  833024  834208  1185      +  45:171  55:197   0:185  0:23  0.9971859
5      Chr1 1914038 1914100    63      +   32:47   17:29    0:32   0:2  0.9969474
6      Chr1 1627523 1628436   914      +  70:312  66:308   0:358  0:45  0.9968858
7      Chr1 1253594 1256864  3271      + 418:797 370:812  10:680  3:95  0.9966642
8      Chr1 1498676 1499243   568      +  64:110   48:83    1:75   0:3  0.9966609
9      Chr1  777184  777313   130      +   21:25    8:13    0:28   0:4  0.9965997
10     Chr1 1346398 1346455    58      +  81:118   32:32   2:133   0:2  0.9963051
   ordering      FDR.DE     FWER.DE
1       B>A 0.001468295 0.001468295
2       B>A 0.001598219 0.003193902
3       A>B 0.001654184 0.004954374
4       A>B 0.001944161 0.007754524
5       A>B 0.002165842 0.010783418
6       A>B 0.002323908 0.013864072
7       A>B 0.002468464 0.017153624
8       A>B 0.002577290 0.020435424
9       A>B 0.002668737 0.023766251
10      A>B 0.002771351 0.027373309
```

Finally, to be a good citizen, we stop the cluster we started earlier:

```
> if(!is.null(cl))
+     stopCluster(cl)
```

Session Info

```
> sessionInfo()
```

```
R version 3.2.1 (2015-06-18)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.9.5 (Mavericks)

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
attached base packages:
[1] stats4    parallel  stats     graphics  grDevices utils     datasets  methods
[9] base

other attached packages:
 [1] segmentSeq_2.2.2        ShortRead_1.26.0        GenomicAlignments_1.4.1
 [4] Rsamtools_1.20.4        Biostrings_2.36.2       XVector_0.8.0
 [7] BiocParallel_1.2.19     baySeq_2.2.0            perm_1.0-0.0
[10] abind_1.4-3             GenomicRanges_1.20.5    GenomeInfoDb_1.4.1
[13] IRanges_2.2.5           S4Vectors_0.6.3         BiocGenerics_0.14.0

loaded via a namespace (and not attached):
 [1] zlibbioc_1.14.0      lattice_0.20-33      hwriter_1.3.2        tools_3.2.1
 [5] grid_3.2.1           Biobase_2.28.0       latticeExtra_0.6-26  lambda.r_1.1.7
 [9] futile.logger_1.4.1  RColorBrewer_1.1-2   futile.options_1.0.0 bitops_1.0-6
[13] BiocStyle_1.6.0
```

References

[1] Thomas J. Hardcastle and Krystyna A. Kelly and David C. Baulcombe. *Identifying small RNA loci from high-throughput sequencing data.* Bioinformatics (2012).

[2] Thomas J. Hardcastle and Krystyna A. Kelly. *Empirical Bayesian analysis of paired high-throughput sequencing data with a beta-binomial distribution.* BMC Bioinformatics (2013).