Using *crlmm* to genotype data from Illumina's Infinium BeadChips

Matt Ritchie

July 2, 2009

1 Getting started

In this user guide we read in and genotype data from 40 HapMap samples which have been analyzed using Illumina's 370k Duo BeadChips. This data is available in the *hapmap370k* package. Additional chip-specific model parameters and basic SNP annotation information used by CRLMM is stored in the *human370v1c* package. These can be downloaded from http://rafalab.jhsph.edu/software.html and must be installed for the following code to work.

2 Reading in data

The function `readIdatFiles` extracts the Red and Green intensities from the binary `idat` files output by Illumina's scanning device. The file `samples370k.csv` contains information about each sample.

```
> options(width = 50)

> library(Biobase)
> library(crlmm)
> library(hapmap370k)
> data.dir = system.file("idatFiles", package = "hapmap370k")
> samples = read.csv(file.path(data.dir,
+     "samples370k.csv"), as.is = TRUE)
> samples[1:5, ]

  HapMap.Name Gender         Plate Well
1     NA06991 Female WG1000442-DNA  E11
2     NA07000 Female WG1000442-DNA  D08
3     NA10859 Female WG1000453-DNA  B02
```

```
4     NA11882 Female WG1000453-DNA  D08
5     NA06993   Male WG1000447-DNA  D11
  SentrixPosition
1    4030186347_A
2    4030186263_B
3    4019585415_B
4    4031058127_B
5    4031058211_B
```

```
> RG = readIdatFiles(samples, path = data.dir,
+     arrayInfoColNames = list(barcode = NULL,
+         position = "SentrixPosition"),
+     saveDate = TRUE)
```

Reading in this data takes approximately 90 seconds and peak memory usage was 1.2 GB of RAM on our linux system. The `RG` object is an *NChannelSet* which stores the Red and Green intensities, the number of beads and standard errors for each bead-type. The scanning date of each array is stored in the `scanDates` slot.

```
> class(RG)

[1] "NChannelSet"
attr(,"package")
[1] "Biobase"

> dim(RG)

Features  Samples
  381079       40

> slotNames(RG)

[1] "assayData"         "phenoData"
[3] "featureData"       "experimentData"
[5] "annotation"        "scanDates"
[7] ".__classVersion__"

> channelNames(RG)

[1] "G"   "Gnb" "Gse" "R"   "Rnb" "Rse"

> exprs(channel(RG, "R"))[1:5, 1:5]
```

```
         1    2    3    4     5
10008  321  170 2961 3468   262
10010 1738 3702 3105 3425    70
10025   80  101  145   29    21
10026 5043 1856 6519 8304  9872
10039 4905 2464 9080 9788 10867
```

```
> exprs(channel(RG, "G"))[1:5, 1:5]
```

```
         1    2    3    4    5
10008 4183 4484 3765 3558 6502
10010 2593   51 3824 3528 6154
10025 2768 2322 3435 3471 3608
10026  216 2840  211  164  188
10039  297 3016  345  361  380
```

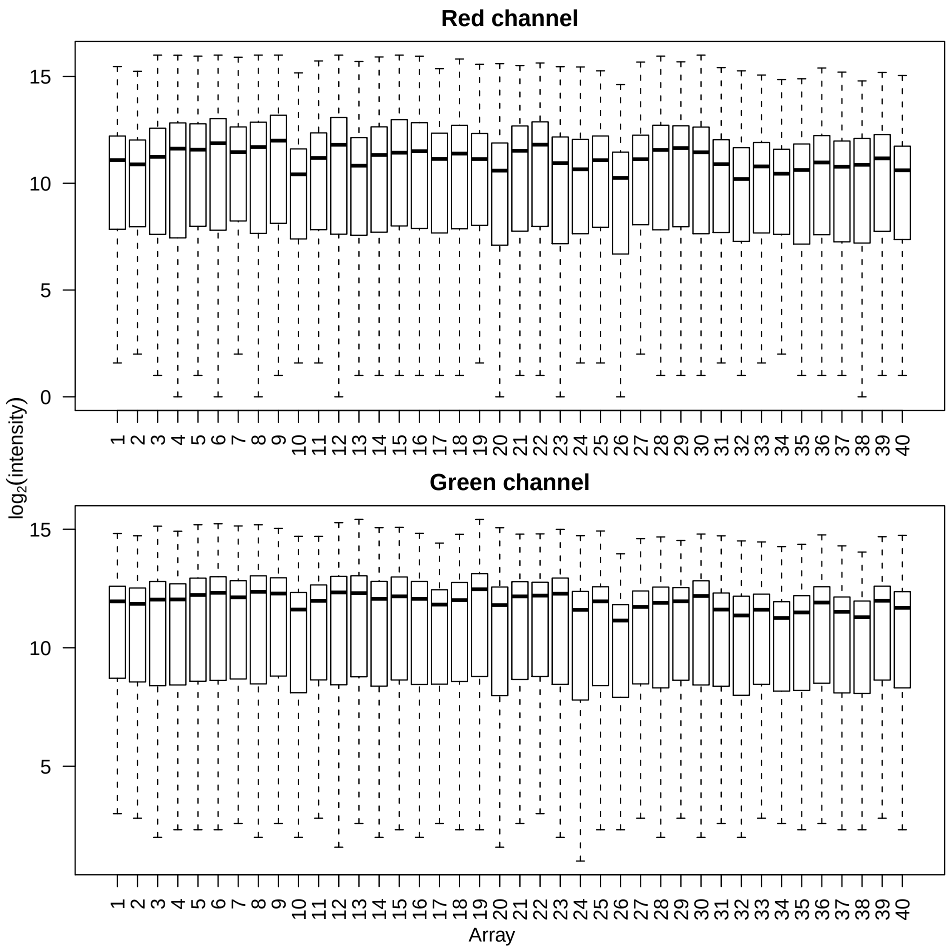
```
> pd = pData(RG)
> pd[1:5, ]
```

```
  HapMap.Name Gender         Plate Well
1     NA06991 Female WG1000442-DNA  E11
2     NA07000 Female WG1000442-DNA  D08
3     NA10859 Female WG1000453-DNA  B02
4     NA11882 Female WG1000453-DNA  D08
5     NA06993   Male WG1000447-DNA  D11
  SentrixPosition
1    4030186347_A
2    4030186263_B
3    4019585415_B
4    4031058127_B
5    4031058211_B
```

```
> scandatetime = strptime(scanDates(RG),
+     "%m/%d/%Y %H:%M:%S %p")
> datescanned = substr(scandatetime, 1,
+     10)
> scanbatch = factor(datescanned)
> levels(scanbatch) = 1:16
> scanbatch = as.numeric(scanbatch)
```

Plots of the summarised data can be easily generated to check for arrays with poor signal.

```
> par(mfrow = c(2, 1), mai = c(0.4, 0.4,
+     0.4, 0.1), oma = c(1, 1, 0, 0))
> boxplot(log2(exprs(channel(RG, "R"))),
+     xlab = "Array", ylab = "", main = "Red channel",
+     outline = FALSE, las = 2)
> boxplot(log2(exprs(channel(RG, "G"))),
+     xlab = "Array", ylab = "", main = "Green channel",
+     outline = FALSE, las = 2)
> mtext(expression(log[2](intensity)), side = 2,
+     outer = TRUE)
> mtext("Array", side = 1, outer = TRUE)
```



3 Genotyping

Next we use the function `crlmmIllumina` which performs preprocessing followed by genotyping using the CRLMM algorithm.

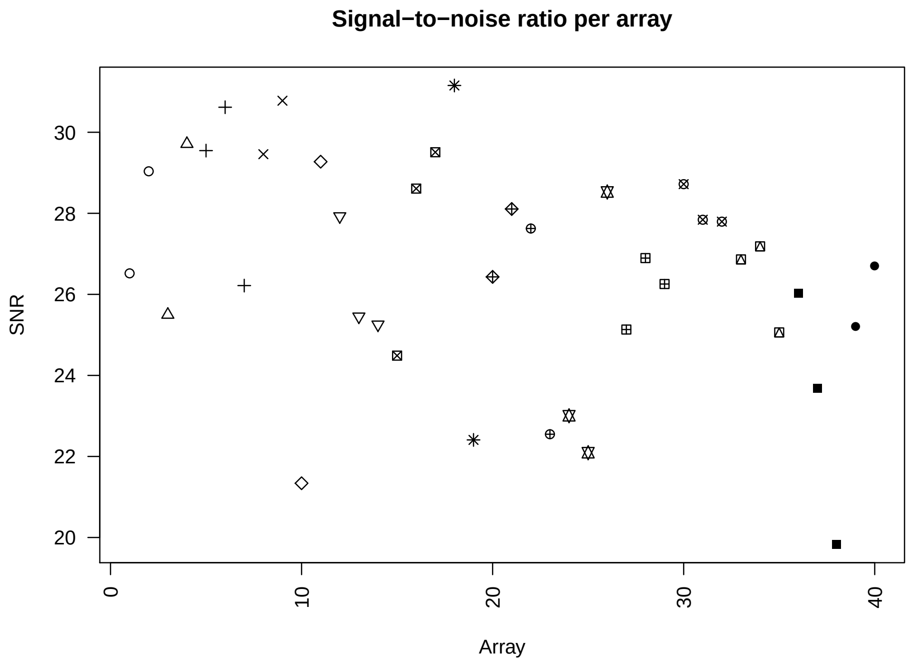
```
> crlmmResult = crlmmIllumina(RG = RG, cdfName = "human370v1c",
+     sns = pData(RG)$ID, returnParams = TRUE)
```

This analysis took 470 seconds to complete and peak memory usage was 3.3 GB on our system. The output stored in `crlmmResult` is a *SnpSet* object.

```
> class(crlmmResult)

[1] "SnpSet"
attr(,"package")
[1] "Biobase"

> dim(crlmmResult)

Features  Samples
  346451       40

> slotNames(crlmmResult)

[1] "assayData"         "phenoData"
[3] "featureData"       "experimentData"
[5] "annotation"        "scanDates"
[7] ".__classVersion__"

> calls(crlmmResult)[1:10, 1:5]

           1 2 3 4 5
rs12354060 3 3 3 3 3
rs6650104  3 1 1 3 1
rs12184279 3 3 3 3 3
rs12564807 3 3 3 3 3
rs3115860  1 1 1 3 2
rs3115850  2 3 3 2 2
rs7515489  3 3 3 3 3
rs12124819 1 2 2 1 1
rs17160939 1 1 1 1 1
rs12086311 3 3 3 3 3
```

Plotting the *SNR* reveals no obvious batch effects in this data set (different symbols are used for arrays scanned on different days).

```
> plot(crlmmResult[["SNR"]], pch = scanbatch,
+     xlab = "Array", ylab = "SNR", main = "Signal-to-noise ratio per array",
+     las = 2)
```



4 System information

This analysis was carried out on a linux machine with 32GB of RAM using the following packages:

```
> sessionInfo()

R version 2.9.1 RC (2009-06-25 r48837)
x86_64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.iso885915;LC_NUMERIC=C;LC_TIME=en_US.iso885915;LC_COLLATE=en_US.iso88591

attached base packages:
[1] tools     stats     graphics  grDevices
[5] utils     datasets  methods   base

other attached packages:
[1] human370v1cCrlmm_1.0.0 hapmap370k_1.0
[3] crlmm_1.3.7            Biobase_2.5.4
[5] weaver_1.10.0          codetools_0.2-2
[7] digest_0.3.1
```

```
loaded via a namespace (and not attached):
 [1] affyio_1.12.0        annotate_1.22.0
 [3] AnnotationDbi_1.6.1  Biostrings_2.12.7
 [5] DBI_0.2-4            ellipse_0.3-5
 [7] genefilter_1.24.2    IRanges_1.2.3
 [9] mvtnorm_0.9-7        oligoClasses_1.6.0
[11] preprocessCore_1.6.0 RSQLite_0.7-1
[13] splines_2.9.1        survival_2.35-4
[15] xtable_1.5-5
```