biomvRCNS: Copy Number study and Segmentation for multivariate biological data.

Yang Du*

April 18, 2013

Abstract

With high throughput experiments like tiling array and NGS, researchers are looking for continuous homogeneous segments or signal peaks, which would represent chromatin states, methylation ratio, transcripts or genome regions of deletion and amplification. While in a normal experimental set-up, these profiles would be generated for multiple samples or conditions with replicates. In the package *biomvRCNS*, a Hidden Semi Markov Model and one homogeneous segmentation model are implemented and tailored to handle multiple genomic profiles, with the aim of assisting in transcripts detection using high throughput technology like RNA-seq or tiling array, and copy number analysis using aCGH or targeted sequencing.

1 Introduction

To begin with *biomvRCNS*, load the package and read the manual page.

```
> library(biomvRCNS)
```

In the package, 3 main functions are provided for the batch processing of multiple chromosome regions across samples: `biomvRhsmm`, a hidden semi Markov model (HSMM); `biomvRseg`, a maximum likelihood based homogeneous segmentation model; and a third `biomvRmgmr`, custom batch function using max-gap-min-run algorithm. In the following sections we will illustrate their functionalities using example data. Currently the package does not deal with data correction, so input should be normalized by reference or paired sample and corrected for factor of interest before passing down.

2 Example of array CGH data set of Coriell cell lines

Extracted from packge *DNACopy*, the `coriell` data contains two aCGH studies (GM05296 and GM13330) of Corriel cell lines taken from Snijders et al. [2001]. In particular, with 2271 mapped features in total across 22 autosomes and chromosome X.

All three main functions accept common data matrix plus positional information as input or a *GRanges* object with data matrix stored in the meta columns. To get started, we first build a *GRanges* object from *data.frame*.

```
> data('coriell', package='biomvRCNS')
> head(coriell, n=3)
```

*email: duyang@fbn-dummerstorf.de

```
       Clone Chromosome Position Coriell.05296 Coriell.13330
1 GS1-232B23          1        1      0.000359      0.207470
2 RP11-82d16          1      469      0.008824      0.063076
3 RP11-62m23          1     2242     -0.000890      0.123881
```

```
> xgr<-GRanges(seqnames=paste('chr', coriell[,2], sep=''),
+         IRanges(start=coriell[,3], width=1, names=coriell[,1]))
> values(xgr)<-DataFrame(coriell[,4:5], row.names=NULL)
> xgr<-xgr[order(xgr)]
> head(xgr, n=3)
```

```
GRanges with 3 ranges and 2 metadata columns:
             seqnames       ranges strand | Coriell.05296
                <Rle>    <IRanges>  <Rle> |     <numeric>
  GS1-232B23     chr1 [   1,    1]      * |      0.000359
  RP11-82d16     chr1 [ 469,  469]      * |      0.008824
  RP11-62m23     chr1 [2242, 2242]      * |      -0.00089
             Coriell.13330
                 <numeric>
  GS1-232B23       0.20747
  RP11-82d16      0.063076
  RP11-62m23      0.123881
  ---
  seqlengths:
    chr1 chr10 chr11 chr12 chr13 ...  chr5  chr6  chr7  chr8  chr9
      NA    NA    NA    NA    NA ...    NA    NA    NA    NA    NA
```

Please be sure that the data is sorted with respect to their positions before feeding to the models.

2.1 Genomic segmentation with Hidden-semi Markov model

First we use the hidden-semi Markov model with the batch function `biomvRhsmm`, which will sequentially process each chromosome identified by the *seqnames* (using dummy name when no *GRanges* supplied in `x` or `xRange`), thus for non-continuous regions on the same chromosome user should give different *seqnames* to each part of the data. Within this package, there is one argument *grp*, for all main batch functions, which is used to assign data columns to groups according to the experimental design, say technical replicates or biological replicates. Sample columns within the same group could be treated simultaneously in the modelling process as well as iteratively. [1] In this example, the two profiles are considered independent and not similar, thus been given different values in the *grp* vector. Additionally there is a built-in automatic grouping method, given a valid clustering method *cluster.m* and *grp* set to `NULL`. By default, all data columns are assumed to be from the same group.

```
> rhsmm<-biomvRhsmm(x=xgr, maxbp=4E4, J=3, soj.type='gamma',
+         emis.type='norm', grp=c(1,2))
```

```
> show(rhsmm)
```

```
Object is of class: 'biomvRCNS'
List of parameters used in the model:
J, maxk, maxbp, maxgap, soj.type, emis.type, q.alpha, r.var,
```

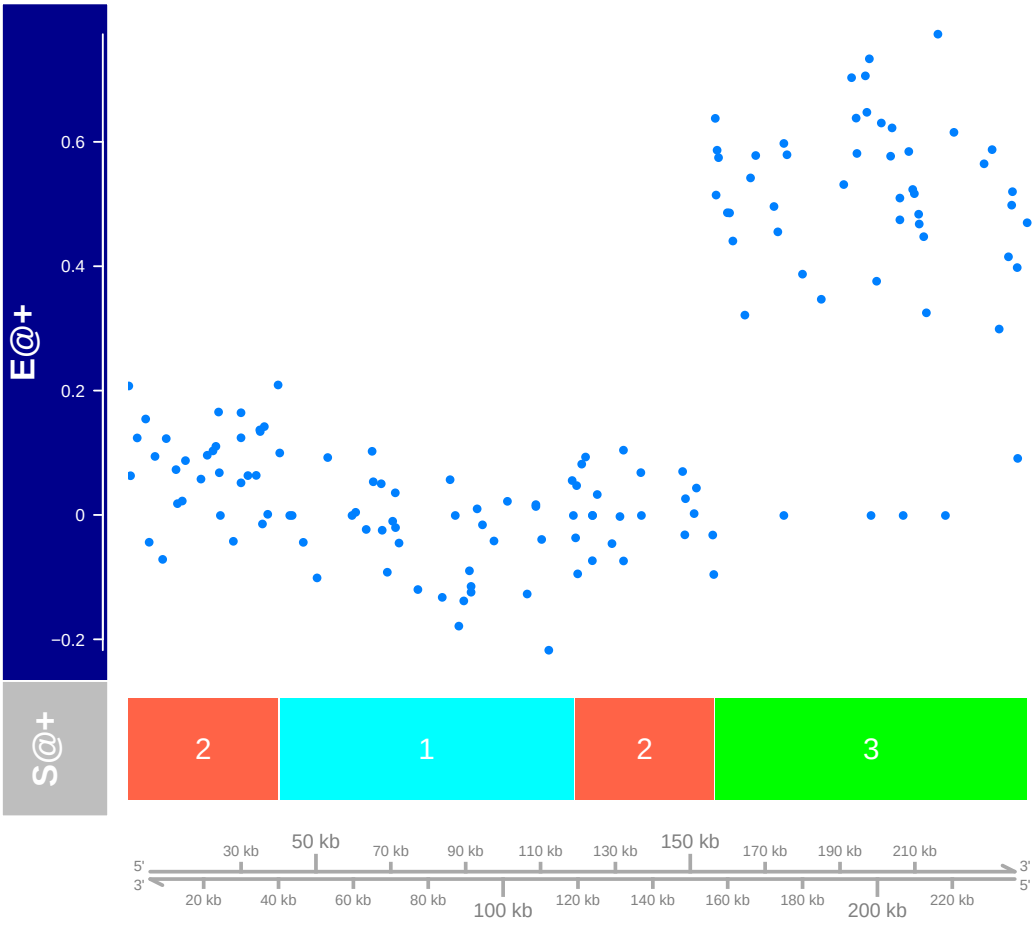
[1] Simultaneous treatment within group is currently available for *emis.type* equals 'mvnorm' or 'mvt' in `biomvRhsmm`, *poolGrp=TRUE* in `biomvRmgmr` and *twoStep=FALSE* in `biomvRseg`.

```
iterative, cMethod, maxit, tol, grp, cluster.m, avg.m, trim, na.rm

The segmented ranges:
GRanges with 197 ranges and 3 metadata columns:
        seqnames            ranges strand   |        SAMPLE
           <Rle>         <IRanges>  <Rle>   |   <character>
    [1]     chr1 [     1,  35001]      *   | Coriell.05296
    [2]     chr1 [ 40327,  85833]      *   | Coriell.05296
    [3]     chr1 [ 91001, 132148]      *   | Coriell.05296
    [4]     chr1 [147954, 193091]      *   | Coriell.05296
    [5]     chr1 [203907, 235001]      *   | Coriell.05296
    ...      ...               ...    ... ...           ...
  [193]     chr9  [33136,  75244]      *   | Coriell.13330
  [194]     chr9  [78040, 115001]      *   | Coriell.13330
  [195]     chr9  [ 8639,  14050]      *   | Coriell.13330
  [196]     chr9  [24326,  33001]      *   | Coriell.13330
  [197]     chr9  [75733,  77808]      *   | Coriell.13330
             STATE        AVG
       <character>  <numeric>
    [1]           2  0.0045915
    [2]           2  0.0129660
    [3]           2  0.0066565
    [4]           2 -0.0056100
    [5]           2 -0.0028350
    ...         ...        ...
  [193]           2   -0.02824
  [194]           2   -0.01081
  [195]           1   -0.19048
  [196]           1   -0.16646
  [197]           3    0.09102
  ---
  seqlengths:
    chr1 chr10 chr11 chr12 chr13 ...  chr5  chr6  chr7  chr8  chr9
      NA    NA    NA    NA    NA ...    NA    NA    NA    NA    NA
```

In the above run, we limit the model complexity by setting the *maxbp* to $4E4$, which will restrict the maximum sojourn time to *maxbp*. J is the number of states in the HSMM model, this argument can be given explicitly or estimated from prior information provided in *xAnno*. Argument *soj.type* defines the type of sojourn distribution; with Gamma distributed sojourn, the neighbouring position will tend to have the same state, and transit to other states if far apart. In this way the sojourn distribution fully incorporate the positional information into the probabilistic framework. Argument *emis.type* controls the distribution of emission probability, in this case the log2 ratio of aCGH data is considered to follow Normal distribution. The function will then call C codes and estimate the most likely state sequence, with either *cMethod='F-B'* or *cMethod='Viterbi'*. The *F-B* method uses a forward-backward algorithm described in Guédon [2003], which gives a smooth state sequences, whereas the Viterbi algorithm with *cMethod='Viterbi'* will use the states profile estimated by the forward-backward algorithm and rebuild the most likely state sequence. The function returns an object of class *biomvRCNS*, in which the `res` slot is a *GRanges* object contain the summary of each estimated segments. There are three meta columns: column `SAMPLE` gives the column name of which sample this segment belongs to; column `STATE`, the estimated state for each segments, the lower state number represents state with lower mean value, thus in this example, a state of 1 could represent region of deletion and 3 for region of duplication, whereas state 2 could be considered copy neutral; column `AVG`, gives the segment average value.

A `plot` method has been implemented for *biomvRCNS* object using package *Gviz*, by default the `plot` method tries to output graphics to multiple EPS/PDF files for each chromosome region and sample. Here we set *tofile=FALSE* to output graphics to the current device, and only show resulting graphics for chromosome 1 from sample Coriell.13330.



2.2 Using other methods provided in the package

In this section, we use the other two batch functions to process the `coriell` data. First we use `biomvRseg`, in which a similar segmentation method like in the package *tillingArray* [Huber et al., 2006] is implemented and extended to handle Poison and Negative binomial distributed data. The function shares several argument with `biomvRhsmm`, like *maxbp* and *grp*. The *maxseg* gives the maximum number of segment per chromosome region, while the optimal number of segment per chromosome region is determined internally by assessing the likelihood with optional penalty terms, by default *penalty='BIC'* is used. Another option is to use modified Bayes information criterion *penalty='mBIC'* [Zhang and Siegmund, 2007], as in the CBS algorithm used in *DNAcopy*. The function proceed in the following manner: assuming within each group sample columns exhibit similar patterns, and thus be processed simultaneously in the first step. By maximizing the likelihood the optimal number of segments is selected for each group. And in a second step if *twoStep=TRUE* or merging is necessary, the candidate segments produced in the first step are merged with respect to each sample, thus forcing sample columns in the same group to have a more unified segmentation result yet keeping it possible to have sample specific pattern.

```
> rseg<-biomvRseg(x=xgr, maxbp=4E4, maxseg=10, family='norm', grp=c(1,2))
```

```
> head(rseg@res)
```

```
GRanges with 6 ranges and 3 metadata columns:
      seqnames           ranges strand |        SAMPLE        AVG
         <Rle>        <IRanges>  <Rle> |   <character>  <numeric>
  [1]     chr1 [     1,  35001]      * | Coriell.05296  0.0045915
  [2]     chr1 [ 35106,  36207]      * | Coriell.05296  0.1335200
  [3]     chr1 [ 37117,  85833]      * | Coriell.05296  0.0129660
  [4]     chr1 [ 87215,  89493]      * | Coriell.05296 -0.1010400
  [5]     chr1 [ 91001, 132148]      * | Coriell.05296  0.0066565
  [6]     chr1 [132171, 136942]      * | Coriell.05296  0.1421530
            STATE
      <character>
  [1]         LOW
  [2]        HIGH
  [3]        HIGH
  [4]         LOW
  [5]         LOW
  [6]        HIGH
  ---
  seqlengths:
    chr1 chr10 chr11 chr12 chr13 ...  chr5  chr6  chr7  chr8  chr9
      NA    NA    NA    NA    NA ...    NA    NA    NA    NA    NA
```

After the example run, the function returns a *biomvRCNS* object, containing similar information as the previous `biomvRhsmm` run, except that the *STATE* column now only have a binary state value of either "HIGH" or "LOW", which is simply graded as 'HIGH' if the segment mean is higher than the grand mean of the whole region, and 'LOW' otherwise.

It is also possible to use the simple max-gap-min-run algorithm to segment aCGH profiles, by calling `biomvRmgmr`. But due to the binary nature of the algorithm, one have to run twice in order to get both extremely high and low segments, then combine the resulting *GRanges* manually.

```
> rmgmrh<-biomvRmgmr(xgr, q=0.9, high=T, maxgap=1000, minrun=2500, grp=c(1,2))
> rmgmrl<-biomvRmgmr(xgr, q=0.1, high=F, maxgap=1000, minrun=2500, grp=c(1,2))
> res<-c(rmgmrh@res, rmgmrl@res)
```

3 Example of RNA-seq data from ENCODE

The data contains gene expressions and transcript annotations in the region of the human TP53 gene (chr17:7,560,001-7,610,000 from the Human February 2009 (GRCh37/hg19) genome assembly), which is part of the long RNA-seq data generated by ENCODE [Consortium, 2004] /Cold Spring Harbor Lab, containing 2 cell types (GM12878 and K562) with 2 replicates each. The libraries were sequenced on the Illumina GAIIx platform as paired-ends for 76 or 101 cycles for each read. The average depth of sequencing was 200 million reads (100 million paired-ends). The data were mapped against hg19 using Spliced Transcript Alignment and Reconstruction (STAR).

To generate local read counts, alignment files were pulled from UCSC (`http://hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeCshlLongRnaSeq/`) using package *Rsamtools*. And subsequently reads were counted in each non-overlapping unit sized window for the region (chr17:7,560,001-7,610,000). In the pre-compiled data `encodeTP53` , a window size of 25bp was used with the chunk of code below.

```
> winsize<-25
> cgr<-GRanges("chr17", strand='-',
+         IRanges(start=seq(7560001, 7610000, winsize), width =winsize))
> bf<-system.file("extdata", "encodeFiles.txt", package = "biomvRCNS")
```

```
> bamfiles<-read.table(bf, header=T, stringsAsFactors=F)
> library(Rsamtools)
> which<-GRanges("chr17", IRanges(7560001, 7610000))
> param<-ScanBamParam(which=which, what=scanBamWhat())
> for(i in seq_len(nrow(bamfiles))){
+         frd<-scanBam(bamfiles[i,1], param=param)
+         frdgr<-GRanges("chr17", strand=frd[[1]]$strand,
+                 IRanges(start=frd[[1]]$pos , end = frd[[1]]$pos+frd[[1]]$qwidth-1))
+         mcols(cgr)<-DataFrame(mcols(cgr), DOC=countOverlaps(cgr, frdgr))
+ }
```

The pre-compiled data `encodeTP53` also includes the regional annotation of TP53 RNAs isoforms, `gmgr`, which were derived from the ENCODE Gene Annotations (GENCODE), `http://hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeGencodeV4/wgEncodeGencodeManualV4.gtf.gz)` , and subset to only isoforms of TP53 gene. The annotation object `gmgr` could be rebuilt with the following lines using the included file under `extdata`.

```
> af<-system.file("extdata", "gmodTP53.csv", package = "biomvRCNS")
> gtfsub<-read.table(af, fill=T, stringsAsFactors=F)
> idx<-gtfsub[,3]=='CDS' | gtfsub[,3]=='UTR'
> gmgr<-GRanges("chr17", IRanges(start=gtfsub[idx, 4], end=gtfsub[idx, 5],
+         names=gtfsub[idx, 13]), strand='-', TYPE=gtfsub[idx, 3])
```

3.1 Transcript detection with Hidden-semi Markov model

We first load the `encodeTP53` data, poll the read counts for each cell type and add 1 to the base count to increase stability.

```
> data(encodeTP53)
> cgr<-encodeTP53$cgr
> gmgr<-encodeTP53$gmgr
> mcols(cgr)<-DataFrame(
+         Gm12878=1+rowSums(as.matrix(mcols(cgr)[,1:2])),
+         K562=1+rowSums(as.matrix(mcols(cgr)[,3:4])) )
```

For count data from sequencing, the *emis.type* could be set to either `'pois'` or `'nbinom'`, though `'pois'` is preferred for sharp boundary detection. For the sojourn settings, instead of using the uninformative flat prior, we here use estimates from other data source as a prior. We load the *TxDb.Hsapiens.UCSC.hg19.knownGene* known gene database, and pass the *TranscriptDb* object to *xAnno*. Then internally sojourn parameters and state number J will be estimated from *xAnno* by calling function `sojournAnno`. When given a *TranscriptDb* object to *xAnno*, state number would be set to 3 and each represents 'intergenic', 'intron', 'exon'. One can also supply a named *list* object with initial values for paramters of distribution specified by *soj.type*.

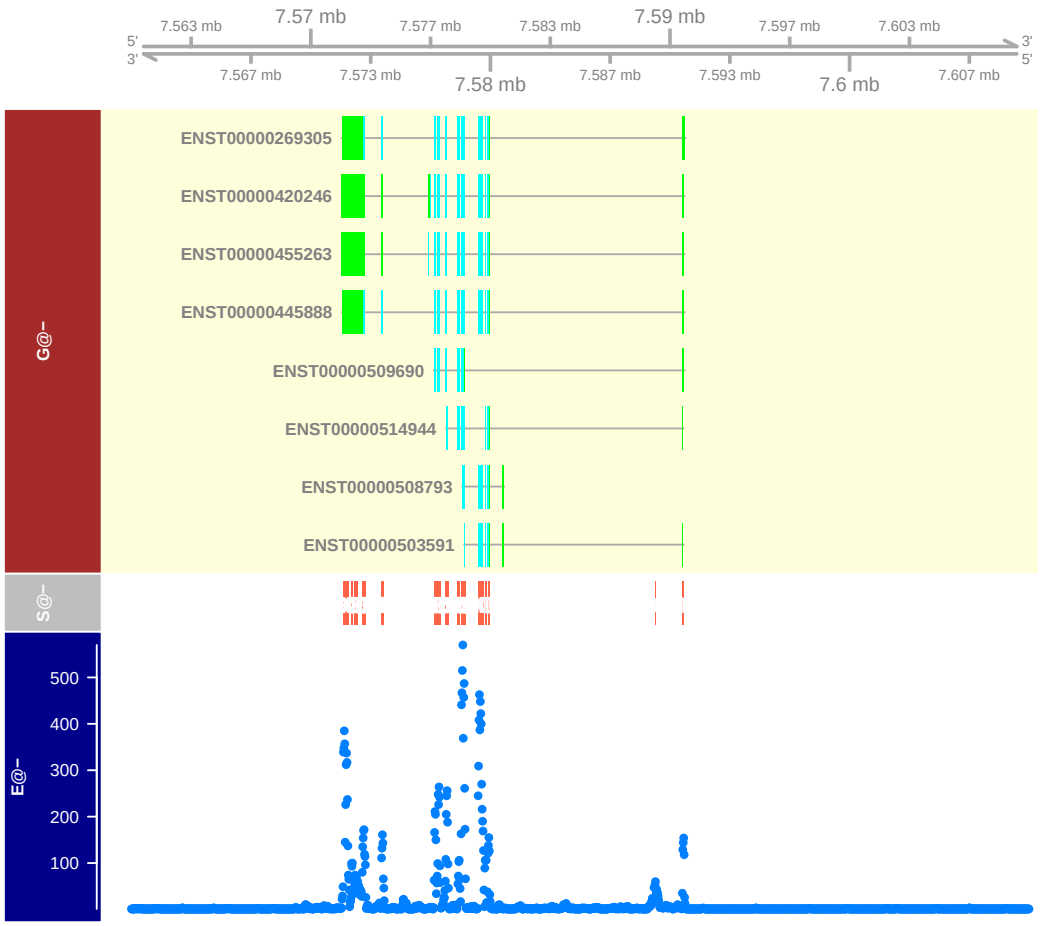
```
> library(TxDb.Hsapiens.UCSC.hg19.knownGene)
> txdb<-TxDb.Hsapiens.UCSC.hg19.knownGene
> rhsmm<-biomvRhsmm(x=cgr, xAnno=txdb, maxbp=1E3, soj.type='gamma',
+         emis.type='pois', cMethod='F-B', q.alpha=0.01)
```

As in the ENCODE guide [Consortium, 2011], the study identified the p53 isoform observed in K562 cells has a longer 3'UTR than the isoform seen in the GM12878 cell line. So here we plot our model estimates and consider the third state, namely 'exon', to represent detected transcripts. And the HSMM model clearly picked up the extra transcripts of the K562 cell line at the 3'UTR.

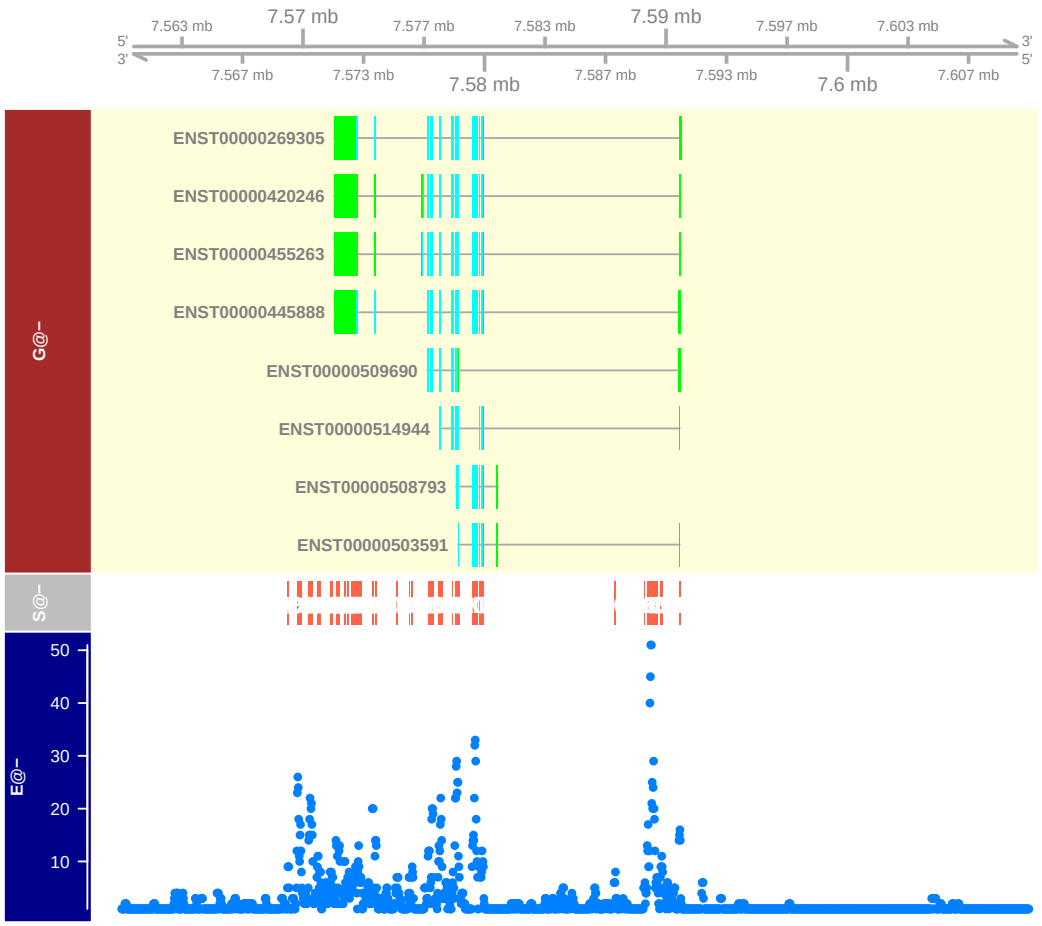
```
> rhsmm@res[mcols(rhsmm@res)[,'STATE']=='exon']


GRanges with 51 ranges and 3 metadata columns:
       seqnames             ranges strand   |      SAMPLE       STATE
          <Rle>          <IRanges>  <Rle>   | <character> <character>
   [1]    chr17 [7571801, 7572125]      -   |     Gm12878        exon
   [2]    chr17 [7572251, 7572350]      -   |     Gm12878        exon
   [3]    chr17 [7572426, 7572550]      -   |     Gm12878        exon
   [4]    chr17 [7572601, 7572625]      -   |     Gm12878        exon
   [5]    chr17 [7572851, 7573050]      -   |     Gm12878        exon
   ...      ...                ...    ... ...         ...         ...
  [47]    chr17 [7588826, 7588850]      -   |        K562        exon
  [48]    chr17 [7588951, 7589400]      -   |        K562        exon
  [49]    chr17 [7589426, 7589525]      -   |        K562        exon
  [50]    chr17 [7589676, 7589825]      -   |        K562        exon
  [51]    chr17 [7590701, 7590800]      -   |        K562        exon
             AVG
       <numeric>
   [1]       312
   [2]        96
   [3]        61
   [4]        60
   [5]       127
   ...       ...
  [47]       6.0
  [48]      20.0
  [49]       6.0
  [50]       9.0
  [51]      14.5
  ---
  seqlengths:
   chr17
      NA
```

›53@chr17.7560000−7610000@Gm128



TP53@chr17.7560000−7610000@K562



Now we can locate those novel detected fragments in K562 cell line comparing to the annotation and those detected in Gm12878 cell line. One can then follow up those findings either by gene structure prediction using local nucleotides composition or by experimental validation.

```
> nK2gm<-findOverlaps(rhsmm@res[k], gmgr)@queryHits
> nK2G<-findOverlaps(rhsmm@res[k], rhsmm@res[g])@queryHits
> rhsmm@res[k][setdiff(seq_len(sum(k)), unique(c(nK2G, nK2gm)))]
```

```
GRanges with 18 ranges and 3 metadata columns:
       seqnames            ranges strand   |      SAMPLE       STATE
          <Rle>         <IRanges>  <Rle>   | <character> <character>
   [1]    chr17 [7569151, 7569225]      -   |        K562        exon
   [2]    chr17 [7569651, 7569925]      -   |        K562        exon
   [3]    chr17 [7570301, 7570550]      -   |        K562        exon
   [4]    chr17 [7570751, 7570850]      -   |        K562        exon
   [5]    chr17 [7570901, 7571000]      -   |        K562        exon
   ...      ...                ...    ... ...         ...         ...
  [14]    chr17 [7587126, 7587175]      -   |        K562        exon
  [15]    chr17 [7587201, 7587225]      -   |        K562        exon
  [16]    chr17 [7588826, 7588850]      -   |        K562        exon
  [17]    chr17 [7589426, 7589525]      -   |        K562        exon
```

```
  [18]    chr17 [7589676, 7589825]      -   |        K562        exon
            AVG
       <numeric>
   [1]         9
   [2]        15
   [3]        16
   [4]        10
   [5]         8
   ...       ...
  [14]         6
  [15]         8
  [16]         6
  [17]         6
  [18]         9
  ---
  seqlengths:
   chr17
      NA
```

The other 2 batch functions could also be similarly applied here.

```
> rseg<-biomvRseg(x=cgr, maxbp=1E3, maxseg=20, family='pois')
> rmgmr<-biomvRmgmr(x=cgr, q=0.99, maxgap=50, minrun=100)
```

4 More

To be continued ...

5 Session information

```
> sessionInfo()
```

```
R Under development (unstable) (2013-04-12 r62558)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=C                 LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] grid      parallel  stats     graphics  grDevices utils
[7] datasets  methods   base

other attached packages:
[1] TxDb.Hsapiens.UCSC.hg19.knownGene_2.9.0
[2] GenomicFeatures_1.13.0
```

```
[3] AnnotationDbi_1.23.1
[4] Biobase_2.21.0
[5] biomvRCNS_1.1.1
[6] Gviz_1.5.0
[7] GenomicRanges_1.13.5
[8] IRanges_1.19.1
[9] BiocGenerics_0.7.0

loaded via a namespace (and not attached):
 [1] biomaRt_2.17.0     Biostrings_2.29.0  biovizBase_1.9.1
 [4] bitops_1.0-5       BSgenome_1.29.0    cluster_1.14.4
 [7] colorspace_1.2-2   DBI_0.2-5          dichromat_2.0-0
[10] Hmisc_3.10-1       labeling_0.1       lattice_0.20-15
[13] munsell_0.4        mvtnorm_0.9-9994   plyr_1.8
[16] RColorBrewer_1.0-5 RCurl_1.95-4.1     Rsamtools_1.13.3
[19] RSQLite_0.11.3     rtracklayer_1.21.1 scales_0.2.3
[22] stats4_3.1.0       stringr_0.6.2      tools_3.1.0
[25] XML_3.96-1.1       zlibbioc_1.7.0
```

References

The ENCODE Project Consortium. The encode (encyclopedia of dna elements) project. *Science*, 306(5696):636–640, 2004.

The ENCODE Project Consortium. A user's guide to the encyclopedia of dna elements (encode). *PLoS Biol*, 9(4): e1001046, 04 2011. doi: 10.1371/journal.pbio.1001046. URL http://dx.doi.org/10.1371%2Fjournal.pbio.1001046.

Yann Guédon. Estimating Hidden Semi-Markov Chains from Discrete Sequences. *Journal of Computational and Graphical Statistics*, 12(3):604–639, 2003. ISSN 10618600. doi: 10.2307/1391041. URL http://dx.doi.org/10.2307/1391041.

Wolfgang Huber, Joern Toedling, and Lars M. Steinmetz. Transcript mapping with high-density oligonucleotide tiling arrays. *Bioinformatics*, 22:1963–1970, 2006.

Antoine M Snijders, Norma Nowak, Richard Segraves, Stephanie Blackwood, Nils Brown, Jeffrey Conroy, Greg Hamilton, Anna Katherine Hindle, Bing Huey, Karen Kimura, et al. Assembly of microarrays for genome-wide measurement of dna copy number. *Nature genetics*, 29:263–264, 2001.

Nancy R. Zhang and David O. Siegmund. A modified bayes information criterion with applications to the analysis of comparative genomic hybridization data. *Biometrics*, 63(1):22–32, 2007. ISSN 1541-0420. doi: 10.1111/j.1541-0420.2006.00662.x. URL http://dx.doi.org/10.1111/j.1541-0420.2006.00662.x.