VanillaICE: Hidden Markov Models for the Assessment of Chromosomal Alterations using High-throughput SNP Arrays

Robert Scharpf

July 12, 2012

Abstract

This package provides an implementation of a hidden Markov Model for high throughput SNP arrays. Users of this package should already have available locus-level estimates of copy number and B allele frequencies or genotype calls. Copy number estimates can be relative or absolute.

1 Overview

This vignette requires that you have

- an absolute estimate of the *total* copy number organized such that rows correspond to loci and columns correspond to samples

  and / or

- a matrix of B allele frequencies or genotype calls (1=AA, 2 = AB, 3= BB): rows correspond to loci and columns correspond to samples

If Affymetrix CEL files or Illumina Idat files were processed with the R package *crlmm*, see the vignette `crlmmDownstream` included with this package.
Other HMM implementations for the joint analysis of copy number and genotype include QuantiSNP [1] and PennCNV [3].

Data considerations. The HMM implemented in this package is most relevant for heritable diseases for which integer copy numbers are expected. For somatic cell diseases such as cancer, we suggest circular binary segmentation, as implemented in the R package DNAcopy [2].

Citing this software. Robert B Scharpf, Giovanni Parmigiani, Jonathan Pevsner, and Ingo Ruczinski. Hidden Markov models for the assessment of chromosomal alterations using high-throughput SNP arrays. *Annals of Applied Statistics*, 2(2):687–713, 2008.

2 Organizing the locus-level data

This package includes simulated genotype and copy number data for 9165 SNPs.

```
> library(oligoClasses)
> library(VanillaICE)
> library2(SNPchip)
> library2(Biobase)
> library2(IRanges)
> data(locusLevelData)
```

The copy number estimates in the `locusLevelData` object were multiplied by 100 and saved as an integer. An object of class `oligoSnpSet` is instantiated from the simulated data in the next code chunk. Note that the assay data elements for the `oligoSnpSet` object should be integers. The `integerMatrix` scales the copy number matrix by a factor of 100 (by default) and returns a matrix of integers. When available, B allele frequencies should be scaled by a factor of 1000. For example,
integerMatrix(bAlleleFreq, scale=1000) \verb

```
> cn <- log2(locusLevelData[["copynumber"]]/100)
> oligoSet <- new("oligoSnpSet",
+                 copyNumber=integerMatrix(cn),
+                 call=locusLevelData[["genotypes"]],
+                 callProbability=locusLevelData[["crlmmConfidence"]],
+                 annotation=locusLevelData[["platform"]],
+                 genome="hg19")
> oligoSet <- oligoSet[!is.na(chromosome(oligoSet)), ]
> oligoSet <- oligoSet[chromosome(oligoSet) < 3, ]
```

3 Fitting the HMM

When jointly modeling the copy number and genotypes/allele frequencies, we assume that the genotype and copy number estimates are independent conditional on the underlying hidden state. The method `hmm` is defined for several classes, including objects of class `oligoSnpSet` and `BeadStudioSet`.
The viterbi algorithm is used to obtain the most likely sequence of hidden states given the observed data. The value returned is an object of class `RangedDataHMM` with genomic coordinates of the normal and altered regions. We also return the log-likelihood ratio (LLR) of the predicted sequence in an interval versus the likelihood of diploid copy number. For intervals inferred to have diploid copy number, the LLR is zero.

```
> oligoSet <- oligoSet[chromosome(oligoSet) <= 2, ]
> fit.van <- hmm(oligoSet)
```

The `GRangesList` object groups ranges by sample.

```
> names(fit.van)

[1] "NA06993"

> fit.van <- fit.van[["NA06993"]]
```

Figure 1 plots the data for chromosome 1 and the predictions from the hidden markov model:

To find which markers are included in each genomic interval returned by the `hmm` method, one can use the `findOverlaps` method in the *oligoClasses*. For example, the second interval in the `RangedDataHMM` object `fit.van` contains 102 markers.

```
> fit.van[2, ]

GRanges with 1 range and 3 elementMetadata cols:
      seqnames               ranges strand | numberProbes     state
         <Rle>            <IRanges>  <Rle> |    <integer> <integer>
  [1]     chr1 [49825223, 54637167]      * |          102         4
                  LLR
              <numeric>
  [1] 18.5161306800502
```

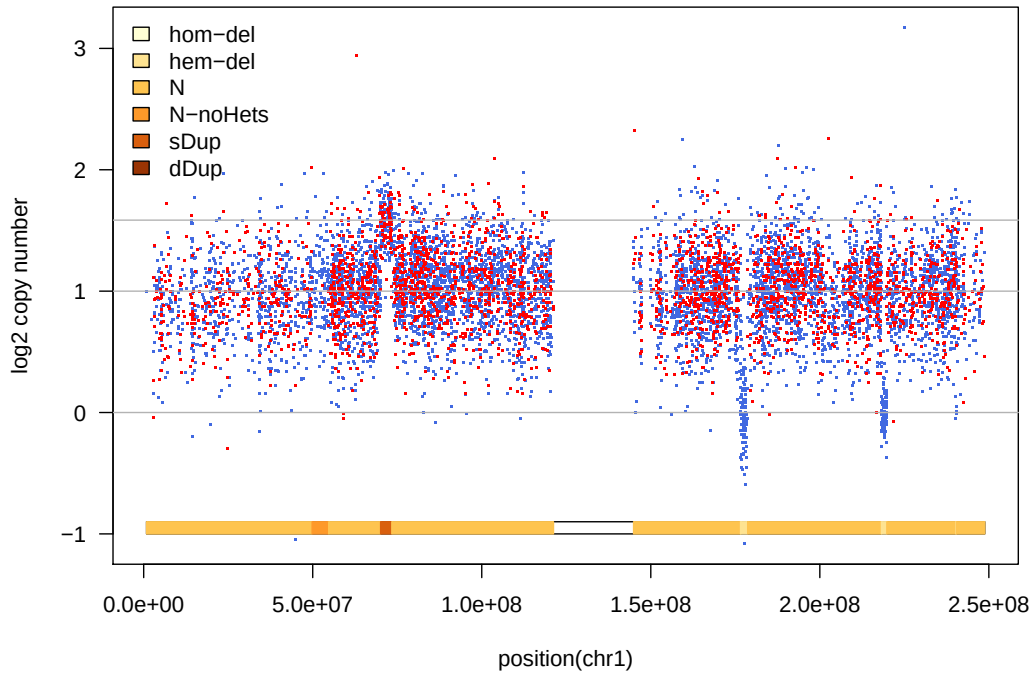


Figure 1: Plot of artificial data for one chromosome.

```
---
seqlengths:
      chr1      chr2
 249250621 243199373
```

To find the names of the 102 markers that are included in this genomic interval, one could do the following

```
> frange <- makeFeatureGRanges(oligoSet)
> mm <- as.matrix(findOverlaps(fit.van, frange))
> markersInRange <- featureNames(oligoSet)[mm[,1]==2]
```

Multipanel displays can be useful for visualizing the low-level data for copy number alterations. We extend the `xyplot` method in the R package lattice for two common use-cases: by-locus and by-sample.

By locus. To plot the genomic data for a set of ranges at a given locus, we provide an unevaluated code chunk below (our example contains only a single sample):

```
> xyplot(cn ~ x | id, oligoSet, range=RangedDataObject, ylim=c(-0.5,4), panel=xypanel)
```

Note that the default in the above command is to use a common x- and y-scale for each panel. To allow the x-axes to change for each panel, one could set the x-scales to 'free':

```
> xyplot(cn ~ x | id, oligoSet, range=RangedDataObject, ylim=c(-0.5,4), scales=list(x="free"),
+        panel=xypanel)
```

The function `xypanel` provides default colors for annotating the plotting symbols by genotype and by whether the markers are polymorphic. The `RangedDataHMM` must be passed by the name *range* to the `xyplot` method.

By sample. To plot the low-level data for multiple alterations occurring in a single sample, one can again pass a `RangedDataHMM` object with name *range* to the `xyplot`. For example, see Figure 2. The code for producing Figure 2 is in the following code chunk. Note that the formula in the example below conditions on *range* instead of *id*. The conditioning variable for displaying multiple panels of a single sample must be called 'range'. We plot a 2 Mb window surrounding each of the alterations in the simulated data by specifying `frame=2e6`.

```
> ranges.altered <- fit.van[!state(fit.van) %in% c(3,4) & coverage2(fit.van) > 5, ]
> elementMetadata(ranges.altered)$sampleId <- sampleNames(oligoSet)
> xy.example <- xyplot(cn ~ x | range, oligoSet, range=ranges.altered, frame=2e6,
+                      scales=list(x="free"), ylim=c(-1,3),
+                      panel=xypanel, cex.pch=0.4, pch=21, border="grey",
+                      ylab=expression(log[2]("copy number")))
```

See `?xyplot` for additional details.

Note also that `scales` must be set to *free* in the above call to `xyplot`.

3.1 ICE HMM

This will likely be phased out in favor of using B allele frequencies instead of genotype calls / call probabilities.

To compute emission probabilities that incorporate the *crlmm* genotype confidence scores, we set `ICE` to `TRUE`. This option is limited too a few platforms and the Affy 100k platforms is not one of them.

```
> VanillaICE:::icePlatforms()
```

```
[1] "pd.genomewidesnp.6"
[2] "genomewidesnp6"
[3] "genomewidesnp6Crlmm"
[4] "pd.mapping250k.nsp"
[5] "pd.mapping250k.sty"
[6] "pd.mapping250k.nsp, pd.mapping250k.sty"
```

For illustration, we assign 'genomewidesnp6' to the annotation slot since this platform is supported.

```
> ann <- annotation(oligoSet)
> annotation(oligoSet) <- "genomewidesnp6"
> fit.ice <- hmm(oligoSet, ICE=TRUE)
> fit.ice
```

```
GRangesList of length 1:
$NA06993
GRanges with 20 ranges and 3 elementMetadata cols:
       seqnames                ranges strand   | numberProbes
          <Rle>             <IRanges>  <Rle>   |    <integer>
   [1]     chr1 [   846864,  49772452]      *   |          996
   [2]     chr1 [ 49825223,  54637167]      *   |          102
   [3]     chr1 [ 54726362,  70065480]      *   |          902
   [4]     chr1 [ 70081878,  73381312]      *   |          202
```

```
   [5]     chr1 [ 73401482, 121226982]      *   |         2499
   [6]     chr1 [144908589, 176547669]      *   |         1294
   [7]     chr1 [176548473, 178534221]      *   |          101
   [8]     chr1 [178659933, 218173294]      *   |         1898
   [9]     chr1 [218219379, 219806187]      *   |           99
   ...      ...                    ...     ... ...          ...
  [12]     chr1 [240363241, 248794371]      *   |          160
  [13]     chr2 [   170616,  68466618]      *   |           33
  [14]     chr2 [ 75654059,  79296379]      *   |            3
  [15]     chr2 [ 81048974,  88467934]      *   |            4
  [16]     chr2 [102998279, 105880487]      *   |            2
  [17]     chr2 [114471782, 114471782]      *   |            1
  [18]     chr2 [115837414, 200784238]      *   |           41
  [19]     chr2 [206226740, 234747414]      *   |           15
  [20]     chr2 [240767758, 240767758]      *   |            1
           state               LLR
       <integer>         <numeric>
   [1]         3                 0
   [2]         4  18.5161306800502
   [3]         3                 0
   [4]         5  391.815641097378
   [5]         3                 0
   [6]         3                 0
   [7]         2  296.983405439314
   [8]         3                 0
   [9]         2   350.84947229125
   ...       ...               ...
  [12]         3                 0
  [13]         3                 0
  [14]         5 -2.00457337117874
  [15]         2              <NA>
  [16]         5   1.7371256691648
  [17]         2                 0
  [18]         3                 0
  [19]         5  -2.1622947963438
  [20]         2                 0

---
seqlengths:
      chr1      chr2
 249250621 243199373
```

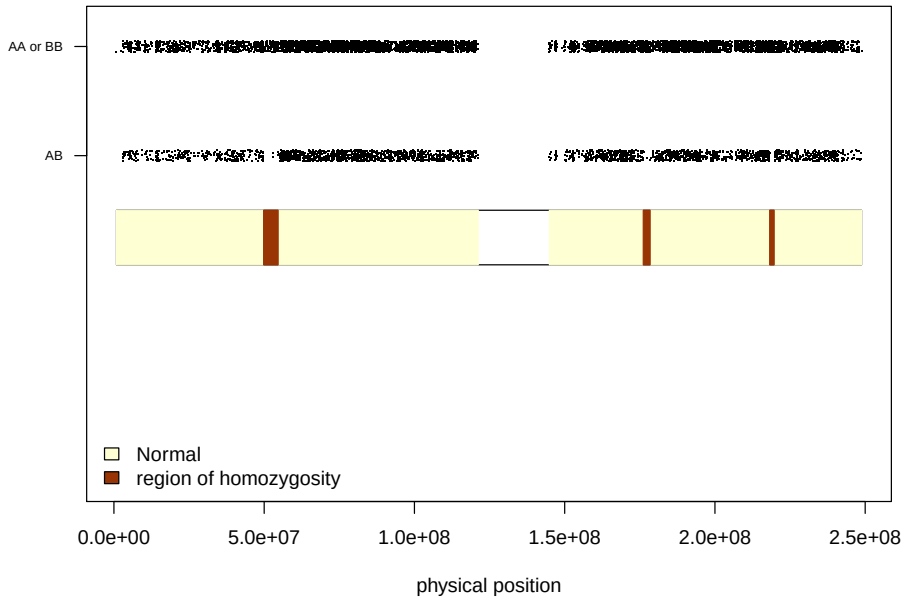
```
> annotation(oligoSet) <- ann
```

3.2 Other options

Regions of homozygosity. A HMM for genotype-only data can be used to find long stretches of homozygosity. Note that hemizygous deletions are also identified as 'ROH' when copy number is ignored (as the biallelic genotypte call in a hemizygous deletions tends to be all homozygous calls).

```
> snpSet <- as(oligoSet, "SnpSet2")
> fit.gt <- hmm(snpSet, prGtHom=c(0.7, 0.99), normalIndex=1L, S=2L)
```

A suggested visualization:



4 Trouble shooting

Read in previously saved data for chromosome 1:

```
> bset <- readRDS(file.path(system.file("extdata", package="VanillaICE"), "bset.rds"))
```

The hmm is fit independently to each arm. We can determine the arm for each marker in the `bset` object using the function `getArm`.

```
> library(oligoClasses)
> arm <- getArm(bset)
> table(arm)

arm
chr1p chr1q
12277 10756
```

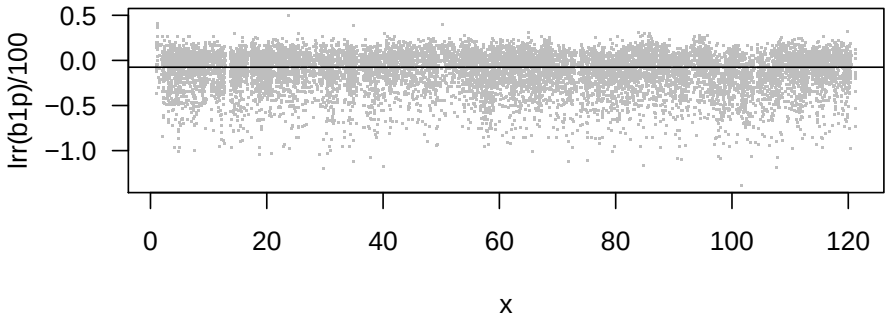
We will fit the hmm to arm 'chr1p' and monitor the updates to the mean parameters for the copy number states by setting `verbose=TRUE`:

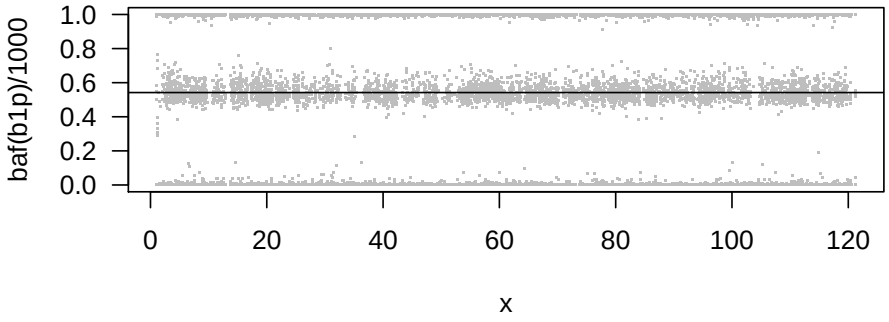
```
> b1p <- bset[arm=="chr1p", ]
> fit1p <- hmm(b1p, verbose=TRUE)

        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.3333333 0.5000000 0.6666667 0.9000000 1.0000000
[1] -2.0 -0.4  0.0  0.0  0.5  1.0
         A       AAAB        AAB         AB        ABB       ABBB
```

```
0.00000000 0.06111378 0.33333333 0.54085415 0.67370390 0.90000000
         B
1.00000000
[1] -2.0000000 -0.4000000 -0.0662899 -0.0662899  0.2292503  1.0000000
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.3333333 0.5410469 0.6666667 0.9000000 1.0000000
[1] -2.00000000 -0.43098096 -0.07268442 -0.07268442  0.21192707
[6]  1.00000000
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.3333333 0.5413561 0.6666667 0.9000000 1.0000000
[1] -2.00000000 -0.50696980 -0.07458662 -0.07458662  0.20673279
[6]  1.00000000
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.3333333 0.5418140 0.6666667 0.9000000 1.0000000
[1] -2.00000000 -0.57890024 -0.07547437 -0.07547437  0.20563497
[6]  1.00000000
```

Each line beginning with the header "A AAAB " is one iteration of the Viterbi algorithm. The numeric vector below the header is the mean for the different genotype states. Note that we use the same mean for homozygous genotypes (mean of A = mean of AA = mean of AAA). The next line is the mean of the log R ratios for copy number states homozygous deletion, hemizygous deletion, diploid, copy-neutral homozygosity, single copy gain, and two-copy gain, respectively. We can plot the data and see whether the means in the last iteration are consistent with what we observe.

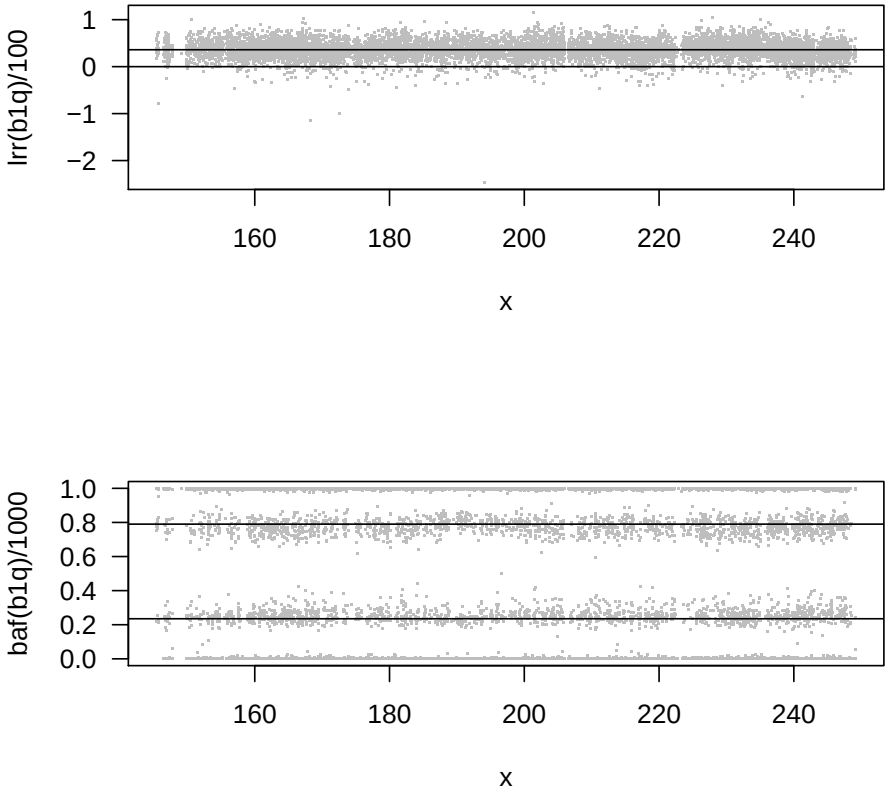
```
> par(mfrow=c(2,1), las=1)
> x <- position(b1p)/1e6
> plot(x, lrr(b1p)/100, pch=".", col="grey")
> abline(h=-0.07547)
> plot(x, baf(b1p)/1000, pch=".", col="grey")
> abline(h=0.542)
```






```
> b1q <- bset[arm=="chr1q", ]
> fit1q <- hmm(b1q, verbose=TRUE)

        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.3333333 0.5000000 0.6666667 0.9000000 1.0000000
[1] -2.0 -0.4  0.0  0.0  0.5  1.0
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1460467 0.2559652 0.5000000 0.7666329 0.8487884 1.0000000
[1] -2.0000000 -0.4000000  0.0000000  0.0000000  0.3915698  0.6335398
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.2519157 0.5000000 0.7753510 0.9000000 1.0000000
[1] -2.0000000 -0.4000000  0.0000000  0.0000000  0.3667183  0.6335398
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.2474432 0.5000000 0.7793528 0.9000000 1.0000000
[1] -2.0000000 -0.4000000  0.0000000  0.0000000  0.3613872  0.6335398
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.2416484 0.5000000 0.7844886 0.9000000 1.0000000
[1] -2.0000000 -0.4000000  0.0000000  0.0000000  0.3601088  0.6335398
        A      AAAB       AAB        AB       ABB      ABBB         B
0.0000000 0.1000000 0.2354266 0.5000000 0.7901539 0.9000000 1.0000000
[1] -2.0000000 -0.4000000  0.0000000  0.0000000  0.3597777  0.6335398
```

```
> par(mfrow=c(2,1), las=1)
> x <- position(b1q)/1e6
> plot(x, lrr(b1q)/100, pch=".", col="grey")
> abline(h=c(0, .359778))
> plot(x, baf(b1q)/1000, pch=".", col="grey")
> abline(h=c(0.235, 0.790))
```



5 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 2.15.1 (2012-06-22), `x86_64-apple-darwin11.4.0`
- Locale:
  `en_US.US-ASCII/en_US.US-ASCII/en_US.US-ASCII/C/en_US.US-ASCII/en_US.US-ASCII`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: Biobase 2.17.6, BiocGenerics 0.3.0, DBI 0.2-5, IRanges 1.15.19, oligo 1.21.4, oligoClasses 1.19.36, pd.mapping50k.hind240 1.6.0, pd.mapping50k.xba240 1.6.0, RColorBrewer 1.0-5, RSQLite 0.11.1, SNPchip 2.3.11, VanillaICE 1.19.28

- Loaded via a namespace (and not attached): affxparser 1.29.6, affyio 1.25.0, annotate 1.35.3, AnnotationDbi 1.19.20, BiocInstaller 1.5.11, Biostrings 2.25.6, bit 1.1-8, codetools 0.2-8, compiler 2.15.1, crlmm 1.15.12, ellipse 0.3-7, ff 2.2-7, foreach 1.4.0, genefilter 1.39.0, GenomicRanges 1.9.30, grid 2.15.1, iterators 1.0.6, lattice 0.20-6, msm 1.1.1, mvtnorm 0.9-9992, preprocessCore 1.19.0, splines 2.15.1, stats4 2.15.1, survival 2.36-14, tools 2.15.1, XML 3.9-4, xtable 1.7-0, zlibbioc 1.3.0

References

[1] Stefano Colella, Christopher Yau, Jennifer M Taylor, Ghazala Mirza, Helen Butler, Penny Clouston, Anne S Bassett, Anneke Seller, Christopher C Holmes, and Jiannis Ragoussis. QuantiSNP: an Objective Bayes Hidden-Markov Model to detect and accurately map copy number variation using SNP genotyping data. *Nucleic Acids Res*, 35(6):2013–2025, 2007.

[2] Adam B Olshen, E. S. Venkatraman, Robert Lucito, and Michael Wigler. Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics*, 5(4):557–72, Oct 2004.

[3] Kai Wang, Mingyao Li, Dexter Hadley, Rui Liu, Joseph Glessner, Struan F A Grant, Hakon Hakonarson, and Maja Bucan. Penncnv: an integrated hidden markov model designed for high-resolution copy number variation detection in whole-genome snp genotyping data. *Genome Res*, 17(11):1665–1674, Nov 2007.

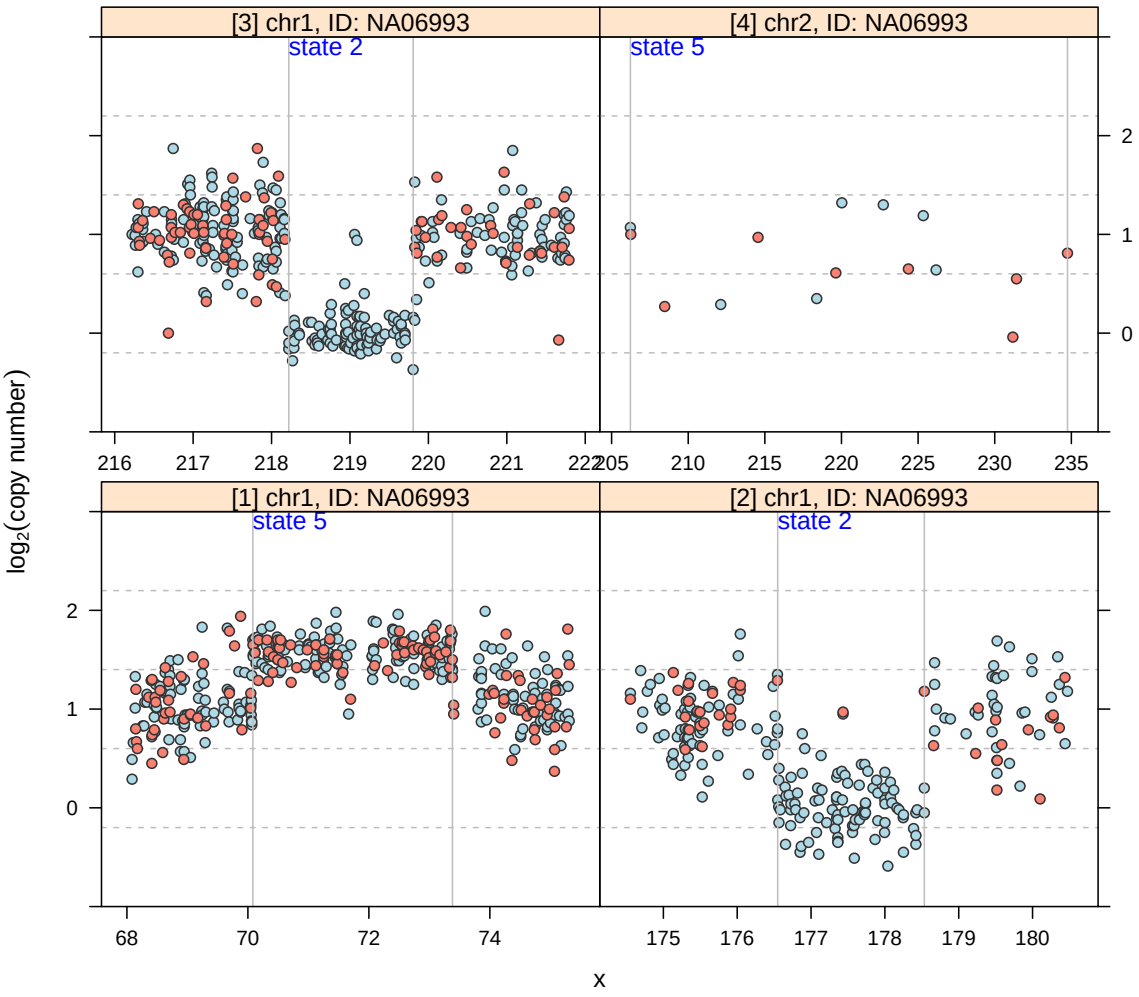


Figure 2: The method `xyplot` is used to create a multi-panel display of alterations in a single sample. Each panel displays a single copy number alteration detected by the HMM that is boxed by a rectangle. The alteration is framed by specifying the number of basepairs to plot upstream and downstream of the alteration. Here, we used a frame of 2 Mb. Homozygous SNPs with diallelic geotypes 'AA' and 'BB' are shaded blue; SNPs with diallelic genotype call 'AB' are shaded in red.