


Authors: Mark Robinson

Created: 2008-11-25

Last updated: 2008-11-25

 To help support this work, please consider citing the following relevant references in your publications or talks whenever using their methods or results:

- H. Bengtsson, K. Simpson, J. Bullard, et al. *aroma.affymetrix: A generic framework in R for analyzing small to very large Affymetrix data sets in bounded memory*. Tech. rep. 745. Department of Statistics, University of California, Berkeley, Feb. 2008.

## Setup

### Raw data

Test set: tissues (11 human tissues, each with 3 biological replicates, run by Affymetrix and publicly available from [www.affymetrix.com](http://www.affymetrix.com))

Path: `rawData/tissues/HuGene-1_0-st-v1/`

### Annotation data

For Gene 1.0 ST array, we will use a binary-converted version of the 'unsupported' CDF that is provided by Affymetrix. Instead of creating these yourself, you can simply download it from the following links, depending on which organism you are working with:

- Human: `HuGene-1_0-st-v1,r3.cdf`
- Mouse: `MoGene-1_0-st-v1,r3.cdf`
- Rat: `RaGene-1_0-st-v1,r3.cdf`

For the PM plate arrays (these are newer PM-only versions of the older generation HG-U133, MG-430, RG-230 chips), you can get a binary version of the CDF directly from the "Library Files" archive from Affymetrix. See the `HT_HG-U133_Plus_PM` page for details on the human array and see the Affymetrix web site for details. The below scripts should work by just replacing the "chipType" and tags where appropriate.

Be sure to put the CDF file in your `annotationData/chipTypes/<chipType>/` where `<chipType>` is the platform you are using (e.g. `HuGene-1_0-st-v1` for Human Gene 1.0 ST, `HT_HG-U133_Plus_PM` for the human PM plate arrays etc.)

Important: Make sure you replicate the structure outlined in Setup - this will allow the Aroma framework to easily find your data sets and CDFs.

## Low-level analysis

Most of what follows is very similar to the use case for Human Exon arrays. In fact, this vignette was created by cutting and pasting the Human Exon array analysis page and modified. Thanks are due to Elizabeth Purdom and Ken Simpson for the original document.

```
library("aroma.affymetrix")
verbose <- Arguments$getVerbose(-8, timestamp=TRUE)
```

## Getting annotation data files

In this vignette we will use the standard CDF (which you can download from the link above). We setup the CDF as:

```
chipType <- "HuGene-1_0-st-v1"
cdf <- AffymetrixCdfFile$byChipType(chipType, tags="r3")
print(cdf)
```

This gives:

```
AffymetrixCdfFile:
Path: annotationData/chipTypes/HuGene-1_0-st-v1
Filename: HuGene-1_0-st-v1,r3.cdf
Filesize: 16.67MB
Chip type: HuGene-1_0-st-v1,r3
RAM: 0.00MB
File format: v4 (binary; XDA)
Dimension: 1050x1050
Number of cells: 1102500
Number of units: 33252
Cells per unit: 33.16
Number of QC units: 0
```

## Defining CEL set

Next we setup the CEL set with the above custom CDF, assuming the data has been

```
cs <- AffymetrixCelSet$byName("tissues", cdf=cdf)
print(cs)
```

This gives:

```
AffymetrixCelSet:
Name: tissues
Tags:
Path: rawData/tissues/HuGene-1_0-st-v1
Platform: Affymetrix
```

```
Chip type: HuGene-1_0-st-v1,r3
Number of arrays: 33
Names: TisMap_Brain_01_v1_WTGene1, TisMap_Brain_02_v1_WTGene1,
..., TisMap_Thyroid_03_v1_WTGene1
Time period: 2006-10-03 12:29:27 -- 2006-10-05 13:39:50
Total file size: 349.03MB
RAM: 0.03MB
```

## Background Adjustment and Normalization

In order to do RMA background correction, we setup a correction method and run it by:

```
bc <- RmaBackgroundCorrection(cs)
csBC <- process(bc, verbose=verbose)
```

We then setup a quantile normalization method:

```
qn <- QuantileNormalization(csBC, typesToUpdate="pm")
print(qn)
```

which gives:

```
QuantileNormalization:
Data set: tissues
Input tags: RBC
User tags: *
Asterisk ('*') tags: QN
Output tags: RBC,QN
Number of files: 33 (349.03MB)
Platform: Affymetrix
Chip type: HuGene-1_0-st-v1,r3
Algorithm parameters: (subsetToUpdate: NULL, typesToUpdate: chr "pm",
subsetToAvg: NULL, typesToAvg: chr "pm", .targetDistribution: NULL)
Output path: probeData/tissues,RBC,QN/HuGene-1_0-st-v1
Is done: FALSE
RAM: 0.00MB
```

and we then run it by:

```
csN <- process(qn, verbose=verbose)
```

Then, `print(csN)` gives:

```
AffymetrixCelSet:
Name: tissues
Tags: RBC,QN
Path: probeData/tissues,RBC,QN/HuGene-1_0-st-v1
Platform: Affymetrix
```

```
Chip type: HuGene-1_0-st-v1,r3
Number of arrays: 33
Names: TisMap_Brain_01_v1_WTGene1, TisMap_Brain_02_v1_WTGene1,
..., TisMap_Thyroid_03_v1_WTGene1
Time period: 2006-10-03 12:29:27 -- 2006-10-05 13:39:50
Total file size: 349.03MB
RAM: 0.03MB
```

## Summarization

Unlike Exon arrays, the Gene arrays do not (at least by default) have information within the CDF regarding exons, or more specifically, the Gene arrays do not have separate probesets within a transcript cluster and the Exon CDFs do. There has been some recent work that suggests you can use the Gene arrays to do splicing analysis. More on that to come. Here, we'll just fit the RMA "probe-level model" (PLM) in order to get gene-level summaries for each gene. To do this, you would call:

```
plm <- RmaPlm(csN)
print(plm)
```

This gives

```
RmaPlm:
Data set: tissues
Chip type: HuGene-1_0-st-v1,r3
Input tags: RBC,QN
Output tags: RBC,QN,RMA
Parameters: (probeModel: chr "pm"; shift: num 0; flavor: chr "affyPLM"
treatNAsAs: chr "weights").
Path: plmData/tissues,RBC,QN,RMA/HuGene-1_0-st-v1
RAM: 0.00MB
```

To actually fit the PLM to all of the data, do:

```
fit(plm, verbose=verbose)
```

## Quality assessment of PLM fit

To examine NUSE and RLE plots, do:

```
qam <- QualityAssessmentModel(plm)
plotNuse(qam)
plotRle(qam)
```

To extract the estimates, you can use `extractDataFrame()` on the `ChipEffectSet` object that corresponds to the `plm` object:

```
ces <- getChipEffectSet(plm)
gExprs <- extractDataFrame(ces, units=1:3, addNames=TRUE)
```

This will give a `data.frame` with three rows, each row corresponding to a unit/transcript. To get all units, choose `units=NULL` (default). The `addNames=TRUE` argument adds the unit and group names to the entries of the data frame, which will take a bit longer the first time you analyze a chip type.

A few other considerations you may wish to think about:

- Notice that the data frame you get from `extractDataFrame()` gives the gene-level summaries on the linear scale. Typically, you'll want to take 'log2' of those for use with clustering, limma, or other tools.
- You may wish to consult the 'library' files from Affymetrix. Of the 33,252 units in the human CDF file, there are a few thousand that represent control spots and so on. Specifically, the 'Transcript Cluster Annotations CSV' file will contain this and other relevant information with respect to the probesets.

## Appendix