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

Robert Scharpf

February 18, 2010

Abstract

Chromosomal DNA is characterized by variation between individuals at the level of entire chromosomes (e.g. aneuploidy in which the chromosome copy number is altered), segmental changes (including insertions, deletions, inversions, and translocations), and changes to small genomic regions (including single nucleotide polymorphisms). A variety of alterations that occur in chromosomal DNA, many of which can be detected using high density single nucleotide polymorphism (SNP) microarrays, are linked to normal variation as well as disease and therefore of particular interest. These include changes in copy number (deletions and duplications) and genotype (e.g. the occurrence of regions of homozygosity). Hidden Markov models (HMM) are particularly useful for detecting such abnormalities, modeling the spatial dependence between neighboring SNPs. Here, we extend previous approaches that utilize HMM frameworks for inference in high throughput SNP arrays by integrating copy number, genotype calls, and the corresponding measures of uncertainty when available. Using simulated and experimental data, we demonstrate how confidence scores control smoothing in a probabilistic framework.

1 Overview

This vignette describes how to fit a hidden Markov model to locus-level estimates of genotype and/or copy number. 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 genotype calls (1=AA, 2 = AB, 3= BB): rows correspond to loci and columns correspond to samples

Additional options that can improve the HMM predictions include

- a CRLMM confidence score of the genotype call
- standard errors of the copy number estimates

If using the R package *crlmm*, see the *copynumber* vignette in `crlmm/inst/scripts` for locus-level estimation of copy number. Several HMM implementations are now available for the joint analysis of copy number and genotype, including QuantiSNP [2] and PennCNV [5].

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.

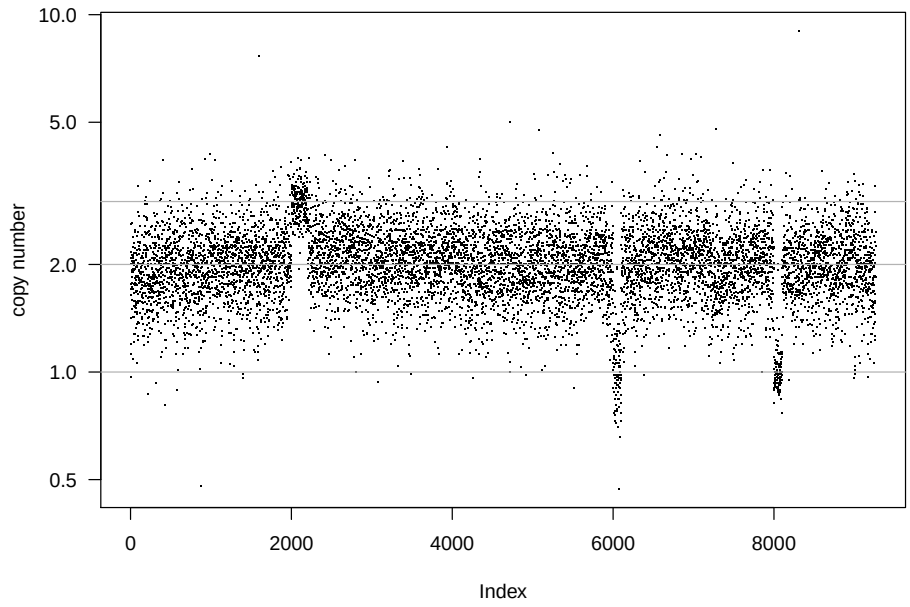
2 Organizing the locus-level data

This package includes simulated genotype and copy number data for approximately 9165 SNPs on chromosome 1 and 100 SNPs on chromosome 2.

```
> library(VanillaICE)
> data(locusLevelData)
```

(The copy number estimates in the locusLevelData object were multiplied by 100 and saved as an integer.) Verify that it is reasonable to assume integer copy number for the HMM by plotting the locus-level estimates as a function of the physical position.

```
> par(las = 1)
> plot(locusLevelData[["copynumber"]][, 1]/100, pch = ".", ylab = "copy number",
+     log = "y")
> abline(h = 1:3, col = "grey70")
```



Next, create an object of class `oligoSnpSet` from the simulated data:

```
> oligoSet <- new("oligoSnpSet", copyNumber = locusLevelData[["copynumber"]]/100,
+     call = locusLevelData[["genotypes"]], callProbability = locusLevelData[["crlmmConfidence"]],
+     annotation = locusLevelData[["platform"]])
```

If confidence scores or inverse standard errors for the copy number estimates are available, these should be supplied to the `cnConfidence` slot in the `assayData`. For illustration, in the following code chunk we transform the copy number estimates to the log scale and calculate a robust estimate of the standard deviation. If uncertainty estimates are not available for copy number, the HMM will calculate robust estimates of the standard deviation as per the code chunk below (assigning confidence scores to the cnConfidence slot is only needed if estimates of uncertainty are available from the software used to obtain copy number estimates).

```
> sds <- VanillaICE:::robustSds(log2(locusLevelData[["copynumber"]]/100))
> oligoSet2 <- new("oligoSnpSet", copyNumber = log2(locusLevelData[["copynumber"]]/100),
+     cnConfidence = 1/sds, call = locusLevelData[["genotypes"]],
+     callProbability = locusLevelData[["crlmmConfidence"]], annotation = locusLevelData[["platform"]]
```

3 Fitting the HMM

Several scenarios are outlined for fitting the HMM. In particular, one may model the genotype and copy number estimates jointly, or each separately.

In general, the following elements are required to fit the HMM:

1. initial state probabilities
2. emission probabilities
3. transition probabilities

Two approaches for computing the emission probabilities are available in the R package *VanillaICE*. One approach is to model the copy number estimates and genotype calls jointly to identify hemizygous deletions, normal copy number, retention of heterozygosity, and amplifications. A second approach is to compute the emisson probabilities directly from the bivariate normal prediction regions for integer copy number on the log A versus log B scale [3]. The latter approach is available only if the *crlmm* package was used to preprocess the raw intensities. See the crlmmDownstream vignette available with this package for additional information, and the technical report describing the estimation procedure for copy number in crlmm [4]. In the remainder of this vignette, we assume that alternative methods were used to obtain the total copy number and/or genotypes.

3.1 Joint HMM for genotype and total copy number

For computing emission probabilities from genotype and total copy number, one can incorporate confidence estimates of the genotype calls if the CRLMM algorithm was used for genotyping [1]. (The R packages *crlmm* and *oligo* both implement the CRLMM genotyping algorithm.) The *vanilla* HMM does not use confidence scores for the genotype calls, whereas the *ICE* HMM integrates the confidence estimates of the genotypes for computing the emission probabilities.

The vanilla HMM. In the following code chunk, we assume the hidden states are hemizygous deletion (hemDel; copy number = 1, probability of a homozygous genotype call is 0.999), normal (copy number = 2, probability of a homozygous genotype calls is 0.7), regions of homozygosity (ROH: copy number = 1, probability of a homozygous genotype call is 0.999), and amplification (copy number greater than 2). The primary assumptions of the HMM are (1) that the copy number estimates are approximately Gaussian and (2) that the hidden state is an integer and the copy number estimates are relatively unbiased (though perhaps very noisy). Transforming the estimates to the log-scale is often helpful.

```
> copyNumber(oligoSet) <- log2(copyNumber(oligoSet))
```

The SNPs should be ordered by chromosome and physical position.

```
> oligoSet <- oligoSet[order(chromosome(oligoSet), position(oligoSet)),
+     ]
```

The initial state probabilities, copy number states, and the probabilitiy of a homozygous genotype call for each state must be supplied as arguments to the `hmmOptions` function.

```
> hmmOpts <- hmmOptions(oligoSet, copynumberStates = log2(c(1,
+     2, 2, 3)), states = c("hem-del", "ROH", "normal", "amp"),
+     normalIndex = 3, log.initialP = rep(log(1/4), 4), prGenotypeHomozygous = c(0.99,
+         0.99, 0.7, 0.7), TAUP = 5e+07)
```

Note that specified hidden states for copy number should be on the same scale as the copy number estimates stored in the `oligoSet` object. As we had log-transformed the copy number estimates, the above code chunk log transforms the copy number hidden states. Again, the above code chunk assumes that the hidden states are integers and that the copy number estimates provide unbiased though noisy estimes of these parameters. Depending on the software and platform used for obtaining locus-level estimates of copy number, one could specify noninteger hidden states in the above code chunk. Conditional on the underlying hidden state, we assume that the copy number is independent of the genotype and we calculate the emission probabilities of each separately. The joint emission probabilities are simply the product of the emission probabilities for the genotype calls and copy number estimates.

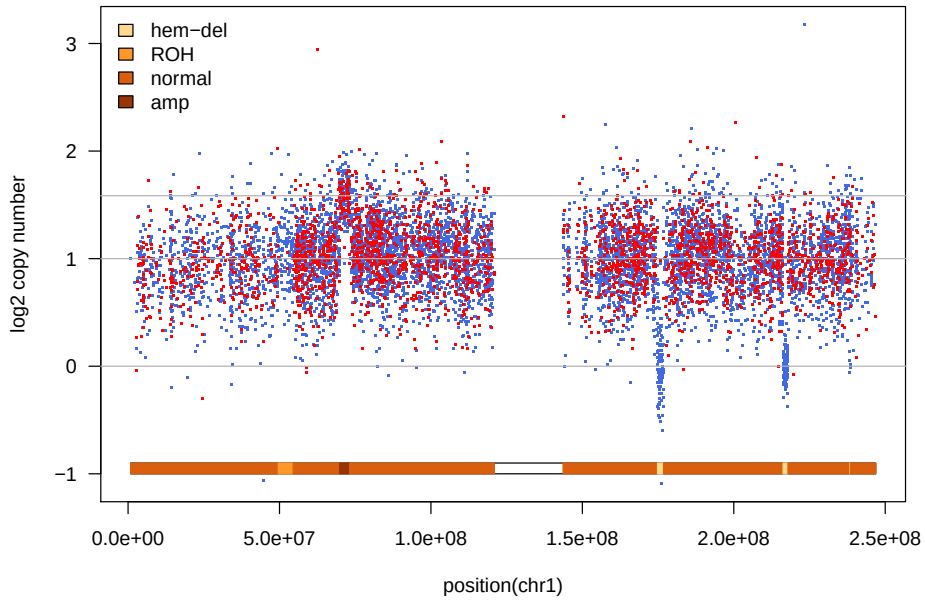
The viterbi algorithm is used to obtain the most likely sequence of hidden states given the observed data. For efficiency, we return an object of class `RangedData` with genomic coordinates of the normal and altered regions. We also return the log-likelihood ratio (LLR) of the predicted sequence in an interval versus the null of normal copy number. For intervals with typical copy number (2) and percent heterozygosity (the 3rd state in the above codechunk), the LLR is zero.

```
> fit.van <- hmm(oligoSet, hmmOpts)
> fit.van
```

```
RangedData with 20 rows and 4 value columns across 2 spaces
         space                 ranges |    sampleId     state numMarkers
   <character>              <IRanges> | <character> <integer>  <integer>
1         chr1 [   836727,  49545039] |     NA06993         3        997
2         chr1 [ 49597810,  54409755] |     NA06993         2        102
3         chr1 [ 54498950,  69838068] |     NA06993         3        902
4         chr1 [ 69854466,  73153900] |     NA06993         4        202
5         chr1 [ 73174070, 120928505] |     NA06993         3       2502
6         chr1 [143619946, 174814292] |     NA06993         3       1295
7         chr1 [174815096, 176800780] |     NA06993         1        100
8         chr1 [176800844, 216239917] |     NA06993         3       1900
9         chr1 [216286002, 217872810] |     NA06993         1         99
10        chr1 [217892177, 238302668] |     NA06993         3        901
         LLR
   <numeric>
1    0.00000
2   16.07784
3    0.00000
4  333.94107
5    0.00000
6    0.00000
7  492.73020
8    0.00000
9  523.90010
10   0.00000
...
<10 more rows>
```

```
> library(RColorBrewer)
> cols <- brewer.pal(5, "YlOrBr")[2:5]

> chr1 <- oligoSet[chromosome(oligoSet) == 1, ]
> fit.chr1 <- fit.van[space(fit.van) == "chr1", ]
> isHet <- snpCall(chr1) == 2
> par(las = 1)
> plot(position(chr1), copyNumber(chr1), pch = ".", cex = 2, col = "royalblue",
+     ylab = "log2 copy number")
> points(position(chr1)[isHet], copyNumber(chr1)[isHet], col = "red",
+     pch = ".", cex = 2)
> abline(h = log2(1:3), col = "grey70")
> sts <- start(fit.chr1)
> ends <- end(fit.chr1)
> xx <- range(c(sts, ends))
> y <- c(-1, -1, -0.9, -0.9)
> polygon(x = c(xx, rev(xx)), y = y, col = "white")
> for (i in 1:nrow(fit.chr1)) {
+     polygon(x = c(sts[i], ends[i], ends[i], sts[i]), y = y, col = cols[fit.chr1$state[i]],
+         border = cols[fit.chr1$state[i]])
+ }
> legend("topleft", fill = cols, legend = hmmOpts$states, bty = "n")
```



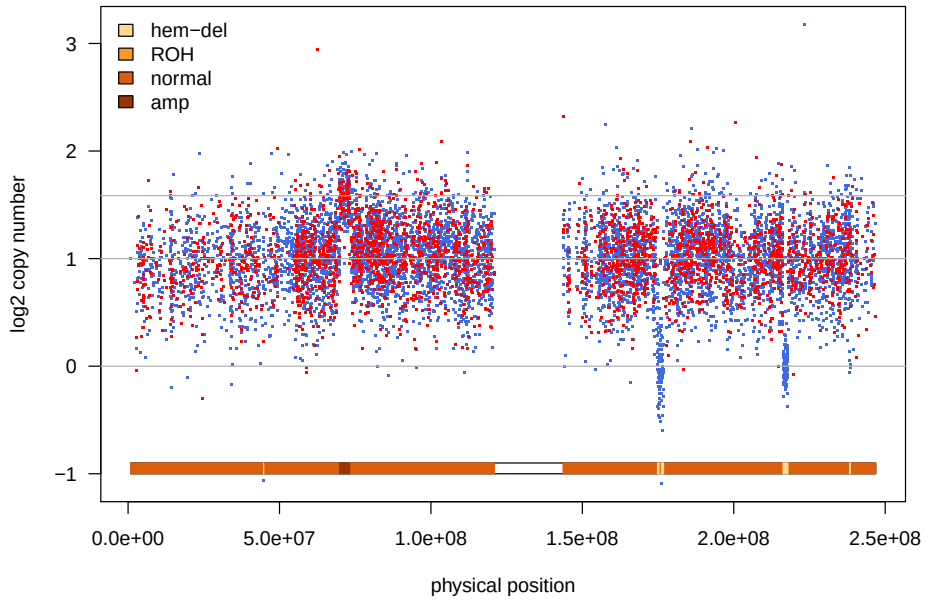
The ICE HMM . When the CRLMM algorithm is used for genotyping, one should incorporate the confidence score of the genotypes. In general, this involves estimating the probability that the genotype call was *correct* from the reference HapMap data where a gold standard is available. The probability of a correct call could vary by platform as well as the population, but here we assume such differences are small.

```
> hmmOpts <- hmmOptions(oligoSet, ICE = TRUE, copynumberStates = log2(c(1,
+     2, 2, 3)), states = c("hem-del", "ROH", "normal", "amp"),
+     normalIndex = 3, log.initialP = rep(log(1/4), 4), rohStates = c(TRUE,
+         TRUE, FALSE, FALSE), TAUP = 5e+07)
> fit.ice <- hmm(oligoSet, hmmOpts)
> fit.ice

RangedData with 22 rows and 4 value columns across 2 spaces
         space                 ranges |    sampleId     state numMarkers
   <character>              <IRanges> | <character> <integer>  <integer>
1         chr1 [   836727,  44722116] |     NA06993         3        878
2         chr1 [ 44762242,  44762242] |     NA06993         1          1
3         chr1 [ 44779351,  69838068] |     NA06993         3       1122
4         chr1 [ 69854466,  73153900] |     NA06993         4        202
5         chr1 [ 73174070, 120928505] |     NA06993         3       2502
6         chr1 [143619946, 174814292] |     NA06993         3       1295
7         chr1 [174815096, 175691958] |     NA06993         1         49
8         chr1 [175699839, 175700199] |     NA06993         3          2
9         chr1 [175726310, 176800780] |     NA06993         1         49
10        chr1 [176800844, 216239917] |     NA06993         3       1900
          LLR
    <numeric>
1    0.000000
2    2.230621
3    0.000000
4  333.941075
5    0.000000
6    0.000000
7  236.334291
8    0.000000
9  261.700167
10   0.000000
...
<12 more rows>

> fit.chr1 <- fit.ice[space(fit.ice) == "chr1", ]
> widths <- width(fit.chr1)
> fit.chr1 <- fit.chr1[order(widths, decreasing = TRUE), ]
> par(las = 1)
> plot(position(chr1), copyNumber(chr1), pch = ".", ylab = "log2 copy number",
+     xlab = "physical position", cex = 2, col = "royalblue")
> points(position(chr1)[isHet], copyNumber(chr1)[isHet], col = "red",
+     pch = ".", cex = 2)
> abline(h = log2(1:3), col = "grey70")
> sts <- start(fit.chr1)
> ends <- end(fit.chr1)
> xx <- range(c(sts, ends))
> y <- c(-1, -1, -0.9, -0.9)
> polygon(x = c(xx, rev(xx)), y = y, col = "white")
> for (i in 1:nrow(fit.chr1)) {
+     polygon(x = c(sts[i], ends[i], ends[i], sts[i]), y = y, col = cols[fit.chr1$state[i]],
+         border = cols[fit.chr1$state[i]])
```

```
+ }
> legend("topleft", fill = cols, legend = hmmOpts$states, bty = "n")
```



3.2 Copy number HMM

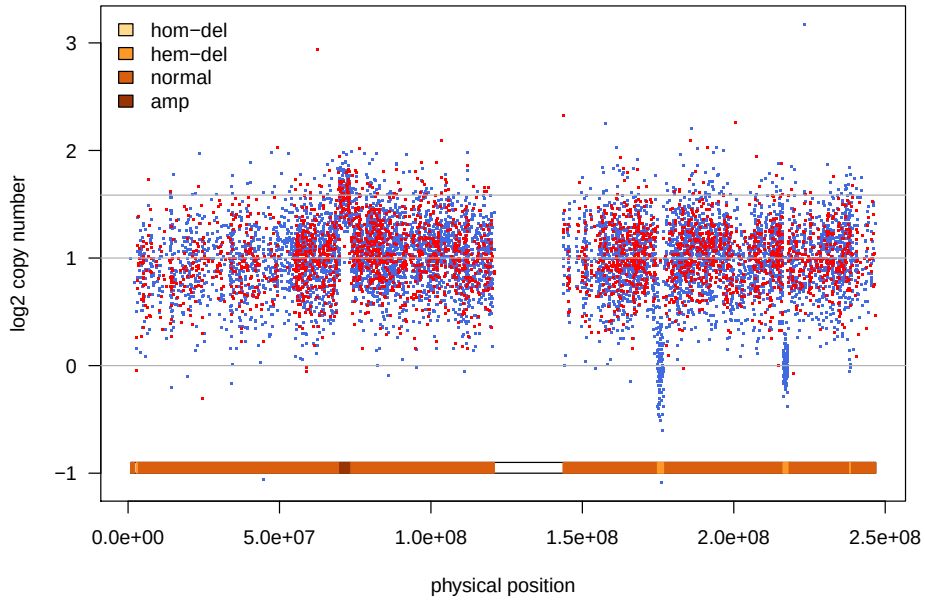
A HMM for copy number only (e.g., if genotypes are ignored or are unavailable) can be fit as follows.

```
> cnSet <- new("CopyNumberSet", copyNumber = log2(locusLevelData[["copynumber"]]/100),
+     annotation = locusLevelData[["platform"]])
> cnSet <- cnSet[order(chromosome(cnSet), position(cnSet)), ]
> hmmOpts <- hmmOptions(cnSet, copynumberStates = log2(c(0, 1,
+     2, 3)), states = c("hom-del", "hem-del", "normal", "amp"),
+     normalIndex = 3, log.initialP = rep(log(1/4), 4), TAUP = 5e+07)
> fit.cn <- hmm(cnSet, hmmOpts)
> fit.cn
```

```
RangedData with 20 rows and 4 value columns across 2 spaces
         space                 ranges |    sampleId     state numMarkers
   <character>              <IRanges> | <character> <integer>  <integer>
1         chr1 [   836727,   2224111] |     NA06993         3          2
2         chr1 [  2915399,   2941694] |     NA06993         2          4
3         chr1 [  2963436,  69838068] |     NA06993         3       1995
4         chr1 [ 69854466,  73153900] |     NA06993         4        202
5         chr1 [ 73174070, 120928505] |     NA06993         3       2502
6         chr1 [143619946, 174814292] |     NA06993         3       1295
7         chr1 [174815096, 176800844] |     NA06993         2        101
8         chr1 [176926556, 216239917] |     NA06993         3       1899
9         chr1 [216286002, 217872810] |     NA06993         2         99
```

```
10        chr1 [217892177, 238302668] |     NA06993         3        901
          LLR
    <numeric>
1    0.000000
2    1.138391
3    0.000000
4  333.941075
5    0.000000
6    0.000000
7  464.877880
8    0.000000
9  489.663497
10   0.000000
...
<10 more rows>
```

```
> fit.chr1 <- fit.cn[space(fit.cn) == "chr1", ]
> widths <- width(fit.chr1)
> fit.chr1 <- fit.chr1[order(widths, decreasing = TRUE), ]
> par(las = 1)
> plot(position(chr1), copyNumber(chr1), pch = ".", ylab = "log2 copy number",
+     xlab = "physical position", cex = 2, col = "royalblue")
> points(position(chr1)[isHet], copyNumber(chr1)[isHet], col = "red",
+     pch = ".", cex = 2)
> abline(h = log2(1:3), col = "grey70")
> sts <- start(fit.chr1)
> ends <- end(fit.chr1)
> xx <- range(c(sts, ends))
> y <- c(-1, -1, -0.9, -0.9)
> polygon(x = c(xx, rev(xx)), y = y, col = "white")
> for (i in 1:nrow(fit.chr1)) {
+     polygon(x = c(sts[i], ends[i], ends[i], sts[i]), y = y, col = cols[fit.chr1$state[i]],
+         border = cols[fit.chr1$state[i]])
+ }
> legend("topleft", fill = cols, legend = hmmOpts$states, bty = "n")
```



3.3 Region of homozygosity (ROH) HMM

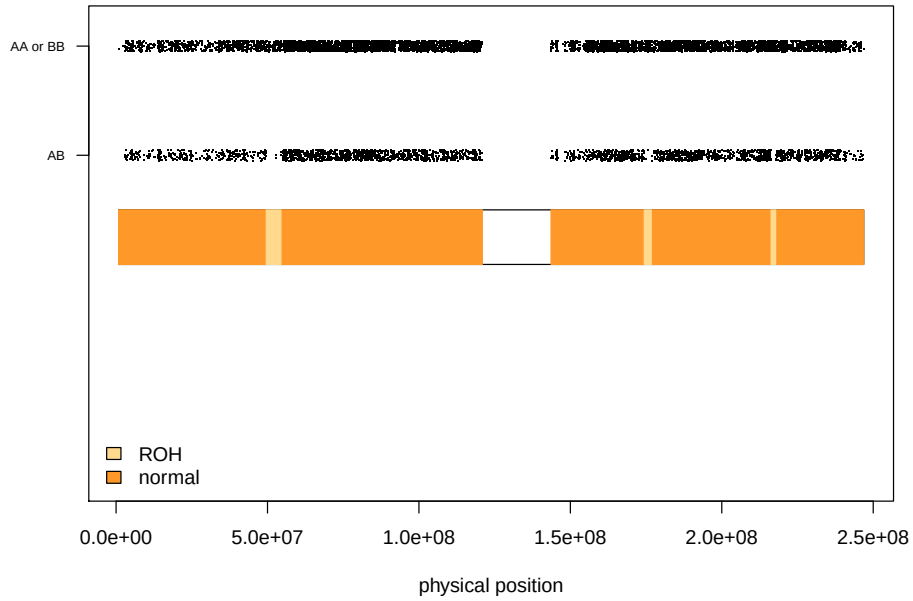
A HMM for genotype-only data can be used to find long stretches of homozygosity. Note that hemizygous deletions are also identified as 'ROH' when copy number is ignored (as the biallelic genotypte call in a hemizygous deletions tends to be all homozygous calls).

```
> snpSet <- new("SnpSet", call = locusLevelData[["genotypes"]],
+     callProbability = locusLevelData[["crlmmConfidence"]], annotation = locusLevelData[["platform"]]
> snpSet <- snpSet[order(chromosome(snpSet), position(snpSet)),
+     ]
> hmmOpts <- hmmOptions(snpSet, states = c("ROH", "normal"), normalIndex = 2,
+     log.initialP = rep(log(1/2), 2), prGenotypeHomozygous = c(0.99,
+         0.7), TAUP = 5e+07)
> fit.gt <- hmm(snpSet, hmmOpts)
> fit.gt
```

```
RangedData with 10 rows and 4 value columns across 2 spaces
         space                 ranges |    sampleId     state numMarkers
   <character>              <IRanges> | <character> <integer>  <integer>
1         chr1 [   836727,  49545039] |     NA06993         2        997
2         chr1 [ 49597810,  54409755] |     NA06993         1        102
3         chr1 [ 54498950, 120928505] |     NA06993         2       3606
4         chr1 [143619946, 174309008] |     NA06993         2       1284
5         chr1 [174418117, 176704067] |     NA06993         1        109
6         chr1 [176800399, 216239917] |     NA06993         2       1902
7         chr1 [216286002, 217771828] |     NA06993         1         97
8         chr1 [217872013, 246860994] |     NA06993         2       1068
9         chr2 [   160616,  88249049] |     NA06993         2         40
```

```
10        chr2 [102364711, 240432695] |     NA06993         2         60
         LLR
   <numeric>
1    0.00000
2   16.06643
3    0.00000
4    0.00000
5   19.29743
6    0.00000
7   21.80976
8    0.00000
9    0.00000
10   0.00000
```

```
> fit.chr1 <- fit.gt[space(fit.gt) == "chr1", ]
> widths <- width(fit.chr1)
> fit.chr1 <- fit.chr1[order(widths, decreasing = TRUE), ]
> gt <- ifelse(snpCall(chr1) == 1 | snpCall(chr1) == 3, 1, 0)
> par(las = 1)
> plot(position(chr1), jitter(gt, amount = 0.05), pch = ".", ylab = "",
+     xlab = "physical position", ylim = c(-3, 1.2), yaxt = "n")
> axis(side = 2, at = c(0, 1), labels = c("AB", "AA or BB"), cex.axis = 0.7)
> sts <- start(fit.chr1)
> ends <- end(fit.chr1)
> xx <- range(c(sts, ends))
> y <- c(-1, -1, -0.5, -0.5)
> polygon(x = c(xx, rev(xx)), y = y, col = "white")
> for (i in 1:nrow(fit.chr1)) {
+     polygon(x = c(sts[i], ends[i], ends[i], sts[i]), y = y, col = cols[fit.chr1$state[i]],
+         border = cols[fit.chr1$state[i]])
+ }
> legend("bottomleft", fill = cols, legend = hmmOpts$states, bty = "n")
```



4 Session Information

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

- R version 2.11.0 Under development (unstable) (2009-11-22 r50541), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.iso885915`, `LC_NUMERIC=C`, `LC_TIME=en_US.iso885915`, `LC_COLLATE=en_US.iso885915`, `LC_MONETARY=C`, `LC_MESSAGES=en_US.iso885915`, `LC_PAPER=en_US.iso885915`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.iso885915`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: Biobase 2.7.3, crlmm 1.5.22, DBI 0.2-4, IRanges 1.5.34, oligoClasses 1.9.27, pd.mapping50k.hind240 0.4.1, pd.mapping50k.xba240 0.4.1, RColorBrewer 1.0-2, RSQLite 0.7-3, VanillaICE 1.9.1
- Loaded via a namespace (and not attached): affyio 1.15.2, annotate 1.25.0, AnnotationDbi 1.9.2, Biostrings 2.15.11, ellipse 0.3-5, genefilter 1.29.3, mvtnorm 0.9-8, preprocessCore 1.9.0, SNPchip 1.11.1, splines 2.11.0, survival 2.35-7, tools 2.11.0, xtable 1.5-6

References

[1] Benilton Carvalho, Henrik Bengtsson, Terence P Speed, and Rafael A Irizarry. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. *Biostatistics*, 8(2):485–499, Apr 2007.

[2] 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.

[3] Joshua M Korn, Finny G Kuruvilla, Steven A McCarroll, Alec Wysoker, James Nemesh, Simon Cawley, Earl Hubbell, Jim Veitch, Patrick J Collins, Katayoon Darvishi, Charles Lee, Marcia M Nizzari, Stacey B Gabriel, Shaun Purcell, Mark J Daly, and David Altshuler. Integrated genotype calling and association analysis of snps, common copy number polymorphisms and rare cnvs. *Nat Genet*, 40(10):1253–1260, Oct 2008.

[4] Robert B Scharpf, Ingo Ruczinski, Benilton Carvalho, Betty Doan, Aravinda Chakravarti, and Rafael Irizarry. A multilevel model to address batch effects in copy number estimation using snp arrays. Submitted Revised Manuscript, Jan 2010.

[5] 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.