# Estimating copy number for Affymetrix 6.0 with the crlmm Package

Rob Scharpf

February, 2009

## 1 Estimating copy number

At present, software for copy number estimation is provided only for the Affymetrix 6.0 platform. This vignette estimates copy number for the HapMap samples.

### 1.1 Preprocess and genotype the samples

See the crlmm vignette for additional details on preprocessing/genotyping.

```
R> library(crlmm)
R> library(genomewidesnp6Crlmm)
R> library(ellipse)
```

You must specify the names of the CEL files and the full path, as well as a directory in which to store the output from crlmm and cnrma.

```
R> celFiles <- list.celfiles("/thumper/ctsa/snpmicroarray/hapmap/raw/affy/1m",
     full.names = TRUE, pattern = ".CEL")
R> outdir <- "/thumper/ctsa/snpmicroarray/rs/data/hapmap/1m/affy"
```

Preprocess and genotype (for more info see the crlmm vignette).

```
R> if (!exists("crlmmResult")) {
     if (file.exists(file.path(outdir, "crlmmResult.rda"))) {
         load(file.path(outdir, "intensities.rda"))
         load(file.path(outdir, "crlmmResult.rda"))
     }
     else {
         crlmmResult <- crlmm(celFiles, save.it = TRUE,
             intensityFile = file.path(outdir,
                 "intensities.rda"))
         save(crlmmResult, file = file.path(outdir,
             "crlmmResult.rda"))
     }
 }
```

Quantile normalize the nonpolymorphic probes and save the results.

```
R> if (!exists("cnrmaResult")) {
     if (file.exists(file.path(outdir, "cnrmaResult.rda")))
         load(file.path(outdir, "cnrmaResult.rda"))
     else {
         cnrmaResult <- cnrma(celFiles)
         save(cnrmaResult, file = file.path(outdir,
             "cnrmaResult.rda"))
     }
 }
```

## 1.2 Copy number

Copy number can be assessed one chromosome at a time. Here we specify chromosome 15 and and load a list of indices to subset the data. The first element in the list correspond to indices of polymorphic probes on chromosome 15; the second element corresponds to indices of nonpolymorphic probes on chromosome 15.

```
R> CHR <- 21
R> CHR_INDEX <- paste(CHR, "index", sep = "")
R> data(list = CHR_INDEX, package = "genomewidesnp6Crlmm")
R> str(index)
```

```
List of 2
 $ snps: int [1:12579] 31618 31619 31620 31621 31622 31623 31624 31625 31626 31627 ...
 $ nps : int [1:12181] 863391 863389 863390 863395 863392 863393 863394 863399 863396 863397
```

Next we load 3 files that were saved from the preprocessing step and then subset these lists using the above indices to extract the preprocessed intensities and genotypes needed for estimating copy number. Specifically, we require 6 items:

- quantile-normalized A intensities (I1 x J)
- quantile-normalized B intensities (I1 x J)
- quantile-normalized intensities from nonpolymorphic (NP) probes (I2 x J)
- genotype calls (I1 x J)
- confidence scores of the genotype calls (I1 x J)
- signal to noise ratio (SNR) of the samples (J)

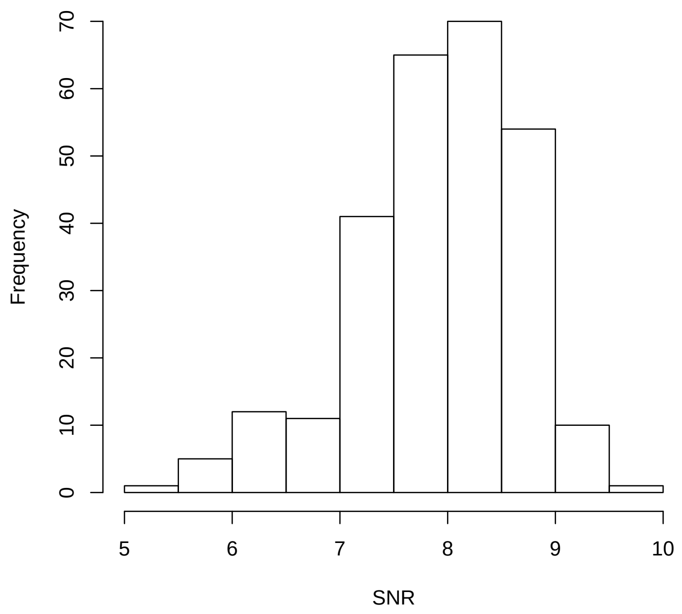
These items are extracted as follows:

```
R> A <- res$A
R> B <- res$B
R> calls <- crlmmResult$calls
R> conf <- crlmmResult$conf
R> SNR <- crlmmResult$SNR
R> NP <- cnrmaResult$NP
R> gc()
```

```
            used   (Mb) gc trigger   (Mb)  max used   (Mb)
Ncells   3280331  175.2    5041497  269.3   5041497  269.3
Vcells 647604057 4940.9  968762575 7391.1 968698363 7390.6
```

Make a histogram of the signal to noise ratio for these samples:

```
R> hist(SNR, xlab = "SNR", main = "")
```



We suggest excluding samples with a signal to noise ratio less than 5. We then extract the probes (rows) that are on chromosome 15 and the samples (columns) that have a suitable signal to noise ratio.

```
R> A <- A[index[[1]], SNR > 5]
R> B <- B[index[[1]], SNR > 5]
R> calls <- calls[index[[1]], SNR > 5]
R> conf <- conf[index[[1]], SNR > 5]
R> NP <- NP[index[[2]], SNR > 5]
```

Batch effects can be very large in the quantile-normalized intensities. Often the chemistry plate is a good surrogate for batch effects, although it can also be the lab, etc. Here we define batch by the chemistry plate. For the HapMap data, plate is often confounded with ancestry.

```
R> sns <- colnames(calls)
R> sns[1]
```

```
[1] "NA06985_GW6_C.CEL"
```

```
R> plate <- substr(basename(sns), 13, 13)
R> table(plate)
```

```
plate
 A  C  Y
90 90 90
```

We are now ready to estimate copy number for each batch. In the current version of this package, one specifies an environment to which intermediate R objects for copy number estimation are stored. Allele-specific estimates of copy number are also stored in this environment.

```
R> if (!exists("e1")) {
     e1 <- new.env()
     computeCopynumber(A = A, B = B, calls = calls,
         conf = conf, NP = NP, plate = plate, envir = e1,
         chrom = CHR, DF.PRIOR = 75)
 }
```

The DF.PRIOR indicates how much we will shrink SNP-specific estimates of the variance and correlation.

```
R> copyA <- get("CA", e1)
R> copyB <- get("CB", e1)
R> copyT <- (copyA + copyB)/100
R> copyT[copyT < 0] <- 0
R> copyT[copyT > 6] <- 6
```

## 2 Suggested plots

### One sample at a time

```
R> data(snpProbes, package = "genomewidesnp6Crlmm")
R> data(cnProbes, package = "genomewidesnp6Crlmm")
R> position <- snpProbes[match(rownames(calls), rownames(snpProbes)),
     "position"]
```

To plot physical position versus copy number for the first sample:

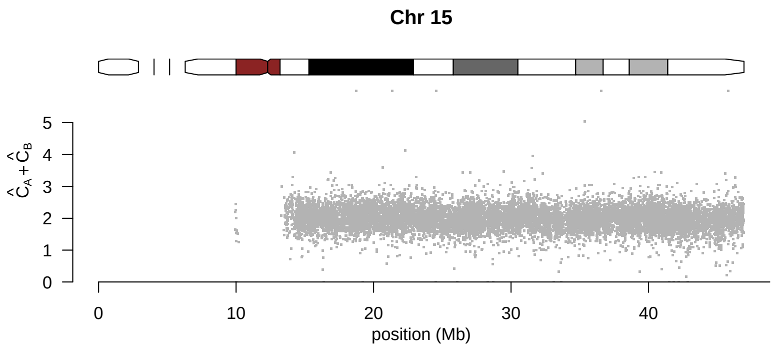


Figure 1: Total copy number (y-axis) for chromosome 22 plotted against physical position (x-axis) for one sample.

```
R> require(SNPchip)
R> par(las = 1)
R> plotCytoband(as.character(CHR), ylim = c(0, 7),
     cytoband.ycoords = c(6.5, 7), label.cytoband = FALSE,
     main = "Chr 15")

NULL

R> points(position, copyT[, 1], pch = ".", cex = 2,
     , xaxt = "n", col = "grey70")
R> axis(1, at = pretty(range(position)), labels = pretty(range(position))/1e+06)
R> axis(2, at = 0:5, labels = 0:5)
R> mtext("position (Mb)", 1, line = 2)
R> mtext(expression(hat(C)[A] + hat(C)[B]), 2, line = 2,
     las = 3)
```

**One SNP at a time** This section needs to be cleaned up (TODO).

```
R> tau2A <- get("tau2A", e1)
R> tau2B <- get("tau2B", e1)
R> sig2A <- get("sig2A", e1)
R> sig2B <- get("sig2B", e1)
R> nuA <- get("nuA", e1)
R> phiA <- get("phiA", e1)
R> nuB <- get("nuB", e1)
R> phiB <- get("phiB", e1)
R> corr <- get("corr", e1)
R> corrA.BB <- get("corrA.BB", e1)
```

```
R> corrB.AA <- get("corrA.BB", e1)
R> A <- get("A", e1)
R> B <- get("B", e1)
```

Here, we plot the prediction regions for total copy number 2 and 3 for the first plate. Plotting symbols are the genotype calls (1=AA, 2=AB, 3=BB); light grey points are from other plates. (You could also add the prediction regions for 0-4 copies, but it gets crowded). Notice that there is little evidence of a plate effect for this SNP.

```
R> par(las = 1, pty = "s")
R> p <- 1
R> J <- grep(unique(plate)[p], plate)
R> ylim <- c(6.5, 13)
R> I <- which(phiA > 10 & phiB > 10)
R> i <- I[1]
R> log2(phiA[i, ])

[1] 8.962896 8.924813 9.546894

R> log2(phiB[i, ])

[1] 7.954196 7.954196 9.501837

R> plot(log2(A[i, ]), log2(B[i, ]), pch = as.character(calls[i,
     ]), col = "grey60", cex = 0.9, ylim = ylim,
     xlim = ylim, xlab = "A", ylab = "B")
R> points(log2(A[i, J]), log2(B[i, J]), col = "black",
     pch = as.character(calls[i, J]))
R> for (CT in 2) {
     if (CT == 2)
         ellipse.col <- "black"
     for (CA in 0:CT) {
         CB <- CT - CA
         A.scale <- sqrt(tau2A[i, p] * (CA == 0) +
             sig2A[i, p] * (CA > 0))
         B.scale <- sqrt(tau2B[i, p] * (CB == 0) +
             sig2B[i, p] * (CB > 0))
         scale <- c(A.scale, B.scale)
         if (CA == 0 & CB > 0)
             rho <- corrA.BB[i, p]
         if (CA > 0 & CB == 0)
             rho <- corrB.AA[i, p]
         if (CA > 0 & CB > 0)
             rho <- corr[i, p]
         lines(ellipse(x = rho, centre = c(log2(nuA[i,
             p] + CA * phiA[i, p]), log2(nuB[i,
```

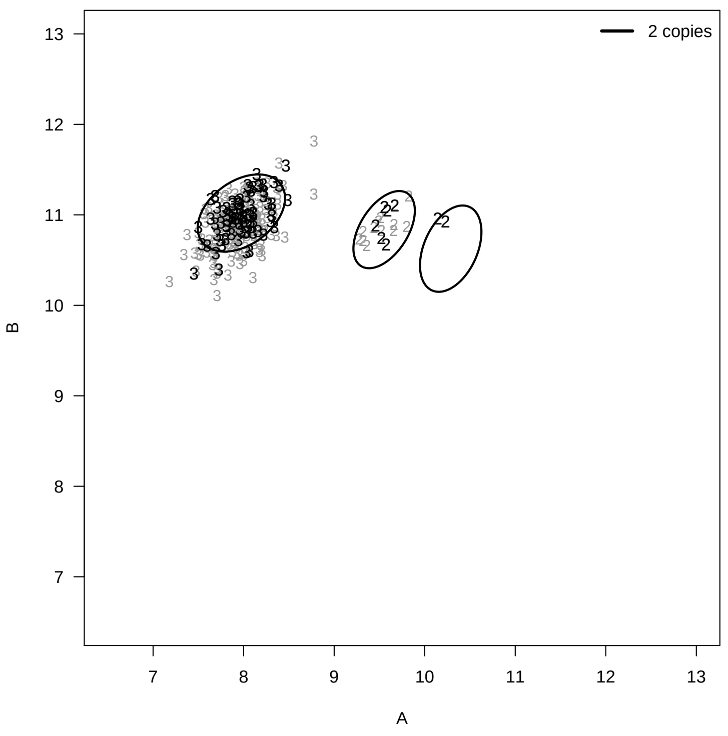


Figure 2: Prediction regions for copy number 2 for one SNP. The plate effects are negligible for this SNP and we only plot the prediction region for the first plate.

```
             p] + CB * phiB[i, p])), scale = scale),
             col = ellipse.col, lwd = 2)
     }
 }
R> legend("topright", lwd = 3, col = "black", legend = "2 copies",
     bty = "n")
```

```
R> par(las = 1, pty = "s", ask = TRUE)
R> p <- 1
R> J <- grep(unique(plate)[p], plate)
R> ylim <- c(6.5, 13)
R> I <- which(phiA > 10 & phiB > 10)
R> col <- c("grey0", "grey30", "grey70")
R> cex <- 3
```

```
R> for (j in seq(along = I)) {
     i <- I[j]
     J1 <- grep("C", plate)
     J2 <- grep("Y", plate)
     J3 <- grep("A", plate)
     plot(log2(A[i, ]), log2(B[i, ]), pch = 21,
         col = "grey60", type = "n", cex = 0.9,
         ylim = ylim, xlim = ylim, xlab = "A",
         ylab = "B")
     points(log2(A[i, J1]), log2(B[i, J1]), col = col[1],
         pch = ".", cex = cex)
     points(log2(A[i, J2]), log2(B[i, J2]), col = col[2],
         pch = ".", cex = cex)
     points(log2(A[i, J3]), log2(B[i, J3]), col = col[3],
         pch = ".", cex = cex)
     P <- 1:3
     ellipse.col <- col
     for (p in seq(along = P)) {
         for (CT in 2) {
             for (CA in 0:CT) {
                 CB <- CT - CA
                 A.scale <- sqrt(tau2A[i, p] *
                   (CA == 0) + sig2A[i, p] * (CA >
                   0))
                 B.scale <- sqrt(tau2B[i, p] *
                   (CB == 0) + sig2B[i, p] * (CB >
                   0))
                 scale <- c(A.scale, B.scale)
                 if (CA == 0 & CB > 0)
                   rho <- corrA.BB[i, p]
                 if (CA > 0 & CB == 0)
                   rho <- corrB.AA[i, p]
                 if (CA > 0 & CB > 0)
                   rho <- corr[i, p]
                 lines(ellipse(x = rho, centre = c(log2(nuA[i,
                   p] + CA * phiA[i, p]), log2(nuB[i,
                   p] + CB * phiB[i, p])), scale = scale),
                   col = ellipse.col[p], lwd = 2)
             }
         }
     }
     legend("topright", lwd = 3, col = ellipse.col,
         legend = unique(plate), bty = "n")
 }
```

Look at the distribution of shifts in the predicted centers across the plates. The biggest shifts are for SNPs that have no observations in a subset of the plates. Here, we should borrow strength across plates to improve the prediction regions (TO DO).

```
R> shiftA <- log2(nuA + phiA)
R> maxA <- apply(shiftA, 1, max, na.rm = TRUE)
R> minA <- apply(shiftA, 1, min, na.rm = TRUE)
R> d <- maxA - minA
R> hist(d)
R> index <- which(d > 1)
R> plate1 <- plate2 <- rep(NA, length(index))
R> for (i in seq(along = index)) {
     plate1[i] <- which(shiftA[index[i], ] == maxA[index[i]])
     plate2[i] <- which(shiftA[index[i], ] == minA[index[i]])
 }
R> par(las = 1)
R> for (j in seq(along = index)) {
     i <- index[j]
     J1 <- grep(plate[plate1[j]], plate)
     J2 <- grep(plate[plate2[j]], plate)
     ylim <- c(6.5, 13)
     plot(log2(A[i, ]), log2(B[i, ]), pch = as.character(calls[i,
         ]), col = "grey60", cex = 0.9, ylim = ylim,
         xlim = ylim, xlab = "A", ylab = "B")
     points(log2(A[i, J1]), log2(B[i, J1]), col = "brown",
         pch = as.character(calls[i, J1]))
     points(log2(A[i, J2]), log2(B[i, J2]), col = "blue",
         pch = as.character(calls[i, J2]))
     CT <- 2
     P <- c(plate1[j], plate2[j])
     for (p in seq(along = P)) {
         if (p == 1)
             ellipse.col = "brown"
         else ellipse.col = "blue"
         for (CA in 0:CT) {
             CB <- CT - CA
             A.scale <- sqrt(tau2A[i, p] * (CA ==
                 0) + sig2A[i, p] * (CA > 0))
             B.scale <- sqrt(tau2B[i, p] * (CB ==
                 0) + sig2B[i, p] * (CB > 0))
             scale <- c(A.scale, B.scale)
             if (CA == 0 & CB > 0)
                 rho <- corrA.BB[i, p]
             if (CA > 0 & CB == 0)
                 rho <- corrB.AA[i, p]
```

```
            if (CA > 0 & CB > 0)
                rho <- corr[i, p]
            lines(ellipse(x = rho, centre = c(log2(nuA[i,
                p] + CA * phiA[i, p]), log2(nuB[i,
                p] + CB * phiB[i, p])), scale = scale),
                col = ellipse.col, lwd = 2)
        }
    }
 }
R> legend("topright", lwd = 3, col = c("brown", "blue"),
     legend = c("2 copies, plate 1", "2 copies, plate 2 "),
     bty = "n")
```

Now look at shifts for which we have at least 3 observations in each genotype cluster (TO DO).

## 3 Session information

```
R> sessionInfo()

R version 2.9.0 Under development (unstable) (2009-02-08 r47879)
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_

attached base packages:
[1] splines   stats     graphics  grDevices utils
[6] datasets  methods   base

other attached packages:
[1] SNPchip_1.7.1            oligoClasses_1.5.5
[3] genefilter_1.23.2        Biobase_2.3.10
[5] ellipse_0.3-5            genomewidesnp6Crlmm_1.0.1
[7] crlmm_1.0.42

loaded via a namespace (and not attached):
[1] affyio_1.11.3        annotate_1.21.3
[3] AnnotationDbi_1.5.15 DBI_0.2-4
[5] preprocessCore_1.5.3 RSQLite_0.7-1
[7] survival_2.34-1      tools_2.9.0
[9] xtable_1.5-4
```