# Microarray data quality metrics

Audrey Kauffmann

July 22, 2008

## 1 Introduction

The workshop will provide an overview of diagnostic plots and metrics for assessing the quality of microarray datasets. The use of the arrayQualityMetrics package allows the production of a report assessing the quality of the experiments. The input object of the function *arrayQualityMetrics* can be an *AffyBatch*, an *ExpressionSet* for non Affymetrix one channel assays, a *NChannelSet* for dual channels assays, or a *BeadLevelList* for Illumina bead arrays.

## 2 Quality report for AffyBatch objects

In this example, we will use a public data set from the ArrayExpress repository.
If you have the development version of R and the new development package ArrayExpress, you can create an object the following way:

```
> #library("ArrayExpress")
> #AEset = ArrayExpress("E-MEXP-461")
> #save(AEset,file="AEset.RData")
```

If not, you can just load the prebuilt object:

```
> load("AEset.RData")
> class(AEset)
```

*AffyBatch* is an S4 class and `AEset` is an instance of this class. By using the function *ArrayExpress*, we have created an object containing the expression values and the sample annotations. The sample annotation is obtained from the sdrf file from the database and is stored in the `phenoData` slot of the created object.

```
> AEset
> colnames(pData(AEset))
```

```
 [1] "Source.Name"
 [2] "Characteristics..CellLine."
 [3] "Characteristics..OrganismPart."
 [4] "Characteristics..Sex."
 [5] "Term.Source.REF"
 [6] "Characteristics..BioSourceType."
 [7] "Term.Source.REF.1"
 [8] "Characteristics..TargetedCellType."
 [9] "Characteristics..DevelopmentalStage."
[10] "Characteristics..Organism."
[11] "Term.Source.REF.2"
[12] "grow.Protocol.REF"
[13] "Sample.Name"
[14] "nucleic_acid_extraction.Protocol.REF"
[15] "Extract.Name"
[16] "Material.Type"
[17] "labeling.Protocol.REF"
[18] "LabeledExtract.Name"
[19] "Label"
[20] "Material.Type.1"
[21] "hybridization.Protocol.REF"
[22] "feature_extraction.Protocol.REF"
[23] "Hybridization.Name"
[24] "Array.Design.REF"
[25] "Term.Source.REF.3"
[26] "Comment..Array.Design.URI."
[27] "FactorValue..RNA.extracted."
[28] "FactorValue..cell.line."
[29] "Scan.Name"
[30] "Array.Data.File"
[31] "Comment..ArrayExpress.FTP.file."
[32] "Comment..ArrayExpress.Data.Retrieval.URI."
[33] "bioassay_data_transformation.Protocol.REF"
[34] "Derived.Array.Data.Matrix.File"
[35] "Comment..Derived.ArrayExpress.FTP.file."
[36] "Comment..Derived.ArrayExpress.Data.Retrieval.URI."
```

## 2.1 Report production

To produce a quality report, the function *arrayQualityMetrics* is called with the following arguments:

- *expressionset*: is an *ExpressionSet*, an *AffyBatch*, a *NChannelSet* or a *BeadLevelList*.

- *outdir*: is the directory in which the result files are created.
- *force*: if TRUE and 'outdir' already exists, 'outdir' will be overwritten.
- *do.logtransform*: if TRUE, the data are log transformed before the analysis.
- *split.plots*: if the number of studied arrays is more than 50 it is adviced to define a number of experiments to represent on the density plots.
- *intgroup*: name of the column of the phenoData to be used to draw a color side bar next to the heatmap.

```
> fac = colnames(pData(AEset))[grep("Factor",colnames(pData(AEset)))]
> library("arrayQualityMetrics")
> arrayQualityMetrics(expressionset = AEset,
+                     outdir = "MEXP461NoNormalisation",
+                     force = FALSE,
+                     do.logtransform = TRUE,
+                     intgroup = fac[2])
```

A report named `QMreport.html` is produced in the subdirectory *MEXP461NoNormalisation*. It contains text illustrated by `.png` files. Each `.png` is linked to the corresponding `.pdf` files in order to provide high quality images.
Some arrays are highlighted as outliers. It is recommended to try different types of normalisation and see if these arrays are still outliers afterwards.

```
> rAEset = rma(AEset)
> arrayQualityMetrics(expressionset = rAEset,
+                     outdir = "MEXP461rmaNormalisation",
+                     force = FALSE,
+                     intgroup = fac[2])
```

As the rma normalisation summarises the probesets, the object `rAEset` is not an *AffyBatch* but an *ExpressionSet*. Thus, the plots that are specific to Affymetrix as RLE and NUSE are not computed.

# 3 Quality report for NChannelSet objects

## 3.1 Creating a NChannelSet

In this section, we use the *CCl4* example data set by Holger Laux, Timothy Wilkes, Amy Burrell and Carole Foy from LGC Ltd. in Teddington, UK. In the experiment, rat hepatocytes were treated either with carbon tetrachloride ($CCl_4$) or with DMSO. In the early $20^{th}$ century, $CCl_4$ was widely used as a dry cleaning solvent, as a refrigerant and in fire extinguishers, however, it was found to have multiple toxic and possible cancerogenous side-effects.

The DMSO treatment served as negative control. Total RNA was hybridized to Agilent *Rat Whole Genome* microarrays. The arrays use a two-color labeling scheme (Cy3 and Cy5), and the experiment was done as a direct comparison with dye-swaps and 3 replicates each. The integrity of the RNA was quantified from the electrophoretic trace of the RNA samples by Agilent's RNA Integrity Number (RIN). The initial samples had a RIN of 9.7. To study the effect of RNA degradation, additional samples were generated by degrading the $CCl_4$ treated RNA sample with ribonuclease A, resulting in RINs of 5.0 and 2.5. The experimental design is described in more detail below. As an example, we will create a *NChannelSet* with the CCl4 data. We first have to load the needed libraries.

```
> library("Biobase")
> library("limma")
> library("CCl4")
> library("matchprobes")
```

### 3.1.1 Read the data and convert them into an RGList

The Genepix (`.gpr`) data files are in the `extdata` directory of the CCl4 package. If you have the package installed, we can locate them on your filesystem with the function *system.file*. If the files are somewhere else, please adapt the following assignment to `datapath`.

```
> datapath = system.file("extdata", package="CCl4")
> dir(datapath)
```

```
 [1] "013162_D_SequenceList_20060815.txt" "251316214319_auto_479-628.gpr"
 [3] "251316214320_auto_478-629.gpr"      "251316214321_auto_410-592.gpr"
 [5] "251316214329_auto_429-673.gpr"      "251316214330_auto_457-658.gpr"
 [7] "251316214331_auto_431-588.gpr"      "251316214332_auto_492-625.gpr"
 [9] "251316214333_auto_487-712.gpr"      "251316214379_auto_443-617.gpr"
[11] "251316214380_auto_493-682.gpr"      "251316214381_auto_497-602.gpr"
[13] "251316214382_auto_481-674.gpr"      "251316214384_auto_450-642.gpr"
[15] "251316214389_auto_456-694.gpr"      "251316214390_auto_456-718.gpr"
[17] "251316214391_auto_475-599.gpr"      "251316214393_auto_460-575.gpr"
[19] "251316214394_auto_463-521.gpr"      "samplesInfo.txt"
```

```
> p = read.AnnotatedDataFrame("samplesInfo.txt", path=datapath)
> p
> CCl4_RGList = read.maimages(files=sampleNames(p),
+    path = datapath,
+    source = "genepix",
+    columns = list(R = 'F635 Median', Rb = 'B635 Median',
+                   G = 'F532 Median', Gb = 'B532 Median'))
```

The function *read.maimages* from the limma package reads the `.gpr` files and builds an *RGList* object from it.

### 3.1.2 Build an NChannelSet from the RGList

Once the *RGList* object has been created, we can build an *NChannelSet*.
The *assayData* have 4 different slots corresponding to the red and green intensities and the red and green background intensities. In addition, the *phenoData* contain the information about the samples and the *featureData* include the features information. You can fill these slots with all the specificities you want to store in your *NChannelSet*. In the case of the use of *arrayQualityMetrics*, the optimal *NChannelSet* include specific *featureData* that are described in the following section.

**X and Y coordinates of the spots** To plot the images of the arrays, *arrayQualityMetrics* needs the coordinates of the spots on the chip. Two slots corresponding to the row and column numbers of the features are thus required in the *featureData*. These slots should be named `X` and `Y`. If the *NChannelSet* does not contain these slots, the images of the arrays will not be produced in the report.

```
> CCl4_RGList$genes[95:105,]
```

```
    Block Row Column           ID            Name
95      1   1     95 A_44_P244495        AA819664
96      1   1     96 A_44_P138289    NM_001024787
97      1   1     97 A_44_P318805       XM_223584
98      1   1     98 A_44_P448307       XM_217432
99      1   1     99 A_44_P306486        AY387074
100     1   1    100 A_44_P559055        AA925039
101     1   1    101 A_44_P126261       XM_214443
102     1   1    102    (-)3xSLv1 NegativeControl
103     1   1    103 BrightCorner    BrightCorner
104     1   2      1 BrightCorner    BrightCorner
105     1   2      2    (-)3xSLv1 NegativeControl
```

```
> CCl4_RGList$genes$X = CCl4_RGList$genes$Row
> CCl4_RGList$genes$Y = CCl4_RGList$genes$Column
```

By using the function *read.maimages*, the slot `genes` of the produced *RGList* automatically contains these coordinates if the source is agilent, genepix or imagene or if the `annotation` argument is set.

**GC content of the reporters** If the GC content of the reporters is known, then it is possible to include it in the *featureData* of the *NChannelSet* under the column name `GC`. Then a study of the GC content effect on intensities of the arrays can be performed. This information is not included in the `CCl4_RGList` data yet. If the GC content or the sequence of the reporters are available in the source data files, we can include it by using the argument `other.columns` of *read.maimages*. As it is not the case in this example, we have to proceed

differently. The file with the sequences of the reporters is in the extdata directory of the package CCl4.

```
> seq = read.AnnotatedDataFrame("013162_D_SequenceList_20060815.txt",
+ path=datapath)
> if(any(duplicated(featureNames(seq))))
+   cat("IDs of the sequence file are not unique \n")
> bc = basecontent(seq$Sequence)
> GC = ((bc[,"C"]+bc[,"G"])/rowSums(bc))*100
> mt = match(featureNames(seq), CCl4_RGList$genes$ID)
> stopifnot(!any(is.na(mt)))
> fData = cbind(CCl4_RGList$genes,GC=rep(as.numeric("NA"),
+ nrow(CCl4_RGList$genes)))
> fData$GC[mt] = GC
```

**Mapping of the reporters** As a second part of the assessment of the platform quality, the report includes a study of the effect of the target mapping of the reporters. Thus a *featureData* slot named `HasTarget` should include logical TRUE if the reporter matches for a coding mRNA and FALSE if not. These information are not included in the `CCl4_RGList` data yet, but the slot `Name` of `CCl4_RGList$genes` give the RefSeq identifiers and we can use this to create the `HasTarget` slot.

```
> fData$hasTarget = (regexpr("^NM", CCl4_RGList$genes$Name) > 0)
```

**Building of the NChannelSet** Now that the *assayData* and *featureData* are ready, we can create the *NChannelSet*.

```
> featureData = new("AnnotatedDataFrame", data = fData)
> assayData = with(CCl4_RGList, assayDataNew(R=R, G=G, Rb=Rb, Gb=Gb))
> varMetadata(p)$channel=factor(c("G", "R", "G", "R"),
+                     levels=c(ls(assayData), "_ALL_"))
> CCl4 <- new("NChannelSet",
+               assayData = assayData,
+               featureData = featureData,
+               phenoData = p)
```

We can parse the information from the *phenoData* to have a column to use as a group of interest in the quality report.

```
> cond = paste(p$RIN.Cy3,p$RIN.Cy5,sep="/")
> poor = grep(cond,pattern="2.5")
> medium = grep(cond,pattern="^5/|/5")
> good = grep(cond,pattern="9.7")
```

```
> cov = rep(0, length = nrow(p))
> cov[good] = "Good"
> cov[medium] = "Medium"
> cov[poor] = "Poor"
> phenoData(CCl4)$RNAintegrity = cov
```

**Normalization** We can normalize the data using the variance stabilization method available in the package vsn.

```
> library("vsn")
> nCCl4 = justvsn(CCl4, subsample=2000)
```

## 3.2 Report production

```
> arrayQualityMetrics(expressionset = nCCl4,
+                     outdir = "CCl4",
+                     force = FALSE,
+                     intgroup = "RNAintegrity")
```

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

- R version 2.7.1 (2008-06-23), x86_64-unknown-linux-gnu
- Locale: C
- Base packages: base, datasets, grDevices, graphics, grid, methods, splines, stats, tools, utils
- Other packages: AnnotationDbi 1.2.2, Biobase 2.0.1, CCl4 1.0.7, DBI 0.2-4, RColorBrewer 1.0-2, RSQLite 0.6-9, affy 1.18.2, affyPLM 1.16.0, affyio 1.8.0, annotate 1.18.0, arrayQualityMetrics 1.6.0, beadarray 1.8.0, gcrma 2.12.1, genefilter 1.20.0, geneplotter 1.18.0, hgu95av2cdf 2.2.0, lattice 0.17-8, latticeExtra 0.4-1, limma 2.14.5, matchprobes 1.12.0, preprocessCore 1.2.0, simpleaffy 2.16.0, survival 2.34-1, vsn 3.6.0, xtable 1.5-2
- Loaded via a namespace (and not attached): KernSmooth 2.22-22