Preprocessing Affymetrix Exon ST and Gene ST Arrays

Benilton S. Carvalho

This document summarizes the RMA workflow for Affymetrix Gene ST and Affymetrix Exon ST arrays.

1 Loading the package

To use the tools provided by the oligo package, the user must load it.

```
R> library(oligo)
```

2 Importing data

To minimize the risks of running into issues with the location of the CEL files, we recommend the user to start by creating variables that contain the fully qualified CEL file names of interest.

As an example, suppose the CEL files corresponding to the Exon arrays are located in a directory called `~/exonExperiment`. Similarly, assume that the CEL files associated to the Gene ST array are located in `~/geneExperiment`. Then, using the `list.celfiles` function, the user can obtain the fully qualified names for the CEL files.

```
R> exonCELs <- list.celfiles("~/exonExperiment", full.names = TRUE)
R> geneCELs <- list.celfiles("~/geneExperiment", full.names = TRUE)
```

With the oligo package, the user is able to read CEL files directly through the `read.celfiles` function.

```
R> affyExonFS <- read.celfiles(exonCELs)
R> affyGeneFS <- read.celfiles(geneCELs)
```

NOTE: Readers who do not have their own data but are still interested in following this vignette can obtain the `affyExonFS` and `affyGeneFS` objects above through the oligoData package, as shown below:

```
R> library(oligoData)
R> data(affyExonFS)
R> affyExonFS
```

```
ExonFeatureSet (storageMode: lockedEnvironment)
assayData: 6553600 features, 20 samples
  element names: exprs
protocolData: none
phenoData
  rowNames: 10_5N.CEL 11_6T.CEL ... 9_5T.CEL (20 total)
  varLabels: exprs
  varMetadata: labelDescription channel
featureData: none
experimentData: use 'experimentData(object)'
Annotation: pd.huex.1.0.st.v2
```

```
R> data(affyGeneFS)
R> affyGeneFS
```

```
GeneFeatureSet (storageMode: lockedEnvironment)
assayData: 1102500 features, 33 samples
  element names: exprs
protocolData
  rowNames: TisMap_Brain_01_v1_WTGene1.CEL
    TisMap_Brain_02_v1_WTGene1.CEL ...
    TisMap_Thyroid_03_v1_WTGene1.CEL (33 total)
  varLabels: exprs dates
  varMetadata: labelDescription channel
phenoData
  rowNames: TisMap_Brain_01_v1_WTGene1.CEL
    TisMap_Brain_02_v1_WTGene1.CEL ...
    TisMap_Thyroid_03_v1_WTGene1.CEL (33 total)
  varLabels: index
  varMetadata: labelDescription channel
featureData: none
experimentData: use 'experimentData(object)'
Annotation: pd.hugene.1.0.st.v1
```

3 RMA - probeset level

Both designs allow the user to obtain summaries at the probeset level. To proceed with RMA at this level, the user should set the `target` argument to `'probeset'`.

```
R> exonPS <- rma(affyExonFS, target = "probeset")
Background correcting
Normalizing
Calculating Expression
R> genePS <- rma(affyGeneFS, target = "probeset")
Background correcting
Normalizing
Calculating Expression
```

4 RMA - transcript level

Summaries at the transcript level are also available. For Exon arrays, there are three possible options for transcript level summarization: **core**, **full** and **extended**. For Gene arrays, only summaries for **core** probesets is available.

```
R> exonCore <- rma(affyExonFS, target = "core")
Background correcting
Normalizing
Calculating Expression
R> exonFull <- rma(affyExonFS, target = "full")
Background correcting
Normalizing
Calculating Expression
R> exonExtd <- rma(affyExonFS, target = "extended")
Background correcting
Normalizing
Calculating Expression
R> geneCore <- rma(affyGeneFS, target = "core")
Background correcting
Normalizing
Calculating Expression
```

5 Retrieving NetAffx Biological Annotation

Biological annotation can be obtained with the `getNetAffx` function. It will return an *AnnotatedDataFrame* object, which can be used as `featureData` slot on the objects returned by *rma*. Users interested in further details about the fields available on this object should consult the Affymetrix documentation for annotation files.

```
R> featureData(exonPS) <- getNetAffx(exonPS, "probeset")
R> featureData(genePS) <- getNetAffx(genePS, "probeset")
R> featureData(exonCore) <- getNetAffx(exonCore, "transcript")
R> featureData(exonFull) <- getNetAffx(exonFull, "transcript")
R> featureData(exonExtd) <- getNetAffx(exonExtd, "transcript")
R> featureData(geneCore) <- getNetAffx(geneCore, "transcript")
```

The user can have an overview of the object by using its name:

```
R> exonCore
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 22011 features, 20 samples
  element names: exprs
protocolData: none
phenoData
  rowNames: 10_5N.CEL 11_6T.CEL ... 9_5T.CEL (20 total)
  varLabels: exprs
  varMetadata: labelDescription channel
featureData
  featureNames: 2315554 2315633 ... 7385696 (22011 total)
  fvarLabels: transcriptclusterid probesetid ... category (17
    total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation: pd.huex.1.0.st.v2
```

The biological annotation made available through NetAffx is now stored in the `featureData` slot.

```
R> featureData(exonCore)
```

```
An object of class "AnnotatedDataFrame"
  rowNames: 2315554 2315633 ... 7385696 (22011 total)
  varLabels: transcriptclusterid probesetid ... category (17
    total)
  varMetadata: labelDescription
```

```
R> varLabels(featureData(exonCore))
```

```
 [1] "transcriptclusterid" "probesetid"          "seqname"
 [4] "strand"              "start"               "stop"
 [7] "totalprobes"         "geneassignment"      "mrnaassignment"
[10] "swissprot"           "unigene"             "gobiologicalprocess"
[13] "gocellularcomponent" "gomolecularfunction" "pathway"
[16] "proteindomains"      "category"
```

As an example, the gene assignment for the 2 first metaprobesets (core) can be obtained as follows:

```
R> pData(featureData(exonCore))[1:2, "geneassignment"]
```

```
[1] "NM_001130045 // TTLL10 // tubulin tyrosine ligase-like family, member 10 // 1p36.33
[2] "NM_080605 // B3GALT6 // UDP-Gal:betaGal beta 1,3-galactosyltransferase polypeptide
```

The same strategy can be applied to any of the other objects.

6 Session Information

```
R> sessionInfo()
```

```
R version 2.12.0 Under development (unstable) (2010-05-26 r52110)
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

locale:
[1] en_GB.UTF-8/en_GB.UTF-8/C/C/en_GB.UTF-8/en_GB.UTF-8

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

other attached packages:
[1] pd.hugene.1.0.st.v1_3.0.2 pd.huex.1.0.st.v2_3.0.2
[3] RSQLite_0.9-2             DBI_0.2-5
[5] oligoData_1.0.0           oligo_1.13.8
[7] oligoClasses_1.11.10      Biobase_2.9.2

loaded via a namespace (and not attached):
[1] affxparser_1.21.1     affyio_1.17.4         Biostrings_2.17.48
[4] IRanges_1.7.39        preprocessCore_1.11.0 splines_2.12.0
[7] tools_2.12.0
```