9-29-2010

USING THE R PACKAGE crlmm FOR GENOTYPING AND COPY NUMBER ESTIMATION

Robert B. Scharpf
Johns Hopkins Bloomberg School of Public Health, Department of Biostatistics, rscharpf@jhsph.edu

Rafael Irizarry
Johns Hopkins Bloomberg School of Public Health, Department of Biostatistics

Walter Ritchie
Walter-Eliza Hall Institute of Medical Research

Benilton Carvalho
University of Cambridge

Ingo Ruczinski
Johns Hopkins Bloombeg School of Public Health, Department of Biostatistics

Using the R Package crlmm for Genotyping and Copy Number Estimation

Robert B Scharpf
Johns Hopkins University

Rafael A Irizarry
Johns Hopkins University

Matt Ritchie
Walter+Eliza Hall Institute of Medical Research

Benilton Carvalho
University of Cambridge

Ingo Ruczinski
Johns Hopkins University

Abstract

Genotyping platforms such as Affymetrix can be used to assess genotype-phenotype as well as copy number-phenotype associations at millions of markers. While genotyping algorithms are largely concordant when assessed on HapMap samples, tools to assess copy number changes are more variable and often discordant. One explanation for the discordance is that copy number estimates are susceptible to systematic differences between groups of samples that were processed at different times or by different labs. Analysis algorithms that do not adjust for batch effects are prone to spurious measures of association. The R package **crlmm** implements a multilevel model that adjusts for batch effects and provides allele-specific estimates of copy number. This paper illustrates a workflow for the estimation of allele-specific copy number, develops marker- and study-level summaries of batch effects, and demonstrates how the marker-level estimates can be integrated with complimentary Bioconductor software for inferring regions of copy number gain or loss. All analyses are performed in the statistical environment R. A compendium for reproducing the analysis is available from the author's website (http://www.biostat.jhsph.edu/~rscharpf/crlmmCompendium/index.html).

Keywords: copy number, batch effects, robust, multilevel model, high-throughput, oligonucleotide array.

1. Introduction

Duplications and deletions spanning kilobases of the genome contribute to a substantial proportion of the genetic variation between individuals. Copy number variants (CNV) account for a greater proportion of differences in terms of sequence composition between two individuals than single nucleotide polymorphisms (SNPs) (Zhang *et al.* 2009). CNV can arise through a number of mechanisms during meiosis and mitosis and are well known to be implicated in cancer through deletions that disrupt tumor suppressor genes or the amplification of oncogenes. Copy number alterations have also been implicated in several genomic disorders,

including complex diseases such as schizophrenia and autism (Karayiorgou *et al.* 2010; Pinto *et al.* 2010).

Current estimates regarding the frequency and size of segmental duplications and deletions in the human genome are largely based on high-throughput arrays that quantitate copy number on a genomic scale. Two such technologies are array comparative genomic hybridization (aCGH) and *genotyping* platforms such as the Affymetrix oligonucleotide arrays and the Illumina BeadArrays. While each of these platforms rely on the hybridization of probes to sample preparations containing target DNA sequence, differences exist in the size of the probes, the number of probes per target sequence, and whether the hybridization is competitive. Unlike aCGH, genotyping arrays can be used to identify copy-neutral regions of homozygosity that, while common in apparently normal individuals, can suggest rare genetic events such as uniparental isodisomy (UPD). UPD has been implicated in heritable diseases such as Prader-Willi syndrome (Altug-Teber *et al.* 2005). While the resolution is potentially much greater in genotyping arrays due to the shorter probe length, shorter probe lengths tend to result in more probe-to-probe variability with respect to cross-hybridization to the alternative allele, nonspecific binding, and differences in basepair composition. Reliable inference of copy number gain or loss at a single 25 - 100 basepair locus is not currently possible, and statistical methods that smooth the locus-level estimates as a function of the physical position in the genome are needed.

Despite robust-to-outlier approaches for normalization, we have observed systematic differences in the copy number between groups of samples that can be perfectly predicted by the timestamp on the CEL files. We refer to such systematic difference in copy number between groups of samples as batch effects. That larger studies tend to have more substantial batch effects than smaller studies is consistent with our conjecture that the nonstatic nature of experimental reagents and laboratory conditions contribute over time to batch effects. Irrespective of etiology, we have found that the scan date of the array and chemistry plate are useful surrogates for batch (Scharpf *et al.* 2010). With an appropriate experimental design that involves randomization of samples to chemistry plate, batch effects are a nuisance variable that can be successfully modeled and removed.

Existing analytic strategies for identifying alterations in copy number have largely adopted a one- or two-step approach. In the one-step approach, assessments of CNV are made from the raw intensities using the joint distribution across samples. For instance, Zhang *et al.* (2009) developed a Correlation Matrix Diagonal Segmentation (CMDS) that identifies recurrent alterations in a population. While we have not formally evaluated the impact of batch effects using this approach, it is important to note that the differences in raw intensities between groups of samples, whether driven by biological causes or by technological artifacts such as batch effects, are similar in terms of their effects on the data. A safe strategy when adopting such an approach would be to filter loci associated with experimental factors such as chemistry plate or scan date.

In contrast to the one-step approach, two-step approaches generally derive estimates of copy number and uncertainty at each marker, followed by smoothing of the marker-level estimates at the second stage. The motivation for the two-step approach is that the marker-level estimates are too imprecise to provide reliable copy number estimates. However, marker-specific estimates can be useful for at least two reasons. First, single-locus estimates are typically derived from the joint distribution of intensities across samples and, through inspection of the joint distribution, batch effects can be modeled and removed. Secondly, plots of the marker-

level estimates can be useful for assessing copy number mosaicism. Mosaicism occurs when mixtures of cell populations with different mutations give rise to noninteger copy number estimates. For instance, many tumors are comprised of a mixture of cell populations representing different levels of tumor evolution. The choice of appropriate statistical methods for smoothing at the second stage can therefore be informed by visualizations of the marker-level estimates. In particular, hidden Markov models (HMMs) (Fridlyand *et al.* 2004; Colella *et al.* 2007; Wang *et al.* 2007; Scharpf *et al.* 2008) are generally more appropriate for germline diseases in which latent, integer copy number states are reasonable. By contrast, segmentation algorithms such as circular binary segmentation (Olshen *et al.* 2004; Venkatraman and Olshen 2007) may be more appropriate for diseases such as cancer. While segmentation algorithms estimate segment means, HMMs provide direct inference about the latent copy number states of interest and can be used to identify copy-neutral regions of homozygosity.

This paper describes software for the first of a two-stage approach for identifying CNV in high-throughput genotyping arrays. Specifically, the implementation of a multi-level model for copy number estimation in the R package **crlmm**. We illustrate our approach on 1258 HapMap samples that were assayed on the Affymetrix 6.0 platform. Section 2 discusses the steps for preprocessing and genotyping the HapMap samples with **crlmm**. Locus-level copy number estimation is described in Section 3. Section 4 illustrates how copy number estimates from **crlmm** can be passed to HMMs or segmentation algorithms that smooth the locus level estimates. Closing remarks are provided in Section 5.

2. Preprocessing and genotyping

This document is written in `Sweave` and is available as part of a compendium from the following website: http://www.biostat.jhsph.edu/~rscharpf/crlmmCompendium/index.html. The compendium contains code, R functions, and data for reproducing the figures in this analysis, and helps to insure that the required packages are available and current. The website places code extracted from this `Sweave` document for each of the figures alongside thumbnail versions of the figures. While reproducing the figures in this paper is possibly by installing the compendium, reproducing the complete analysis described in this `Sweave` file requires two additional steps. First, one would need to obtain the CEL files for the HapMap phase 3 data and verify that any additional R packages beyond those that are required for installing the compendium are available. See Section 6 for the R session information from our analysis. Secondly, the following codechunk specifying the path to the CEL files and the directory to store results should be edited as appropriate.

We begin our analysis of the HapMap data by loading the compendium and enabling large data support (LDS). LDS is enabled in the **crlmm** package simply by loading the R package **ff**. The **ff** is available from CRAN (http://cran.r-project.org/).

We begin our analysis by loading the compendium

```
> library(crlmmCompendium)
> library(ff)
```

With LDS enabled, one can fine-tune the RAM required for the genotyping and copy number estimation. In general, the computational tasks that require all samples do not require all probes and vice versa. In the following code, the functions `ocProbesets` and `ocSamples`

indicate that we will process the data, where possible, in strata of 50,000 probes or 200 samples. By pulling only data for a subset of the probes and/or samples into active memory, we reduce the required RAM for processing large datasets and lessen the dependency on high performance computing clusters with large amounts of free RAM. The drawback to this approach is the increase in I/O, particularly if the data is saved over a network.

```
> ldPath(outdir)
> ocProbesets(50000)
> ocSamples(200)
```

We complete the set-up for our analysis of the HapMap samples by specifying the names of the CEL files and defining a surrogate for batch. A useful surrogate for batch is the scan date of the array or the chemistry plate. For the HapMap phase 3 data, the chemistry plate is the first 5 letters of the CEL filename. We extract the plate names from the filenames in the following code.

```
> filenames <- list.celfiles(pathToCels, full.names = TRUE,
+     pattern = ".CEL")
> batch <- substr(basename(filenames), 1, 5)
```

While the preprocessing and genotyping of Affymetrix CEL files or Illumina IDAT files does not require a minimum number of samples, allele-specific copy number estimation is more difficult for batches with few samples. To sidestep this difficulty, we exclude the plates CHEAP, CORER, and TESLA that each have fewer than 10 samples. Statistical approaches to improve estimation of allele-specific copy number for small batches is a future area of methodological development in **crlmm**.

```
> excludeBatches <- names(table(batch))[table(batch) <
+     10]
> exclude <- batch %in% excludeBatches
> filenames <- filenames[!exclude]
> batch <- as.factor(batch[!exclude])
```

Preprocessing. Preprocessing refers to normalization of the raw fluorescence intensities to remove technological artifacts that may affect the location and scale of the intensities measured from the optical scanners across arrays. Recent platforms for Affymetrix and Illumina include probes for polymorphic loci as well as probes for nonpolymorphic regions. At polymorphic loci, the raw intensities for each allele are quantile normalized to a target reference distribution obtained from the HapMap phase 2 samples (Bolstad *et al.* 2003). The Affymetrix 6.0 platform contains 3 or 4 identical probes for each allele. The normalized intensities for a set of identical probes are summarized by the median. For nonpolymorphic loci, only one probe per loci is available and the intensities are quantile normalized without a subsequent summarization step. Following the normalization and summarization of the intensities at both the polymorphic and nonpolymorphic loci, the polymorphic markers are genotyped by the `crlmm` algorithm. Additional details regarding the preprocessing and genotyping of Affymetrix CEL files and Illumina IDAT files are described elsewhere (Carvalho *et al.* 2007; **?**; Ritchie *et al.* 2009).

The steps for preprocessing and quantile-normalizing Affymetrix CEL files in **crlmm** are wrapped in the function `genotype`. (Users that only want the genotype calls and do not intend to estimate copy number should use the `crlmm` function instead.) The object returned by the `genotype` function is an instance of the S4 class `CNSet` and serves as a container for the normalized intensities and the genotype calls. The class `CNSet` extends the **eSet** class definition in **Biobase** and thereby inherits a lot of the infrastructure for manipulating high-dimensional set forth in the R package **Biobase**. The class extends **eSet** with additional slots for `batch` and `batchStatistics`. These slots are described in greater detail in Section 3. As the preprocessing and genotyping is computationally intensive, the script that includes the following code chunk would typically be submitted using `R CMD batch`.

```
> container <- genotype(filenames = filenames, cdfName = "genomewidesnp6",
+     copynumber = TRUE, batch = batch)
```

By default, the sample names for the `container`, accessible by `sampleNames(container)` are the CEL filenames. The **crlmmCompendium** contains a mapping from the CEL filenames to the more familiar HapMap identifiers. The following code changes the sample labels from the filename to the HapMap identifiers.

```
> container <- useHapMapIds(container)
> sampleNames(container)[1:5]

[1] "NA06989" "NA11891" "NA12058" "NA11843" "NA07037"
```

The metadata on the samples and features can be listed with the `varLabels` and `fvarLabels`, respectively.

```
> assayDataElementNames(container)

[1] "alleleA"         "alleleB"         "call"
[4] "callProbability"

> varLabels(container)

[1] "SKW"      "SNR"      "gender"   "hapmapId" "celFiles"

> fvarLabels(container)

[1] "chromosome" "position"   "isSnp"
```

As a result of our decision to load the **ff** package, the assay data elements in the `cnSet` object contain pointers to potentially very large objects on disk. One can list all of the **ff** files created during the initialization of the container as in the following code chunk. These files should not be moved or relocated.

```
> list.files(ldPath(), pattern = "\\.ff$")[1:3]
```

```
[1] "A_119495cff.ff" "A_22ae8944a.ff" "B_1625558ec.ff"
```

The underlying data structures are intended to be handled seamlessly through the provided interface in **crlmm**. For instance, in the following code chunk we open file connections to the **ff** objects and access the quantile normalized intensities for the first 5 markers and the first 6 samples for allele A.

```
> invisible(open(container))
> system.time(res1 <- A(container)[1:5, 1:6])
```

```
   user  system elapsed
  0.003   0.000   0.004
```

The above query is not instantaneous as these items pull data from large `ff` objects on disk to active memory. Note that issuing the bracket operator, `[,]`, in the above commmand without specifying the rows or columns would pull all of the data from disk to active memory, defeating the purpose of using the **ff** package. Subset operations on the `container` object should be used with care. For instance, note the substantial difference in time for the following command that returns the same result as in the preceding code chunk.

```
> system.time(res2 <- A(container[1:5, 1:6]))
```

```
   user  system elapsed
  0.430   0.028   0.460
```

```
> all.equal(res1, res2)
```

```
[1] TRUE
```

```
> invisible(close(container))
```

In the analysis of genomewide association data, it is often useful to visualize the genotype clusters for loci of interest. All that is required for such a visualization is the platform-specific identifier for the SNP of interest. In the example below, we plot the genotype clusters for SNP_A-4247386. The object `genotypeSet` that contains the data for this SNP is available in the compendium accompanying this manuscript (http://www.biostat.jhsph.edu/~rscharpf/crlmmCompendium/index.html), and was generated from the following commands.

```
> snpid <- "SNP_A-4247386"
> i <- match(snpid, featureNames(container))
> invisible(open(container))
> genotypeSet <- container[i, ]
> invisible(close(container))
```

The following code chunk extracts the normalized intensities, the genotype calls, and the confidence scores for the genotypes. The extracted data is then plotted in two complimentary ways. The left panel in Figure 1 is a scatterplot of the log 2 normalized intensities for each allele shaded by the genotype call. In the right panel, we instead shade the plotting symbols by the genotype confidence score with lower confidence scores corresponding to darker shades of gray.

```
> data(genotypeSet)
> a <- as.matrix(log2(A(genotypeSet)))
> b <- as.matrix(log2(B(genotypeSet)))
> gt <- as.integer(calls(genotypeSet))
> col <- brewer.pal(3, "Set1")[gt]
> gt.conf <- as.numeric(confs(genotypeSet))
> min.conf <- min(gt.conf)
> max.conf <- max(gt.conf)
> sc <- (gt.conf - min.conf)/(max.conf - min.conf)
> bg <- rep(NA, ncol(genotypeSet))
> for (j in seq_along(bg)) bg[j] <- grey(sc[j])
> par(las = 1, mfrow = c(1, 2), mar = c(0.5, 0.2, 0.5,
+     0.2), oma = c(4, 4, 2, 2))
> plot(a, b, bg = col, pch = 21, cex = 0.7, xlab = , ylab = "",
+     cex.axis = 0.8)
> plot(a, b, bg = bg, pch = 21, cex = 0.7, xlab = "", ylab = "",
+     yaxt = "n", cex.axis = 0.8)
> mtext(featureNames(genotypeSet), 3, outer = TRUE)
> mtext(expression(log[2](I[A])), 1, outer = TRUE, line = 2)
> par(las = 3)
> mtext(expression(log[2](I[B])), 2, outer = TRUE, line = 2)
```

3. Locus-level copy number estimation

In large studies, batch effects become evident as the strength of the A and/or B intensities can depend on when the samples were processed and scanned. Algorithms that assign biallelic genotypes to samples based on the ratio of log intensities, as implemented in the **crlmm** algorithm, are more resistant to batch effects as a consequence of robustness of the log ratio to batch differences. However, estimation of allele-specific copy number is more difficult as batch effects and true differences in copy number would be similar in terms of their effects on the measured strength of the allelic intensities. While quantile normalization is an effective means for removing array to array variation and provides additional robustness to outliers in individual samples, such normalization procedures are insufficient for removing batch effects. In this section, we discuss the implementation of the algorithm in **crlmm**, complete our description of the `CNSet` container that was introduced in the preceding section, describe useful accessors for summary statistics at the copy number level, and suggest visualizations that can be used to assess goodness of fit.

Copy number estimation in **crlmm** consists of the following steps. First, we compute robust estimates of the within-genotype location and scale using the median and median absolute

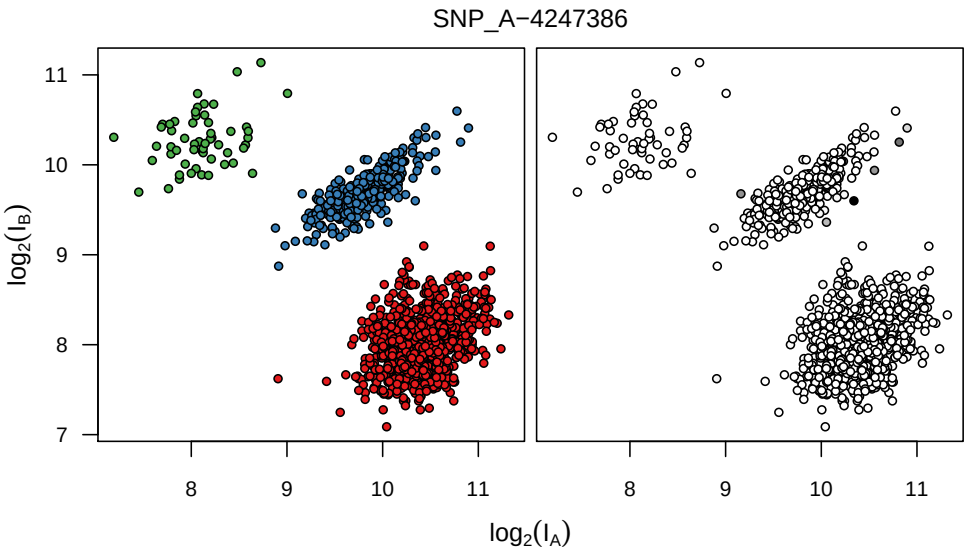

Figure 1: A plot of the genotype clusters for one SNP color coded by the genotype calls (left) and the **crlmm** confidence score (right). While all the confidence scores were high for this SNP, darker shades of grey in the right panel correspond to samples with a relatively lower confidence score.

deviation (MAD), respectively, for each batch of samples. At polymorphic loci, many of the genotypes may be unobserved. For such loci, we impute the unobserved within-genotype medians using regression (see Scharpf *et al.* (2010)). Shrinking the within-genotype variance estimates to the median value across all SNPs provides additional robustness to outliers. For each locus we assume a linear relationship between allelic dosage and the median within-genotype intensity. The intercept and slope coefficients describing the linear relationship are estimated using weighted least squares regression. The above steps are implemented in the R function `crlmmCopynumber`. Using the default settings for this function, one only pass the container returned by the `genotype` function in the previous section.

```
> cnSet <- crlmmCopynumber(container)
```

Batch-specific statistics estimated during the copy number step are stored in the `batchStatistics` slot of the `CNSet` object. Each element in this slot has dimension $R \times C$, where R is the number of markers and C is the number of batches. Batch summary statistics include the within-genotype cluster median and the MAD for each SNP, the correlation of the normalized A and B intensities within each cluster (correlations were computed on the $\log 2$ scale), and the number of AA, AB, and BB genotypes. Accessors in the **crlmm** package return these summary statistics as arrays. If LDS is enabled, each element in the `batchStatistics` slot will be an `ff` object. The following code chunk illustrates a few of the available accessors for batch summary statistics.

```
> Ns(cnSet, i = 1:3, j = 1:2)
> mads(cnSet, i = 1:3, j = 1:2)[, "A", , ]
> medians(cnSet, i = 1:3, j = 1:2)[, "A", , ]
```

The regression coefficients from the model for copy number are also stored in the `batchStatistics` slot. These coefficients are used to compute allele-specific copy number through the following relationship:

$$\hat{c}_{k,ijp} = \max\left\{\frac{1}{\hat{\phi}_{k,ip}}\left(I_{k,ijp} - \hat{\nu}_{k,ip}\right),\ 0\right\} \text{ for } k \in \{A, B\}. \quad (1)$$

The estimates of allele-specific copy number, $\hat{c}_A$ and $\hat{c}_B$, are retrieved from the `cnSet` object using the methods `CA` and `CB`, respectively.

As with genotype calls, a useful means to inspect model fit is to plot the lower level intensities along with statistical summaries for copy number, such as the fit of the regression line. Figure 2 illustrates the fit of the linear model to the A allele intensities for 16 randomly selected polymorphic loci. As the regression is fit independently for each batch, the normalized data plotted in these panels displays only the samples on the GIGAS chemistry plate. Again, the data used for producing Figure 2 is available in the **crlmmCompendium** package, and the code used to generate the data is included in the following code chunk.

```
> invisible(open(cnSet))
> set.seed(123)
> snp.index <- sample(which(isSnp(cnSet) == 1), 16, replace = FALSE)
> sample.index <- which(batch(cnSet) == "GIGAS")
> exampleData1 <- cnSet[snp.index, sample.index]
> invisible(close(cnSet))
```

As the construction of Figure 2 requires accessing several levels of the processed data from the `cnSet` object, we briefly step through the code used to produce this figure. First, we randomly sample the indices of 16 SNPs and define a column index that selects only the samples from a single plate, GIGAS. Next, we extract the normalized intensities for the A and B alleles as well as the genotype calls for the selected markers and SNPs. As the coefficients from the linear model are marker- and batch-specific, we store the coefficients as `ff` objects on disk and maintain pointers to these files in the `cnSet` object. The intercept, ν_A, and slope coefficient, ϕ_A, for the A allele can be extracted with the accessors `nu` and `phi`, respectively.

```
> data(exampleData1)
> a <- as.matrix(A(exampleData1))
> b <- as.matrix(B(exampleData1))
> gt <- as.matrix(calls(exampleData1))
> nuA <- nu(exampleData1, "A")
> phA <- phi(exampleData1, "A")
> col <- brewer.pal(7, "Accent")[c(1, 4, 7)]
```

Looping through the marker indices, we construct boxplots of the normalized intensities for the A allele stratified by the genotype calls. The R function `segments` overlays the fitted regression line. A similar strategy could be used to plot the regression line for the B allele.

```
> par(las = 1, mfrow = c(4, 4), mar = rep(0.5, 4), oma = c(4,
+     4, 4, 4))
> for (i in 1:16) {
+     IA <- split(a[i, ], gt[i, ])
+     names(IA)[names(IA) == "1"] <- "AA"
+     names(IA)[names(IA) == "2"] <- "AB"
+     names(IA)[names(IA) == "3"] <- "BB"
+     ugt <- sort(unique(gt[i, ]))
+     at <- rep(NA, length(ugt))
+     at[ugt == 1] <- 3
+     at[ugt == 3] <- 1
+     at[ugt == 2] <- 2
+     ylim <- c(nuA[i] - 1000, nuA[i] + 2 * phA[i] + 1000)
+     ylim[1] <- max(0, ylim[1])
+     boxplot(IA, col = col[ugt], xlim = c(0.5, 3.5), at = at,
+         xlim = c(0.5, 3.5), ylim = ylim, xaxt = "n",
+         yaxt = "n")
+     graphics:::segments(y0 = nuA[i], x0 = 1, y1 = nuA[i] +
+         2 * phA[i], x1 = 3, lwd = 2, col = "royalblue")
+     if (i >= 13)
+         axis(1, at = 1:3, labels = c("BB", "AB", "AA"))
+ }
> mtext(expression(I[A]), side = 2, outer = TRUE, line = 1)
```

Scatterplots of the log-transformed normalized intensities for the A and B alleles can be useful for visualizing the prediction regions for integer copy number. Using the same set of randomly selected SNPs in the previous codechunk, we plot the prediction regions for copy numbers 1, 2, and 3 in Figure 3.

```
> lA <- log2(a)
> lB <- log2(b)
> cols <- c("blue", "black", "red")
> par(las = 1, mfrow = c(4, 4), mar = rep(0.5, 4), oma = c(4,
+     4, 4, 4))
> for (i in 1:16) {
+     plot(lB[i, ], lA[i, ], col = "grey50", bg = col[gt[i,
+         ]], xaxt = "n", yaxt = "n", pch = 21, cex = 0.8,
+         xlim = c(6.5, 12.5), ylim = c(6.5, 12.5), xlab = "",
+         ylab = "")
+     for (CN in 1:3) lines(exampleData1, i, "GIGAS", CN,
+         col = cols[CN], lwd = 2, x.axis = "B")
+ }
> mtext(expression(log[2](I[B])), 1, outer = TRUE)
> par(las = 3)
> mtext(expression(log[2](I[A])), 2, outer = TRUE)
```

4. Downstream tools

Marker-level estimates of copy number for Affymetrix and Illumina platforms are too noisy to reliably quantitate copy number at a single marker. Approaches that smooth the copy number estimates as a function of the physical position are useful for inferring regions of copy alterations and copy-neutral regions of homozygosity (ROH). This section illustrates how the marker-level estimates of copy number from **crlmm** can be passed to downstream segmentation and HMM algorithms. We illustrate our approach on chromosome 8 of HapMap sample NA19007 for which a large amplification on the p-arm has been previously identified (Redon *et al.* 2006). The normalized intensities for this sample, the genotype calls, and the parameter estimates for copy number are stored in the `redonSet` object available in the **crlmmCompendium** package. The following codechunk was used to generate this object.

```
> marker.index <- which(chromosome(cnSet) == 8)
> invisible(open(cnSet))
> redonSet <- as(cnSet[marker.index, cnSet$hapmapId ==
+     "NA19007"], "CopyNumberSet")
> invisible(close(cnSet))
> redonSet <- redonSet[order(position(redonSet)), ]
> redonSet <- redonSet[-which(is.na(copyNumber(redonSet))),
+     ]
> redonSet <- redonSet[-which(duplicated(position(redonSet))),
+     ]
```

A hidden Markov model. The HMM implemented in the R package **VanillaICE** allows some flexibility for the data inputs and the definition of the hidden states. Using the default settings for the **VanillaICE** version indicated in Section 6, we specify homozygous deletion, hemizygous deletion, normal, and amplification as the hidden states of interest. In the following codechunk, we record the time required to fit the HMM to the 96,876 markers on chromosome 8 and display the output from the `hmm` function. The HMM finds strong evidence for an amplification as indicated by the log likelihood ratio (LLR) comparing the predicted amplification to the null model of no copy number alteration.

```
> data(redonSet)
> hmmOpts <- hmm.setup(redonSet, c("hom-del", "hem-del",
+     "normal", "amp"), copynumberStates = 0:3, normalIndex = 3,
+     log.initialP = rep(log(1/4), 4))
> timing <- system.time(fit.cn <- hmm(redonSet, hmmOpts,
+     verbose = FALSE))
> hmm.df <- as.data.frame(fit.cn)
> print(hmm.df[, c(2:4, 7:9)])
```

```
      start     end   width state numMarkers          LLR
1     21242 1347537 1326296     3        892    0.0000000
2   1347717 1348097     381     2          9    9.8372915
3   1348129 3672962 2324834     3       2476    0.0000000
4   3674352 4126939  452588     4        769 2127.4739235
```

```
5    4127094   4129387     2294     3          6    0.0000000
6    4130651   5566202  1435552     4       2014 5541.0029926
7    5568079   5571263     3185     3          7    0.0000000
8    5575331   5938114   362784     4        486 1049.2834539
9    5938253  25032441 19094189     3      15516    0.0000000
10  25032546  25035892     3347     4         13    2.7949839
11  25039974  43943193 18903220     3      11802    0.0000000
12  46966687 129825448 82858762     3      51522    0.0000000
13 129832305 129833255      951     1          2    2.9233360
14 129846624 135127490  5280867     3       4063    0.0000000
15 135130425 135135878     5454     1         16  123.6816954
16 135144379 137962640  2818262     3       2244    0.0000000
17 137963670 137963684       15     1          2    0.2727200
18 137969915 146268947  8299033     3       5037    0.0000000
```

```
> print(timing)

   user  system elapsed
  1.015   0.001   1.019
```

Circular binary segmentation. CBS is implemented in the R package **DNAcopy** and is particularly useful for cancer data in which noninteger copy numbers are plausible. Again, we adopt the default settings for this algorithm in the following codechunk.

```
> library(DNAcopy)
> CNA.object <- CNA(genomdat = copyNumber(redonSet), chrom = chromosome(redonSet),
+     maploc = position(redonSet), data.type = "logratio",
+     sampleid = sampleNames(redonSet))
> smu.object <- smooth.CNA(CNA.object)
> timing.cbs <- system.time(cbs.segments <- segment(smu.object))

Analyzing: NA19007

> print(cbs.segments, showSegRows = TRUE)

segmented logratio CNA data with 1 samples and 96876 probes
segment(x = smu.object)
                 ID chrom loc.start loc.end num.mark seg.mean
CN_1290009    NA19007     8     21242  320865      157   2.0772
SNP_A-8457345 NA19007     8    321550  321630        2   0.0812
CN_1279103    NA19007     8    324018  585208      184   2.0283
CN_371097     NA19007     8    585310  585896        9   3.4527
SNP_A-8602711 NA19007     8    591043 1347537      540   2.0024
CN_1291837    NA19007     8   1347717 1348097        9   1.0361
              startRow endRow
CN_1290009           1    157
```

```
SNP_A-8457345       158    159
CN_1279103          160    343
CN_371097           344    352
SNP_A-8602711       353    892
CN_1291837          893    901
```

```
> timing.cbs
```

```
   user  system elapsed
 20.236   0.001  20.240
```

As CBS does not *call* deletions and amplifications, we implemented a few simple rules to indicate whether a region is likely to be amplified or deleted. We then coerce the output to an instance of the `IRanges` class `RangedData` to facilitate plotting the intervals and comparisons with the HMM results.

```
> cbs.segments <- cbs.segments$output
> cbs.segments$call <- rep(3, nrow(cbs.segments))
> cbs.segments$call[cbs.segments$seg.mean > 2.5] <- 4
> cbs.segments$call[cbs.segments$seg.mean < 1.25 & cbs.segments$seg.mean >
+     0.75] <- 2
> cbs.segments$call[cbs.segments$seg.mean < 0.75] <- 1
> cbs.ir <- RangedData(IRanges(cbs.segments$loc.start,
+     cbs.segments$loc.end), chrom = cbs.segments$chrom,
+     numMarkers = cbs.segments$num.mark, seg.mean = cbs.segments$seg.mean,
+     cnCall = cbs.segments$call)
```

We plot the predicted states from the HMM and the CBS algorithm beneath the marker-level copy number estimates from **crlmm** in Figure 4. The code for producing this Figure in included in the website describing the compendium.

5. Discussion

We have applied the **crlmm** software to the HapMap phase 3 data, illustrating the steps of preprocessing, the genotyping of polymorphic markers, and the estimation of allele-specific copy number. We organize the normalized intensities, statistical summaries from the genotyping and copy number estimation steps, and meta-data on the features and samples in a single container. This container extends the `eSet` class defined in **Biobase**, with additional slots to accommodate batch-specific statistical summaries relevant for copy number analyses. This organization facilitates visualizations that allow inspection of the genotypes and copy number estimates in the context of the lower-level data. We have provided such visualizations in the course of the estimation steps for copy number using the HapMap data as an exemplar. However, note that it would be straightforward to proceed in the opposite direction – to target specific genomic regions in which copy number estimates are associated with a particular phenotype, followed by more detailed inspection of the loci in this region. While smoothing the locus-level estimates of copy number to infer regions of gain and loss is beyond

the scope of the **crlmm** package, we have illustrated how **crlmm** can be extended by hidden Markov models implemented in the **VanillaICE** package or the circular binary segmentation algorithm implemented in the **DNAcopy** package. Batch effects are common in large high-throughput laboratores. The **crlmm** package models the variation driven by batch as part of the estimation procedure for copy number, permitting inference of copy number gain and loss from batch-adjusted locus-level summaries. We expect that such an approach will reduce the occurrence of spurious associations induced by temporal artifacts such as batch effects.

6. Session information

The R package **crlmm** is available from Bioconductor http://www.bioconductor.org.

This document was prepared using Sweave. Computationally intensive steps, such as the genotype callling and copy number estimation, were precomputed and efficient summaries loaded from files.

- R version 2.12.0 alpha (2010-09-23 r52986), `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, tools, utils
- Other packages: Biobase 2.9.1, bit 1.1-4, crlmm 1.7.14, crlmmCompendium 1.0.3, DNAcopy 1.23.6, ellipse 0.3-5, ff 2.1-4, genefilter 1.31.2, IRanges 1.7.34, MASS 7.3-8, oligoClasses 1.11.8, RColorBrewer 1.0-2, SNPchip 1.13.0, VanillaICE 1.11.3
- Loaded via a namespace (and not attached): affyio 1.17.4, annotate 1.27.1, AnnotationDbi 1.11.6, Biostrings 2.17.47, DBI 0.2-5, mvtnorm 0.9-92, preprocessCore 1.11.0, RSQLite 0.9-2, splines 2.12.0, survival 2.35-8, xtable 1.5-6

Acknowledgements

RBS was supported by grant 4R00HG005015 from the NIH/NHGRI and Johns Hopkins Medical Institutions' CTSA grant. IR was supported by NIH grant R01GM083084. We would like to think Marvin Newhouse for computing support.

References

Altug-Teber O, Dufke A, Poths S, Mau-Holzmann UA, Bastepe M, Colleaux L, Cormier-Daire V, Eggermann T, Gillessen-Kaesbach G, Bonin M, Riess O (2005). "A rapid microarray based whole genome analysis for detection of uniparental disomy." *Hum Mutat*, **26**(2), 153–9. ISSN 1098-1004 (Electronic).

Bolstad BM, Irizarry RA, Astrand M, Speed TP (2003). "A comparison of normalization methods for high density oligonucleotide array data based on variance and bias." *Bioinformatics (Oxford, England)*, **19**(2), 185–193. PUBM: Print; JID: 9808944; 0 (Molecular Probes); ppublish.

Carvalho B, Bengtsson H, Speed TP, Irizarry RA (2007). "Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data." *Biostatistics*, **8**(2), 485–499. doi:10.1093/biostatistics/kxl042. URL http://dx.doi.org/10.1093/biostatistics/kxl042.

Colella S, Yau C, Taylor JM, Mirza G, Butler H, Clouston P, Bassett AS, Seller A, Holmes CC, Ragoussis J (2007). "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. doi:10.1093/nar/gkm076. URL http://dx.doi.org/10.1093/nar/gkm076.

Fridlyand J, Snijders A, Pinkel D, Albertson D, Jain A (2004). "Hidden Markov models approach to the analysis of array CGH data." *Journal of Multivariate Analysis*, **90**, 132–153.

Karayiorgou M, Simon TJ, Gogos JA (2010). "22q11.2 microdeletions: linking DNA structural variation to brain dysfunction and schizophrenia." *Nat Rev Neurosci*, **11**(6), 402–416. doi:10.1038/nrn2841. URL http://dx.doi.org/10.1038/nrn2841.

Olshen AB, Venkatraman ES, Lucito R, Wigler M (2004). "Circular binary segmentation for the analysis of array-based DNA copy number data." *Biostatistics*, **5**(4), 557–72. doi:10.1093/biostatistics/kxh008. URL http://dx.doi.org/10.1093/biostatistics/kxh008.

Pinto D, Pagnamenta AT, Klei L, Anney R, Merico D, Regan R, Conroy J, Magalhaes TR, Correia C, Abrahams BS, Almeida J, Bacchelli E, Bader GD, Bailey AJ, Baird G, Battaglia A, Berney T, Bolshakova N, BÃűlte S, Bolton PF, Bourgeron T, Brennan S, Brian J, Bryson SE, Carson AR, Casallo G, Casey J, Chung BHY, Cochrane L, Corsello C, Crawford EL, Crossett A, Cytrynbaum C, Dawson G, de Jonge M, Delorme R, Drmic I, Duketis E, Duque F, Estes A, Farrar P, Fernandez BA, Folstein SE, Fombonne E, Freitag CM, Gilbert J, Gillberg C, Glessner JT, Goldberg J, Green A, Green J, Guter SJ, Hakonarson H, Heron EA, Hill M, Holt R, Howe JL, Hughes G, Hus V, Igliozzi R, Kim C, Klauck SM, Kolevzon A, Korvatska O, Kustanovich V, Lajonchere CM, Lamb JA, Laskawiec M, Leboyer M, Couteur AL, Leventhal BL, Lionel AC, Liu XQ, Lord C, Lotspeich L, Lund SC, Maestrini E, Mahoney W, Mantoulan C, Marshall CR, McConachie H, McDougle CJ, McGrath J, McMahon WM, Merikangas A, Migita O, Minshew NJ, Mirza GK, Munson J, Nelson SF, Noakes C, Noor A, Nygren G, Oliveira G, Papanikolaou K, Parr JR, Parrini B, Paton T, Pickles A, Pilorge M, Piven J, Ponting CP, Posey DJ, Poustka A, Poustka F, Prasad A, Ragoussis J, Renshaw K, Rickaby J, Roberts W, Roeder K, Roge B, Rutter ML, Bierut LJ, Rice JP, Salt J, Sansom K, Sato D, Segurado R, Sequeira AF, Senman L, Shah N, Sheffield VC, Soorya L, Sousa I, Stein O, Sykes N, Stoppioni V, Strawbridge C, Tancredi R, Tansey K, Thiruvahindrapduram B, Thompson AP, Thomson S, Tryfon A, Tsiantis J, Engeland HV, Vincent JB, Volkmar F, Wallace S, Wang K, Wang Z, Wassink TH, Webber C, Weksberg R, Wing K, Wittemeyer K, Wood S, Wu J, Yaspan BL, Zurawiecki D, Zwaigenbaum L,

Buxbaum JD, Cantor RM, Cook EH, Coon H, Cuccaro ML, Devlin B, Ennis S, Gallagher L, Geschwind DH, Gill M, Haines JL, Hallmayer J, Miller J, Monaco AP, Jr JIN, Paterson AD, Pericak-Vance MA, Schellenberg GD, Szatmari P, Vicente AM, Vieland VJ, Wijsman EM, Scherer SW, Sutcliffe JS, Betancur C (2010). "Functional impact of global rare copy number variation in autism spectrum disorders." *Nature*. doi:10.1038/nature09146. URL http://dx.doi.org/10.1038/nature09146.

Redon R, Ishikawa S, Fitch KR, Feuk L, Perry GH, Andrews TD, Fiegler H, Shapero MH, Carson AR, Chen W, Cho EK, Dallaire S, Freeman JL, Gonzalez JR, Gratacos M, Huang J, Kalaitzopoulos D, Komura D, MacDonald JR, Marshall CR, Mei R, Montgomery L, Nishimura K, Okamura K, Shen F, Somerville MJ, Tchinda J, Valsesia A, Woodwark C, Yang F, Zhang J, Zerjal T, Zhang J, Armengol L, Conrad DF, Estivill X, Tyler-Smith C, Carter NP, Aburatani H, Lee C, Jones KW, Scherer SW, Hurles ME (2006). "Global variation in copy number in the human genome." *Nature*, **444**(7118), 444–454. ISSN 1476-4687 (Electronic). doi:10.1038/nature05329.

Ritchie ME, Carvalho BS, Hetrick KN, TavarÃl' S, Irizarry RA (2009). "R/Bioconductor software for Illumina's Infinium whole-genome genotyping BeadChips." *Bioinformatics*, **25**(19), 2621–2623. doi:10.1093/bioinformatics/btp470. URL http://dx.doi.org/10.1093/bioinformatics/btp470.

Scharpf RB, Parmigiani G, Pevsner J, Ruczinski I (2008). "Hidden Markov models for the assessment of chromosomal alterations using high-throughput SNP arrays." *Annals of Applied Statistics*, **2**(2), 687–713. URL http://dx.doi.org/10.1214/07-AOAS155.

Scharpf RB, Ruczinski I, Carvalho B, Doan B, Chakravarti A, Irizarry R (2010). "A multilevel model to address batch effects in copy number estimation using SNP arrays." *Biostatistics*. doi:10.1093/biostatistics/kxq043.

Venkatraman ES, Olshen AB (2007). "A faster circular binary segmentation algorithm for the analysis of array CGH data." *Bioinformatics*, **23**(6), 657–663. doi:10.1093/bioinformatics/btl646. URL http://dx.doi.org/10.1093/bioinformatics/btl646.

Wang K, Li M, Hadley D, Liu R, Glessner J, Grant SFA, Hakonarson H, Bucan M (2007). "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. doi:10.1101/gr.6861907. URL http://dx.doi.org/10.1101/gr.6861907.

Zhang Q, Ding L, Larson DE, Koboldt DC, McLellan MD, Chen K, Shi X, Kraja A, Mardis ER, Wilson RK, Boreki IB, Province MA (2009). "CMDS: a population-based method for identifying recurrent DNA copy number aberrations in cancer from high-resolution data." *Bioinformatics*. doi:10.1093/bioinformatics/btp708. URL http://dx.doi.org/10.1093/bioinformatics/btp708.

Affiliation:

Robert Scharpf
Department of Oncology,
Johns Hopkins University School of Medicine,
550 N. Broadway, Suite 1103
Baltimore, MD 21218
E-mail: rscharp1@jhmi.edu

Rafael Irizarry and Ingo Ruczinski
Department of Biostatistics Johns Hopkins Bloomberg School of Public Health
615 North Wolfe Street
Baltimore MD 21218
E-mail:rafa@jhu.edu and ingo@jhu.edu

Matthew Ritchie
Bioinformatics Division
The Walter and Eliza Hall Institute of Medical Research
1G Royal Parade
Parkville, Victoria 3052
Australia E-mail: Benilton.Carvalho@cancer.org.uk

Benilton Carvalho
Department of Oncology
University of Cambridge
CRUK Cambridge Research Institute
Li Ka Shing Centre
Robinson Way
Cambridge CB2 ORE
United Kingdom E-mail: mritchie@wehi.EDU.AU

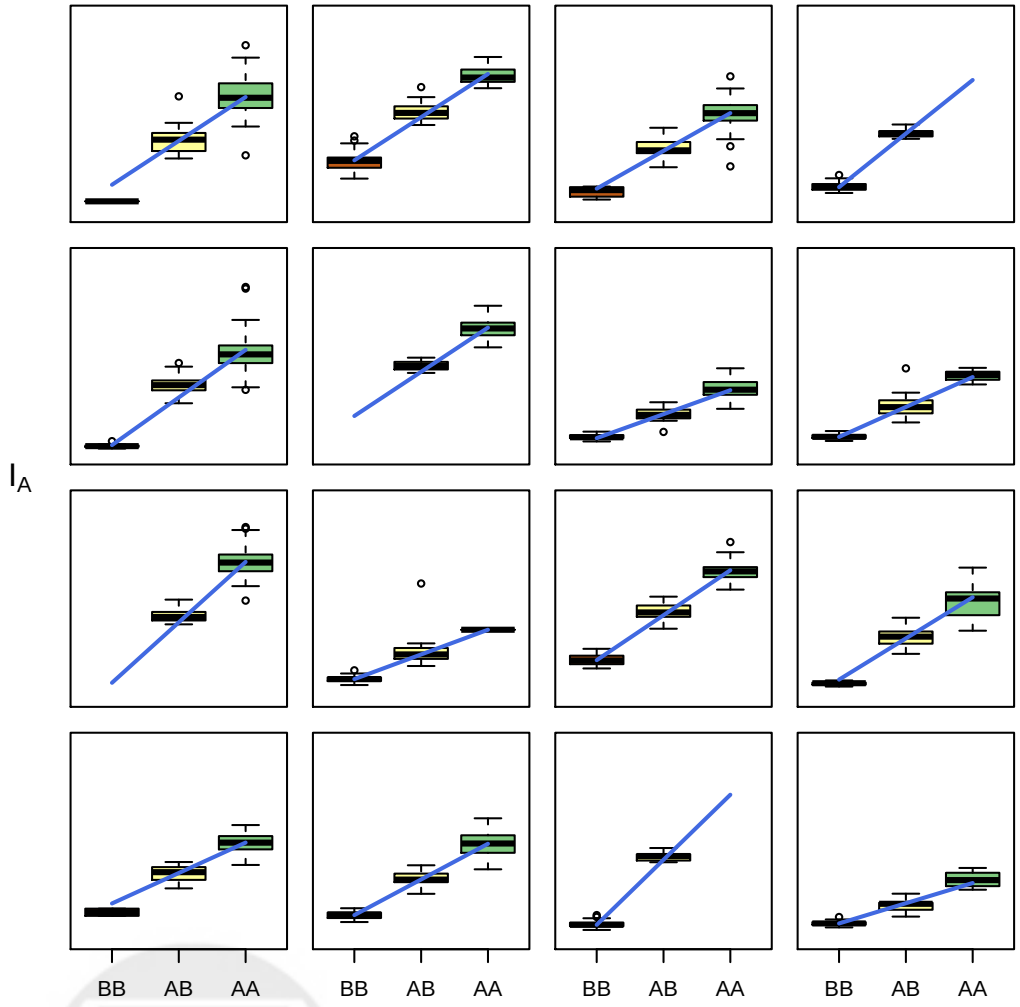

Figure 2: Each panel displays the intensities for the A allele for all samples on the GIGAS plate stratified by the genotype call. The linear model is fitted on the intensity scale (as opposed to the log-scale) with parameters for the intercept and slope that are SNP- and batch-specific. The straight line over-plotted is the estimated background and slope for the GIGAS plate.

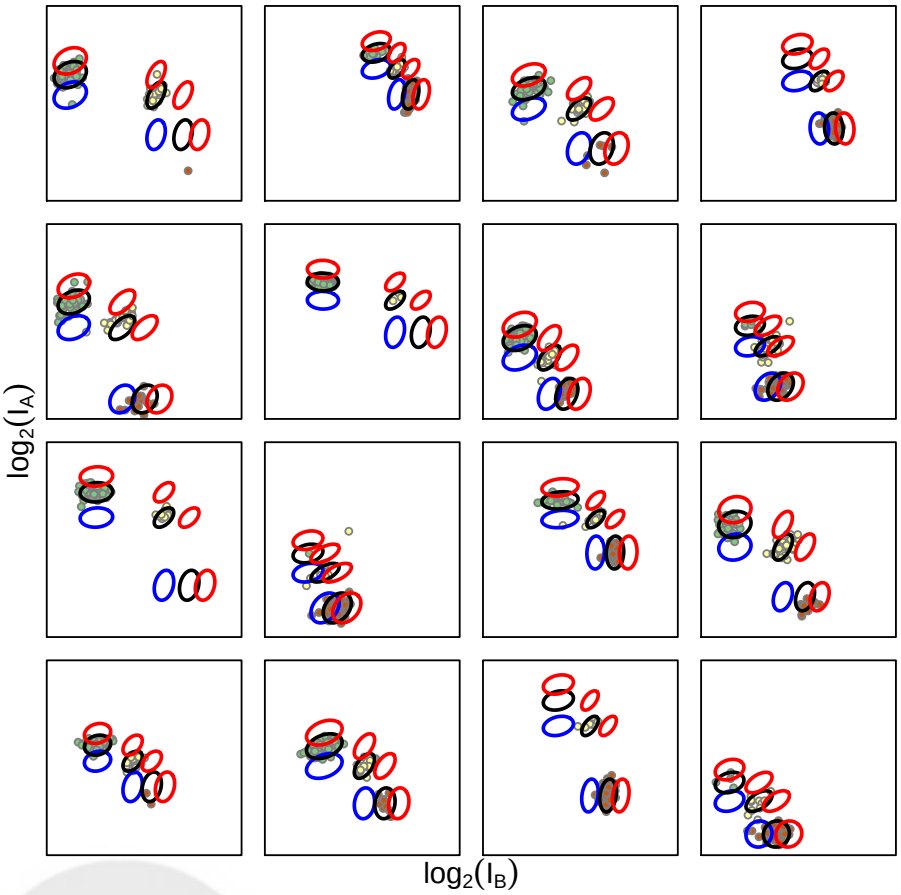

Figure 3: A scatter plot of the log 2 normalized intensities for the 16 SNPs plotted in Figure 2. The colored ellipses are the prediction regions for hemizygous deletion (blue), normal copy number (black), and 3 copies (red).

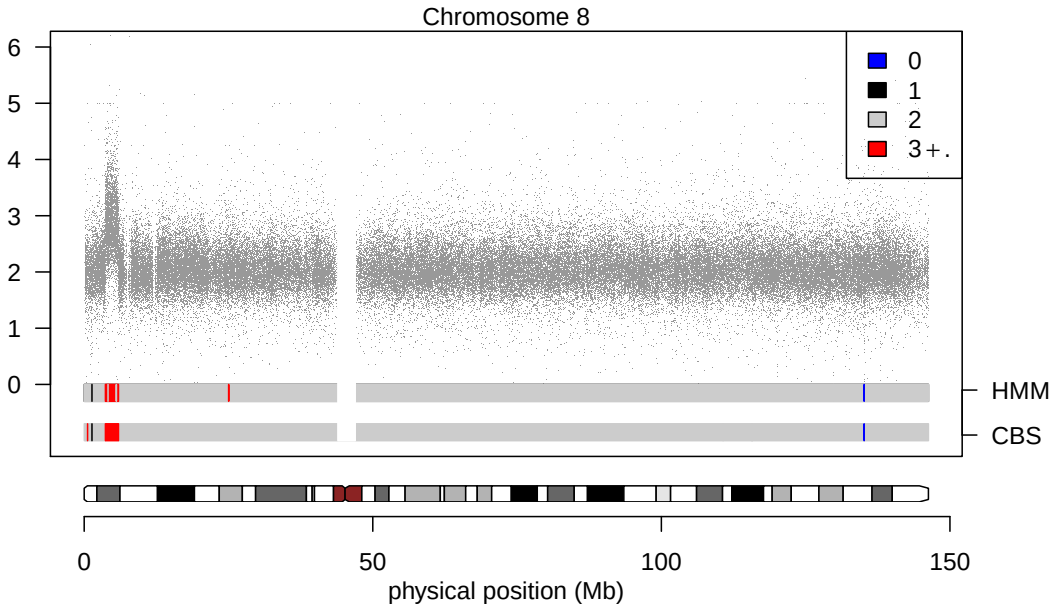

Figure 4: An amplification on the p-arm of chromosome 8 for HapMap sample NA19007. Inferred regions of copy number gain and loss are plotted for a HMM and circular binary segmentation.