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

Yang Du*

3 February 2012

**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 for the main function.

```
> library(biomvRCNS)
> ? 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.

## 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 RclassGRanges 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 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*, thus for non-continuous regions on the same chromosome user should give different *seqnames* to those data. Within this package, there is one argument *grp*, for the main batch function, 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 modeling 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 *clusterm* and *grp* set to `NULL`.

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

```
> show(reshsmm)
```

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

GRanges with 394 ranges and 3 metadata columns:
        seqnames           ranges strand   |        SAMPLE
```

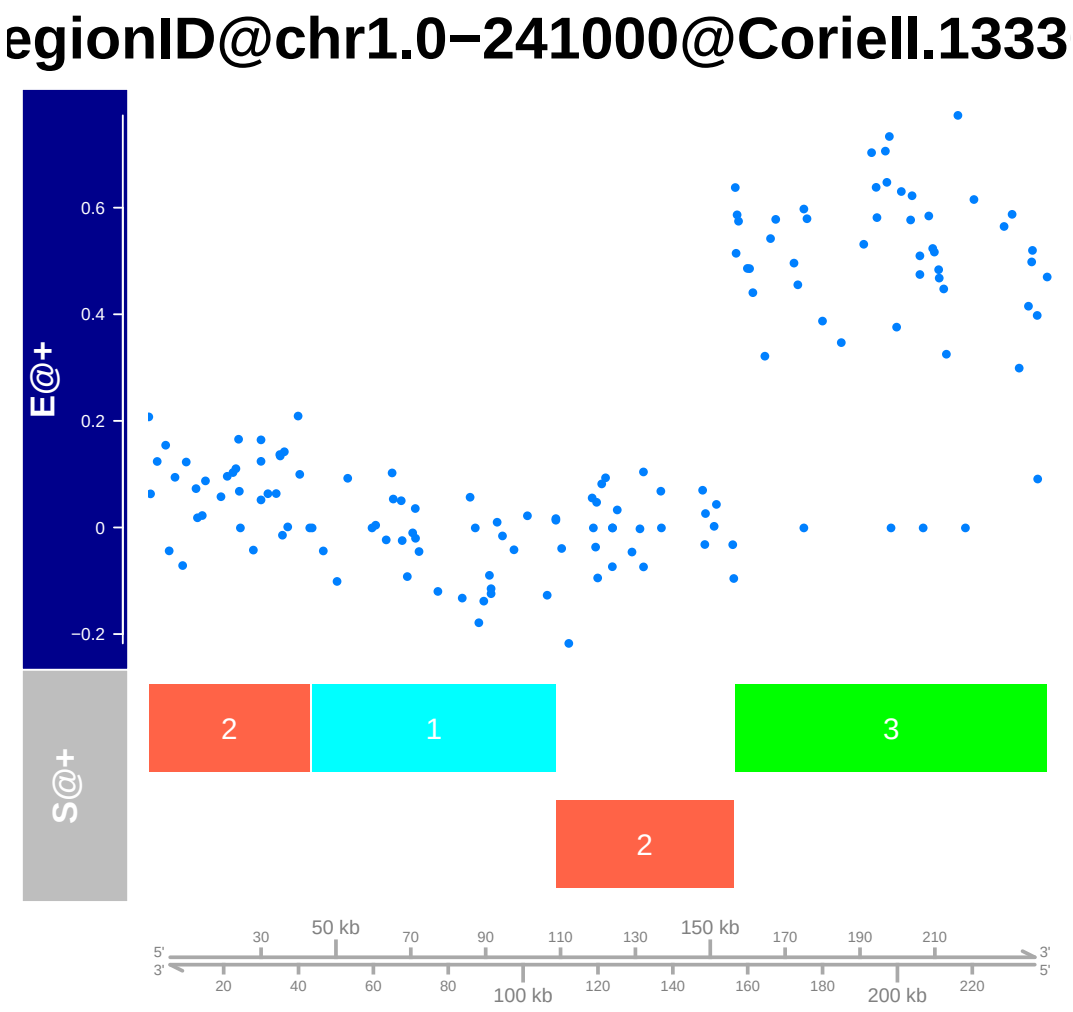
[1] Simultaneous treatment within group is available for *emis.type= 'mvnorm'* in `biomvRhsmm`, *poolGrp=TRUE* in `biomvRmgmr` and *twoStep=FALSE* in `biomvRseg`.

```
        <Rle>        <IRanges>  <Rle>   |   <character>
  [1]     chr1 [     1,  14283]      *   | Coriell.05296
  [2]     chr1 [ 22485,  35001]      *   | Coriell.05296
  [3]     chr1 [ 43634,  69062]      *   | Coriell.05296
  [4]     chr1 [ 91001, 108746]      *   | Coriell.05296
  [5]     chr1 [118443, 131214]      *   | Coriell.05296
  [6]     chr1 [147954, 159916]      *   | Coriell.05296
  [7]     chr1 [166103, 193091]      *   | Coriell.05296
  [8]     chr1 [203907, 216154]      *   | Coriell.05296
  [9]     chr1 [ 15138,  20961]      *   | Coriell.05296
  ...      ...               ...    ... ...           ...
[386]     chr9  [45534,  59802]      *   | Coriell.13330
[387]     chr9  [63082,  74001]      *   | Coriell.13330
[388]     chr9  [78040,  93001]      *   | Coriell.13330
[389]     chr9  [94359, 115001]      *   | Coriell.13330
[390]     chr9  [ 8639,  14050]      *   | Coriell.13330
[391]     chr9  [60056,  62152]      *   | Coriell.13330
[392]     chr9  [34523,  40369]      *   | Coriell.13330
[393]     chr9  [75244,  77808]      *   | Coriell.13330
[394]     chr9  [93749,  94182]      *   | Coriell.13330
           STATE         MEAN
      <character>    <numeric>
  [1]           2 -0.005337727
  [2]           2  0.008203417
  [3]           2  0.021995500
  [4]           2  0.004843500
  [5]           2  0.020345000
  [6]           2  0.034815833
  [7]           2  0.001478909
  [8]           2 -0.002598917
  [9]           3  0.095219333
  ...         ...          ...
[386]           2  -0.03895386
[387]           2  -0.02281038
[388]           2  -0.04401375
[389]           2   0.02875433
[390]           1  -0.18343125
[391]           1  -0.11393500
[392]           3   0.04516110
[393]           3   0.09125357
[394]           3   0.04649300
---
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 $1E4$, 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. Argument *emis.type* control the distribution of emission probability, in this case the log2 ratio of aCGH data is considered to follow a Normal distribution. The function will then call C codes to estimate the most likely state sequence, with either *cMethod='BandF'* or *cMethod='Viterbi'*. The function returns a 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: columan `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 `MEAN`, gives the segment mean.

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 per sample. Here we set *tofile=FALSE* to output graphics to the current device.




## 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* 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'* 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* 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.

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

```
> head(resseg@res)
```

```
GRanges with 6 ranges and 3 metadata columns:
      seqnames          ranges strand |        SAMPLE         MEAN
         <Rle>       <IRanges>  <Rle> |   <character>    <numeric>
  [1]     chr1 [    1, 240001]      * | Coriell.05296  0.019731190
  [2]     chr1 [    1, 240001]      * | Coriell.13330  0.181743289
  [3]    chr10 [    1,  14351]      * | Coriell.05296 -0.005220053
  [4]    chr10 [14545,  28065]      * | Coriell.05296 -0.022005000
  [5]    chr10 [29926,  64188]      * | Coriell.05296 -0.018713632
  [6]    chr10 [65001,  69550]      * | Coriell.05296  0.280758000
           STATE
      <character>
  [1]         LOW
  [2]         LOW
  [3]         LOW
  [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 regions, then combine the resutling *GRanges* manually.

```
> resmgmrh<-biomvRmgmr(x=xgr, q=0.9, high=TRUE, maxgap=1000, minrun=2500, grp=c(1,2))
> resmgmrl<-biomvRmgmr(x=xgr, q=0.1, high=FALSE, maxgap=1000, minrun=2500, grp=c(1,2))
```

```
> res<-c(resmgmrh@res, resmgmrl@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/Cold Spring Harbor Lab, containing 2 cell types (GM12878 and K562) with 2 replicates each.

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", IRanges(start=seq(7560001, 7610000, winsize), width =winsize), strand='-')
> 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", IRanges(start=frd[[1]]$pos , end = frd[[1]]$pos+frd[[1]]$qwidth-1),
+                                 strand=frd[[1]]$strand)
+         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), subset to only isoforms of TP53 gene. `http://hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeGencodeV4/wgEncodeGencodeManualV4.gtf.gz`).

```
> 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)
> mcols(encodeTP53$cgr)<-DataFrame(Gm12878=1+rowSums(as.matrix(mcols(encodeTP53$cgr)[,1:2])),
+         K562=1+rowSums(as.matrix(mcols(encodeTP53$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`. The three states estimated would each represents 'intergenic', 'intron', 'exon'.

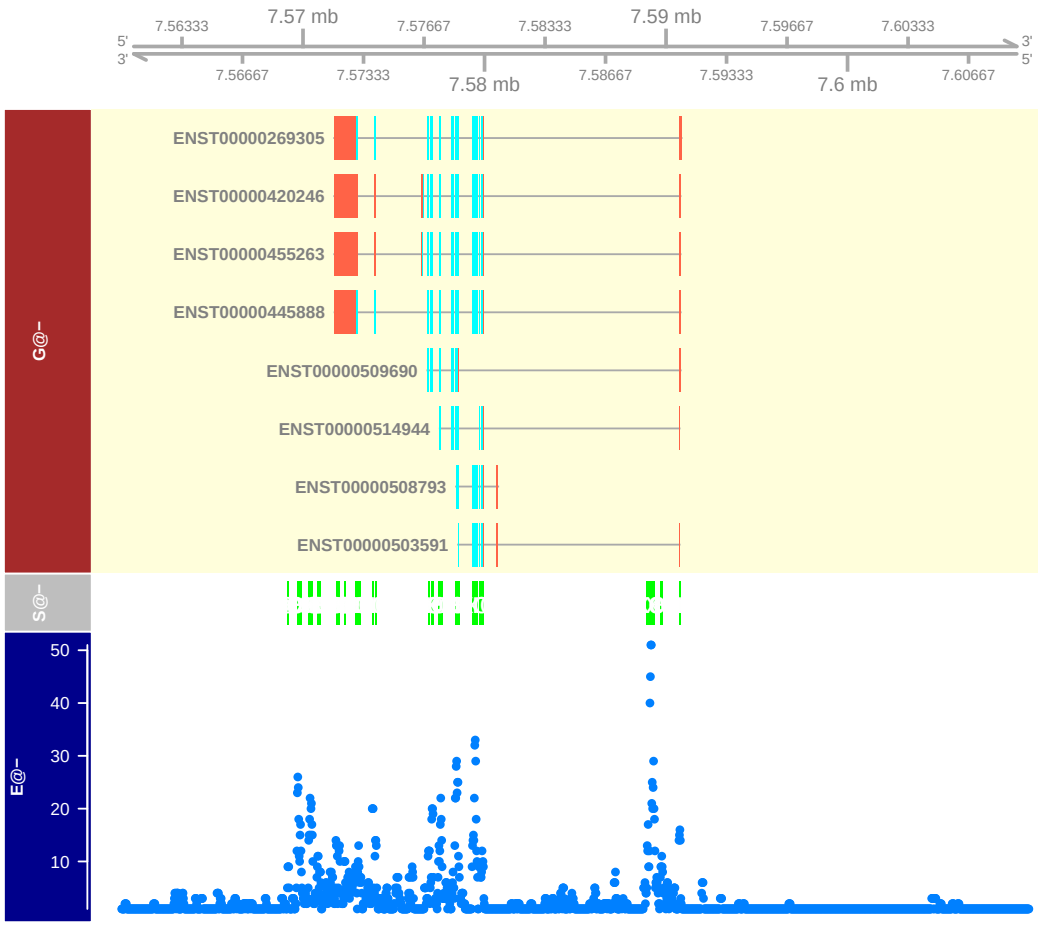
```
> library(TxDb.Hsapiens.UCSC.hg19.knownGene)
> txdb<-TxDb.Hsapiens.UCSC.hg19.knownGene
> reshsmm<-biomvRhsmm(x=encodeTP53$cgr, xAnno=txdb, maxbp=5E3, soj.type='gamma',
+         emis.type='pois', q.alpha=0.01, r.var=1)
```

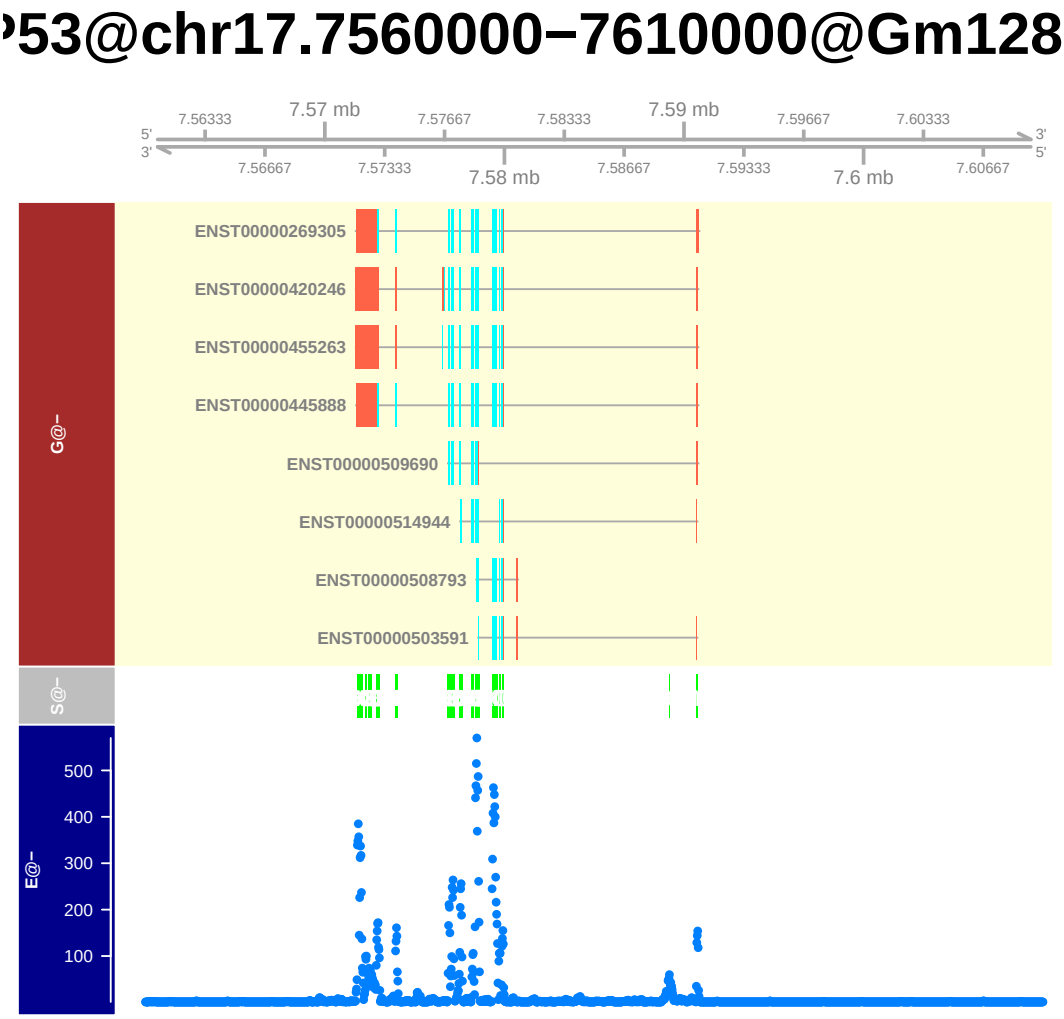
As in the ENCODE guide [Consortium, 2011], the study identified the p53 isoform observed in K562 cells has a longer 3'UTR region than the isoform seen in the GM12878 cell line. So here we plot our model estimates and consider the third state, namely 'exon', to represent detected transcripts. And the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

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

```
GRanges with 37 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
   [6]    chr17 [7573926, 7574050]      -   |     Gm12878        exon
   [7]    chr17 [7576851, 7576975]      -   |     Gm12878        exon
   [8]    chr17 [7577001, 7577225]      -   |     Gm12878        exon
   [9]    chr17 [7577476, 7577650]      -   |     Gm12878        exon
   ...      ...                ...    ... ...         ...         ...
  [29]    chr17 [7576876, 7576975]      -   |        K562        exon
  [30]    chr17 [7577051, 7577200]      -   |        K562        exon
  [31]    chr17 [7577426, 7577675]      -   |        K562        exon
  [32]    chr17 [7578351, 7578625]      -   |        K562        exon
  [33]    chr17 [7579301, 7579625]      -   |        K562        exon
  [34]    chr17 [7579676, 7579950]      -   |        K562        exon
  [35]    chr17 [7588926, 7589400]      -   |        K562        exon
  [36]    chr17 [7589676, 7589825]      -   |        K562        exon
  [37]    chr17 [7590701, 7590800]      -   |        K562        exon
           MEAN
      <numeric>
   [1]  252.2308
   [2]   89.5000
   [3]   63.8000
   [4]   60.0000
   [5]  130.0000
   [6]  122.6000
   [7]  159.0000
   [8]  151.1111
   [9]  165.8571
   ...       ...
  [29] 11.750000
  [30] 15.166667
  [31] 13.200000
  [32] 19.454545
  [33] 17.538462
  [34]  8.909091
  [35] 22.789474
  [36]  8.833333
  [37] 14.750000
  ---
  seqlengths:
   chr17
      NA
```

# TP53@chr17.7560000−7610000@K562






The other 2 functions could also be similarly applied here.

```
> resseg<-biomvRseg(x=encodeTP53$cgr, maxbp=5E3, maxseg=20, family='pois')
> resmgmr<-biomvRmgmr(x=encodeTP53$cgr, q=0.99, maxgap=50, minrun=100)
```

# 4 Session information

```
> sessionInfo()
```

```
R Under development (unstable) (2013-02-28 r62089)
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=C
 [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_2.8.0
 [2] GenomicFeatures_1.11.13
 [3] AnnotationDbi_1.21.11
 [4] Biobase_2.19.2
 [5] biomvRCNS_0.99.0
 [6] Gviz_1.3.13
 [7] GenomicRanges_1.11.33
 [8] IRanges_1.17.35
 [9] BiocGenerics_0.5.6
[10] mvtnorm_0.9-9994

loaded via a namespace (and not attached):
 [1] BSgenome_1.27.1    Biostrings_2.27.11 DBI_0.2-5
 [4] Hmisc_3.10-1       RColorBrewer_1.0-5 RCurl_1.95-3
 [7] RSQLite_0.11.2     Rsamtools_1.11.19  XML_3.95-0.1
[10] biomaRt_2.15.0     biovizBase_1.7.6   bitops_1.0-5
[13] cluster_1.14.3     colorspace_1.2-1   dichromat_2.0-0
[16] labeling_0.1       lattice_0.20-13    munsell_0.4
[19] plyr_1.8           rtracklayer_1.19.9 scales_0.2.3
[22] stats4_3.0.0       stringr_0.6.2      tools_3.0.0
[25] zlibbioc_1.5.0
```

# References


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`.

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 by cgh. *Nature genetics*, 29:263–264, 2001.