illuminaio

M.L. Smith, K.D. Hansen

October 14, 2013

Introduction

illuminaio is designed to provide a single package containing routines for importing data from Illumina BeadArray platforms into *R*. The intention is for *illuminaio* to provide developers of downstream analysis packages with a mechanism for extracting all possible information from IDAT files in a relatively straight forward fashion. The choice of data to retain and how it should be stored is then left for the end user to decide.

This vignette gives examples of how data files can be read and discusses the values that are returned, which will vary depending upon the BeadArray platform used. It also demonstrates that the values extracted by *illuminaio* are the same as those returned when using Illumina's own GenomeStudio software.

Reading Data

Expression Array

The code below gives an example of how an IDAT file (in this case from a Human expression array) can be read, and explores the information that is extracted.

```
> library(illuminaio)
> library(IlluminaDataTestFiles)
> idatFile <- system.file("extdata", "idat", "4343238080_A_Grn.idat",
+                         package = "IlluminaDataTestFiles")
> idat <- readIDAT(idatFile)
```

The first two lines above load libraries require for this vignette. Firstly this package and then *IlluminaDataTestFiles*, a small data package containing the example files that are used throughout this vignette. The third line generates the complete path to one such file. We then use the function `readIDAT` to read the file. This function take only a single argument, the file's path. Although IDAT files are found in multiple formats, `readIDAT` is able to determine this and will call the appropriate reading routine internally. The returned object is a list, the contents of which is explored below.

```
> names(idat)
```

```
[1] "Barcode"  "Section"  "ChipType" "Quants"   "RunInfo"
```

Using the `names` command above lists the data extracted from the file. In this case `Barcode` and `Section` are the identifiers for the BeadChip and can usually be found in the file name as well. `ChipType` describes the BeadArray platform this file was generated from. The `RunInfo` slot holds information about the processing performed on the array, most notably the date upon which it was scanned. Such information may be useful if one is attempting to identify batch effects amongst multiple samples. `Quants` is where the per-bead-type values are found. The commands below assign the `Quants` values to a new variable for convenience, and then print the first six entries from the resulting data.frame.

```
> idatData <- idat$Quants
> head(idatData)
```

```
  MeanBinData TrimmedMeanBinData DevBinData MedianBinData BackgroundBinData
1   179.16405          180.45245   41.46048     177.66107          657.5565
2    50.08747           50.42234   18.76339      46.82187          657.1952
3    47.91959           48.55188   12.52172      49.05127          657.0317
4    52.07837           52.34163   15.55780      47.36008          657.4821
5    57.80818           58.02231   17.63354      58.88446          657.4556
6    43.08765           43.68334   13.46260      43.53027          657.2791
  BackgroundDevBinData CodesBinData NumBeadsBinData NumGoodBeadsBinData IllumicodeBinData
1             1.130837        10008              47                  46             10008
2             1.345547        10010              42                  42             10010
3             1.438468        10014              48                  44             10014
4             1.250080        10017              40                  39             10017
5             1.411696        10019              57                  54             10019
6             1.081928        10020              46                  43             10020
```

We can see that for this expression array a total of 10 values are returned for each bead-type. The column headers are pulled directly from the IDAT file and do not directly match with how things are commonly labelled in GenomeStudio, although for the most part they are relatively easy to decipher. The `CodeBinData`, `MeanBinData` and `NumGoodBeadsBinData` columns are those that are reported by default in GenomeStudio and correspond the ProbeID, AVG_Signal and NBEADS values respectively. `DevBinData` gives the standard deviation for the bead-type and can be used the generate the BEAD_STDERR values GenomeStudio reports. The remaining columns give additional information and are discussed in the file `EncryptedFormat.pdf` that also accompanies this package.

Genotyping Array

The example above focused on reading an IDAT file produced by scanning an expression array. To highlight some of the differences in output we shall now read a file from an Infinium genotyping array.

fixme: I should check exactly which platform this is from - MLS

```
> genotypeIdatFile <- system.file("extdata", "idat", "5723646052_R02C02_Grn.idat",
+                              package = "IlluminaDataTestFiles")
> genotypeIdat <- readIDAT(genotypeIdatFile)
> names(genotypeIdat)
```

```
 [1] "fileSize"      "versionNumber" "nFields"       "fields"        "nSNPsRead"
 [6] "Quants"        "MidBlock"      "RedGreen"      "Barcode"       "ChipType"
[11] "RunInfo"       "Unknowns"
```

The reading of the file proceeds in exactly the same way as before and a list is again returned. However, there are several more data fields returned. Again `Quants` is where the per-bead-type values are stored.

```
> head(genotypeIdat$Quants)
```

```
         Mean   SD NBeads
10600313  415  231     12
10600322 9685 1040      9
10600328 1647  398     11
10600336 3680  624     17
10600345 3616  173      5
10600353 4578 1122     14
```

For genotyping arrays only the four typically reported values are contained within the IDAT file and their column names more closely resemble those that are found in GenomeStudio.

Comparison with GenomeStudio

Now we shall compare the values extracted by *illuminaio* with those reported by Illumina's GenomeStudio software, to ensure our file reading routines are performing correctly.

Importing GenomeStudio Values

```
> gStudio <- read.delim(
+         "http://compbio.sysbiol.cam.ac.uk/Resources/IDATreader/4343238080_A_ProbeSummary.txt",
+         sep = "\t", header = TRUE)
> idatData <- idatData[which(idatData[,"CodesBinData"] %in% gStudio[,"ProbeID"]),]
> gStudio <- gStudio[match(idatData[,"CodesBinData"], gStudio[,"ProbeID"]),]
```

The first line above reads a file that was produced by reading the IDAT file into GenomeStudio and then immediately exporting that data as a tab separated text file. No other processing was performed on the data. *fixme: Currently this file is stored on a web server, although the intention is to include it in the illuminaDataTestFile package.*

However, the two datasets are not quite compatible at the moment. Reading directly from an IDAT file returns values for several bead-types that serve as internal controls and are not annotated by Illumina. These bead-types are excluded automatically by GenomeStudio, so the second line above identifies and removes them from our *illuminaio* data. The two data sets should now contain the same number of bead-types.

The inclusion of these extra bead-types is not the only difference, they are also in different orders. Bead-types are extracted in numerical order from IDAT files, but the GenomeStudio output is sorted alphabetically. The third line reorders the GenomeStudio values to match those from *illuminaio*, making our comparison slightly easier.

Performing Comparison

The code below produces the two plots seen in Figure 1.

```
> par(mfrow = c(1,2))
> plot(idatData[, "MeanBinData"], gStudio[, "X4343238080_A.AVG_Signal"],
+      xlab = "illuminaio", ylab = "GenomeStudio")
> identical(idatData[, "MeanBinData"], gStudio[, "X4343238080_A.AVG_Signal"])
```

```
[1] FALSE
```

```
> hist(idatData[, "MeanBinData"]- gStudio[, "X4343238080_A.AVG_Signal"],
+      breaks = 100, main = "", xlab = "Difference")
```

The first plot shows the summarized bead-intensity values extracted by *illuminaio* on the horizontal axis against Genomestudio's values on the vertical axis. We can see they are highly similar. However, they are not identical, as shown by the third line above. The second plot visualises the distribution of the differences between the two sets of values, showing them to be small. These are most likely introduced by rounding performed by GenomeStudio that is not carried out by *illuminaio*.

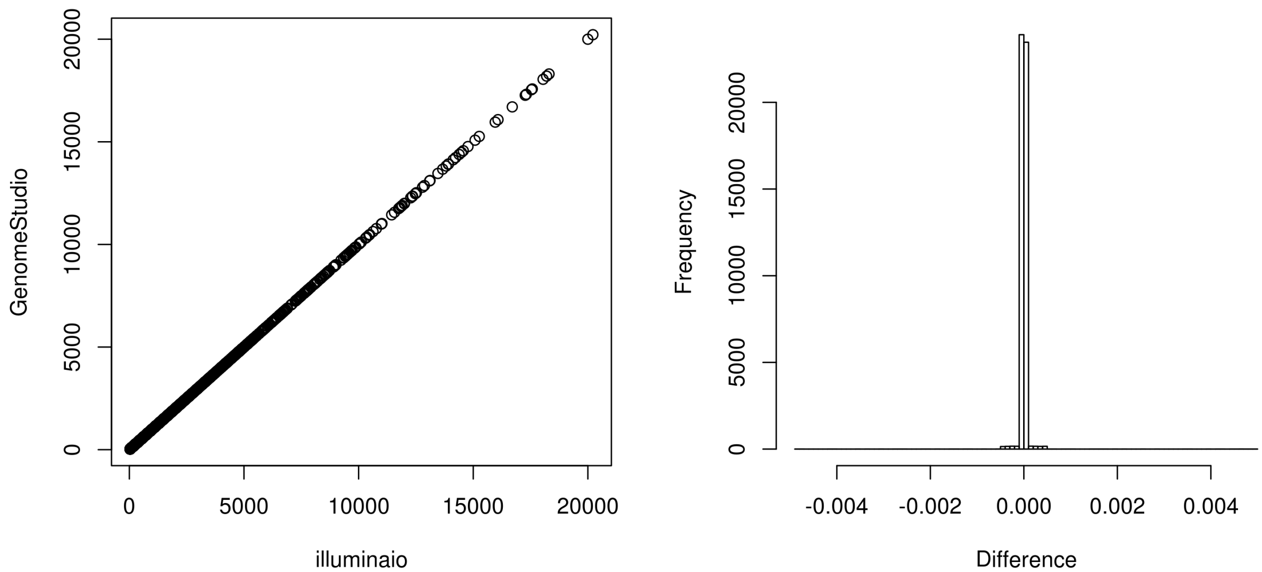

Figure 1: Comparing values obtained by *illuminaio* and GenomeStudio

Session info

Here is the output of `sessionInfo` on the system on which this document was compiled:

```
> toLatex(sessionInfo())
```

- R version 3.0.2 (2013-09-25), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: IlluminaDataTestFiles 0.99.0, illuminaio 0.4.0
- Loaded via a namespace (and not attached): BiocStyle 1.0.0, base64 1.1, tools 3.0.2