

Using the charm package to estimate DNA methylation levels and find differentially methylated regions

Martin Aryee*, Peter Murakami, Rafael Irizarry

March, 2010

Johns Hopkins School of Medicine / Johns Hopkins School of Public Health
Baltimore, MD, USA

1 Introduction

The Bioconductor package **charm** can be used to analyze DNA methylation data generated using McrBC fractionation and two-color Nimblegen microarrays. It is customized for use with the from the custom CHARM microarray [1], but can also be applied to many other Nimblegen designs.

Functions include:

- Quality control
- Finding suitable control probes for normalization
- Percentage methylation estimates
- Identification of differentially methylated regions

As input we will need raw Nimblegen data (.xys) files and a corresponding annotation package built with pdInfoBuilder. This vignette uses the following packages:

- **charm**: contains the analysis functions
- **charmData**: an example dataset
- **pd.charm.hg18.example**: the annotation package for the example dataset
- **BSgenome.Hsapiens.UCSC.hg18**: A BSgenome object containing genomic sequence used for finding non-CpG control probes

Each sample is represented by two xys files corresponding to the untreated (green) and methyl-depleted (red) channels. The 532.xys and 635.xys suffixes indicate the green and red channels respectively.

*aryee@jhu

2 Analyzing data from the custom CHARM microarray

Load the charm package:

```
R> library(charm)
R> library(charmData)
```

3 Read in raw data

Get the name of your data directory (in this case, the example data):

```
R> dataDir <- system.file("data", package = "charmData")
R> dataDir
```

```
[1] "/Library/Frameworks/R.framework/Resources/library/charmData/data"
```

First we read in the sample description file:

```
R> phenodataDir <- system.file("extdata", package = "charmData")
R> pd <- read.delim(file.path(phenodataDir, "phenodata.txt"))
R> phenodataDir
```

```
[1] "/Library/Frameworks/R.framework/Resources/library/charmData/extdata"
```

```
R> pd
```

	filename	sampleID	tissue
1	136421_532.xys	441_liver	liver
2	136421_635.xys	441_liver	liver
3	136600_532.xys	449_spleen	spleen
4	136600_635.xys	449_spleen	spleen
5	3788602_532.xys	449_liver	liver
6	3788602_635.xys	449_liver	liver
7	3822402_532.xys	441_spleen	spleen
8	3822402_635.xys	441_spleen	spleen
9	5739902_532.xys	624_colon	colon
10	5739902_635.xys	624_colon	colon
11	5875602_532.xys	441_colon	colon
12	5875602_635.xys	441_colon	colon

A valid sample description file should contain at least the following (arbitrarily named) columns:

- a filename column
- a sample ID column
- a group label column (optional)

The sample ID column is used to pair the methyl-depleted and untreated data files for each sample. The group label column is used when identifying differentially methylated regions between experimental groups.

The `validatePd` function can be used to validate the sample description file. When called with only a sample description data frame and no further options `validatePd` will try to guess the contents of the columns.

```
R> res <- validatePd(pd)
```

Now we read in the raw data. The `readCharm` command makes the assumption (unless told otherwise) that the two xys files for a sample have the same file name up to the suffixes 532.xys (untreated) and 635.xys (methyl-depleted).

```
R> rawData <- readCharm(files = pd$filename, path = dataDir,
  sampleKey = pd)
```

```
Checking designs for each XYS file... Done.
```

```
Allocating memory... Done.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/136421_532.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/136600_532.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/3788602_532.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/3822402_532.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/5739902_532.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/5875602_532.xys.
```

```
Checking designs for each XYS file... Done.
```

```
Allocating memory... Done.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/136421_635.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/136600_635.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/3788602_635.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/3822402_635.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/5739902_635.xys.
```

```
Reading /Library/Frameworks/R.framework/Resources/library/charmData/data/5875602_635.xys.
```

```
R> rawData
```

```
TilingFeatureSet (storageMode: lockedEnvironment)
```

```
assayData: 243129 features, 6 samples
```

```
  element names: channel1, channel2
```

```
protocolData: none
```

```
phenoData
```

```
  rowNames: 136421, 136600, ..., 5875602 (6 total)
```

```
  varLabels and varMetadata description:
```

```
    sampleID: NA
```

```
    tissue: NA
```

```
    arrayUT: Untreated channel file name
```

```
    arrayMD: Methyl-depleted channel file name
```

```
  additional varMetadata: channel
```

```
featureData: none
experimentData: use 'experimentData(object)'
Annotation: pd.charm.hg18.example
```

4 Array quality assessment

We can calculate array quality scores and generate a pdf report with the `qcReport` command.

A useful quick way of assessing data quality is to examine the untreated channel where we expect every probe to have signal. Very low signal intensities on all or part of an array can indicate problems with hybridization or scanning. The CHARM array and many other designs include background probes that do not match any genomic sequence. Any signal at these background probes can be assumed to be the result of optical noise or cross-hybridization. Since the untreated channel contains total DNA a successful hybridization would have strong signal for all untreated channel genomic probes. The array signal quality score (`pmSignal`) is calculated as the average percentile rank of the signal probes among these background probes. A score of 100 means all signal probes rank above all background probes (the ideal scenario).

```
R> qual <- qcReport(rawData, file = "qcReport.pdf")
R> qual
```

	pmSignal	sd1	sd2
136421	78.56437	0.1950274	0.1932112
136600	81.46541	0.1755225	0.1227921
3788602	83.95419	0.1249030	0.2409803
3822402	81.43751	0.1180708	0.1824810
5739902	82.55727	0.1490854	0.2035761
5875602	79.38069	0.3130266	0.3962373

The PDF quality report is shown in Appendix A. Three quality metrics are calculated for each array:

1. Average signal strength: the average percentile rank of untreated channel signal probes among the background (anti-genomic) probes.
2. Untreated channel signal standard deviation. The array is divided into a series of rectangular blocks and the average signal level calculated for each. Since probes are arranged randomly on the array there should be no large differences between blocks. Arrays with spatial artifacts have a large standard deviation between blocks.
3. Methyl-depleted channel signal standard deviation.

5 Percentage methylation estimates and differentially methylated regions (DMRs)

We now calculate probe-level percentage methylation estimates for each sample. As a first step we need to identify a suitable set of unmethylated control probes from CpG-free regions to be used in normalization.

```
R> library(BSgenome.Hsapiens.UCSC.hg18)
R> ctrlIdx <- getControlIndex(rawData, subject = Hsapiens)
```

The minimal code required to estimate methylation would be `p <- methp(rawData, controlIndex=ctrlIdx)`. However, it is often useful to get `methp` to produce a series of diagnostic density plots to help identify non-hybridization quality issues. The `plotDensity` option specifies the name of the output pdf file, and the optional `plotDensityGroups` can be used to give groups different colors.

```
R> grp <- pData(rawData)$tissue
R> p <- methp(rawData, controlIndex = ctrlIdx, plotDensity = "density.pdf",
  plotDensityGroups = grp)
R> head(p)
```

```
      136421    136600    3788602    3822402    5739902
[1,] 0.2185571 0.3835276 0.3886250 0.5428861 0.3788786
[2,] 0.8015920 0.6426700 0.3546513 0.8644451 0.5337523
[3,] 0.1448220 0.1198934 0.1922395 0.1883505 0.2605561
[4,] 0.7273223 0.4706128 0.4538511 0.4532933 0.3815467
[5,] 0.6506827 0.5270123 0.4106207 0.4303267 0.3997432
[6,] 0.6242838 0.7464497 0.7420501 0.6961507 0.8640721
      5875602
[1,] 0.2927198
[2,] 0.8846106
[3,] 0.6638752
[4,] 0.4589445
[5,] 0.3892579
[6,] 0.8106961
```

The density plots are shown in Appendix B.

We can now identify differentially methylated regions using `dmrFinder`:

```
R> dmr <- dmrFinder(rawData, p = p, groups = grp,
  compare = c("colon", "liver", "colon", "spleen"))
```

```
R> names(dmr)
```

```
[1] "tabs"      "p"          "l"
[4] "chr"       "pos"        "pns"
[7] "index"     "controlIndex" "gm"
[10] "groups"    "args"       "comps"
[13] "package"
```

```
R> names(dmr$tabs)

[1] "colon-liver" "colon-spleen"

R> head(dmr$tabs[[1]])
```

	chr	start	end	p1	p2
319	chr12	88272817	88273811	0.8446471	0.1917546
358	chr13	27090247	27091263	0.7805552	0.1855020
1751	chr6	52637786	52638747	0.7237363	0.1876064
473	chr15	58673084	58673750	0.8252984	0.2904679
131	chr11	14620645	14621065	0.8431744	0.3469629
1116	chr20	60187462	60188125	0.8325999	0.1930089

	regionName	indexStart	indexEnd	area
319	chr12:88266873-88274292	40465	40488	15.669421
358	chr13:27090144-27095500	45272	45291	11.901064
1751	chr6:52635302-52638967	160820	160843	12.867118
473	chr15:58669815-58674073	57657	57675	10.161781
131	chr11:14620645-14623686	28438	28450	6.450749
1116	chr20:60143957-60188418	122601	122620	12.791820

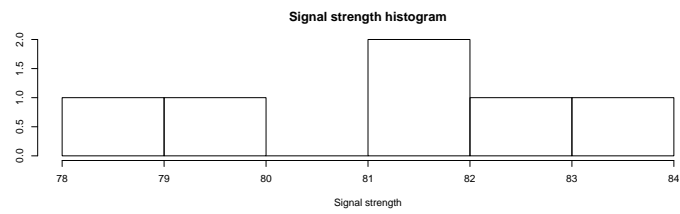
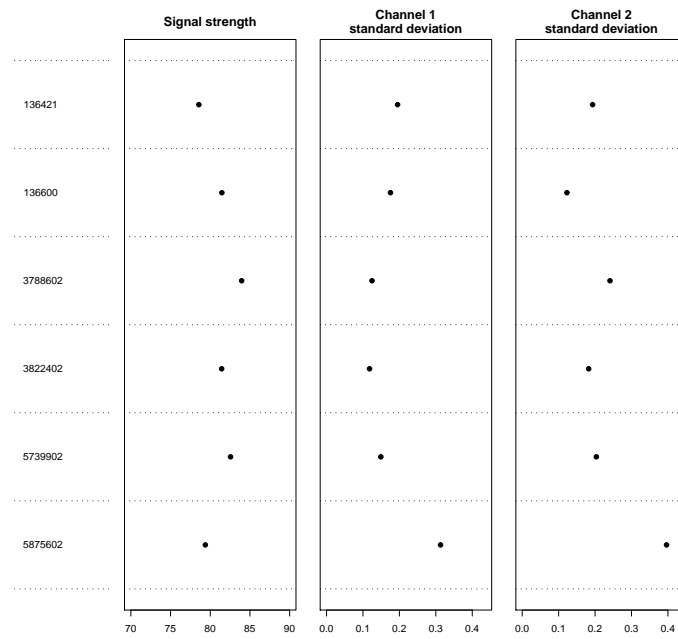
	ttarea
319	782.1598
358	700.7226
1751	665.3653
473	520.4626
131	489.6030
1116	474.9341

When called without the `compare` option, `dmrFinder` performs all pairwise comparisons between the groups.

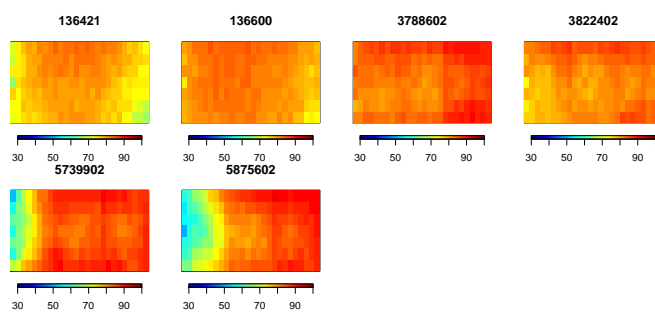
References

- [1] Irizarry et al. Comprehensive high-throughput arrays for relative methylation (charm). *Genome Research*, 18(5):780–790, 2008.

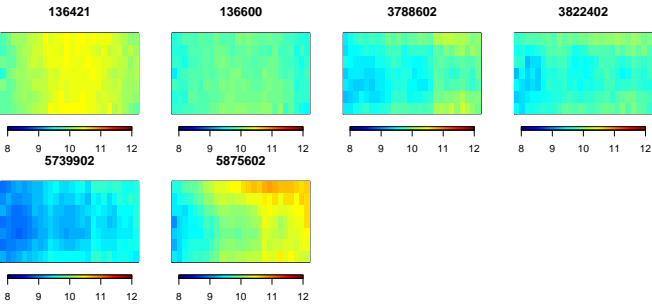
6 Appendix A: Quality report



Untreated Channel: PM probe quality

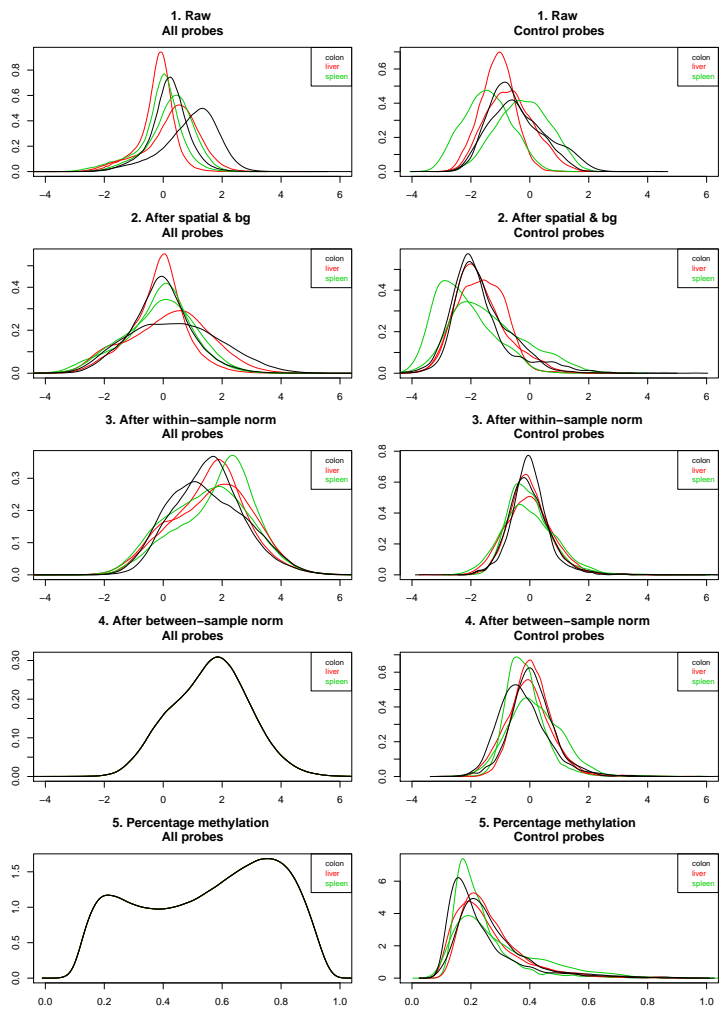


Enriched Channel: PM signal intensity



7 Appendix B: Density plots

Each row corresponds to one stage of the normalization process (Raw data, After spatial and background correction, after within-sample normalization, after between-sample normalization, percentage methylation estimates). The left column shows all probes, while the right column shows control probes.



8 Details

This document was written using:

```
R> sessionInfo()
```

```
R version 2.11.0 RC (2010-04-21 r51799)
i386-apple-darwin9.8.0
```

```
locale:
```

```
[1] C/en_US.UTF-8/C/C/C/C
```

```
attached base packages:
```

```
[1] tools      stats      graphics  grDevices  utils
[6] datasets  methods   base
```

```
other attached packages:
```

```
[1] BSgenome.Hsapiens.UCSC.hg18_1.3.16
[2] BSgenome_1.16.1
[3] Biostrings_2.16.0
[4] GenomicRanges_1.0.1
[5] IRanges_1.6.2
[6] charmData_0.99.1
[7] pd.charm.hg18.example_0.99.2
[8] oligo_1.12.0
[9] oligoClasses_1.10.0
[10] RSQLite_0.9-0
[11] DBI_0.2-5
[12] charm_1.0.1
[13] fields_6.01
[14] spam_0.21-0
[15] SQN_1.0
[16] nor1mix_1.1-2
[17] mclust_3.4.4
[18] Biobase_2.8.0
```

```
loaded via a namespace (and not attached):
```

```
[1] MASS_7.3-5          affxparser_1.20.0
[3] affyio_1.16.0       bit_1.1-4
[5] ff_2.1-2            gtools_2.6.2
[7] multtest_2.4.0      preprocessCore_1.10.0
[9] siggenes_1.22.0     splines_2.11.0
[11] survival_2.35-8
```