

Analysis of Affymetrix microarray data

Benilton Carvalho

Introduction

I Affymetrix microarrays are a popular commercial platform available for a wide range of genomics applications (gene expression profiling, SNP genotyping, ChIP-chip analysis etc.) in different species.

The main distinction between Affymetrix and other array technologies is the use of many short (25mer) probes to measure hybridisation.

In this practical, we explore the basic analysis issues for Affymetrix GeneChips which measure gene expression using multiple (11-20) perfect match (PM) and mismatch (MM) probes concentrated in the 3' region of each transcript. Despite our focus on expression arrays, many of the processing steps, such as quality assessment and normalisation are equally important for other applications of Affymetrix technology to ensure that the signal is comparable between arrays.

In this practical we use several Bioconductor packages (`affy`, `affyPLM`, `limma`, etc.) to read in the raw data, assess data quality, normalise, summarise and measure differential expression in an experiment.

Estrogen data set

I The experiment we will analyse is from [1], and is made up of eight Affymetrix HGU95Av2 GeneChips. The aim of the experiment is briefly described below (excerpt taken from the `factDesign` package vignette).

“The investigators in this experiment were interested in the effect of estrogen on the genes in ER+ breast cancer cells over time. After serum starvation of all eight samples, they exposed four samples to estrogen, and then measured mRNA transcript abundance after 10 hours for two samples and 48 hours for the other two. They left the remaining four samples untreated, and measured mRNA transcript abundance at 10 hours for two samples, and 48 hours for the other two. Since there are two factors in this experiment (estrogen and time), each at two levels (present or absent, 10 hours or 48 hours), this experiment is said to have a 2×2 factorial design.”

The raw data is available in the `estrogen` package from

<http://www.bioconductor.org/help/bioc-views/release/data/experiment/>.

The cel files for this experiment are available in the `extdata` directory of the package. The targets file (`targLimma.txt`) is available with the course data.

Exercise 1: Confirm that there are two replicates of each condition. Read in the intensity data stored in the cel files. How many probes are there on each array?

```

> library(affy)
> pkgPath <- system.file('extdata', package='estrogen')
> tFile <- file.path(pkgPath, 'targLimma.txt')
> targets = read.AnnotatedDataFrame(tFile,header=TRUE,row.names=1,as.is=TRUE)
> pData(targets)
> setwd(pkgPath)
> rawdata = ReadAffy(filenamees=pData(targets)$FileName,phenoData=targets)
> rawdata

```

Diagnostic plots

I As with other microarray technologies, quality assessment is an important part of the analysis process. Data quality can be checked using various diagnostic plots.

Exercise 2: Generate image plots of the first two arrays. Do you observe any serious spatial artefacts which might warrant removal of either of these arrays from further analysis? Produce a boxplot of the intensities. What do you notice about the signal?

```

> image(rawdata[,1])
> image(rawdata[,2])
> boxplot(rawdata, col="red", las=2)

```

☞ It is clear from these plots that there is a systematic difference in signal between the 10 hour arrays and the 48 hour arrays. Most normalisation methods would remove this effect, as they assume that the signal for the majority of genes does not change between arrays. The experimenters felt that there was no possible biological reason why all genes would be more highly expressed at 48 hours than 10 hours, and that these differences must be an experimental artefact, which might be due to changes to the scanning settings.

I *MA* plots are a useful way of comparing the red and green channels from a two-colour microarray. For Affymetrix data, which is a single-channel technology, *M* and *A* values can be calculated using the intensities from a pair of chips, i.e. if X_i is the intensity of a given probe from chip i and X_j is the intensity for the same probe on chip j , then $A = \frac{1}{2}(\log_2 X_i + \log_2 X_j)$ and $M = \log_2 X_i - \log_2 X_j$. In this experiment, there are 8 GeneChips, which gives 28 distinct pair-wise comparisons between arrays. We will focus on a subset of these below.

Exercise 3: Make a pairs plot of the PM intensities from arrays 1-4 and 5-8 to compare the data from the replicate arrays. Based on all the plots you have generated, what would you conclude about the overall quality of this experiment? Would you use all the data in the downstream differential expression analysis?

```

> mva.pairs(pm(rawdata)[,1:4])
> mva.pairs(pm(rawdata)[,5:8])

```

Exercise 4: Generate histograms of the PM and MM intensities from the first array. Do you notice any difference in the signal distribution of the PMs and MMs?

```

> par(mfrow=c(2,1))
> hist(log2(pm(rawdata[,1])), breaks=100, col="blue", main="PM", xlim=c(4,14))
> hist(log2(mm(rawdata[,1])), breaks=100, col="red", main="MM", xlim=c(4,14))

```

☞ Most analysis algorithms do not use the MM probes, as they measure more than just non-specific background.

Probe-level Linear Models

📖 Probe-level Linear Models (PLMs) can be used as an additional tool to assess relative data quality within an experiment. Many different model specifications are possible, with the simplest fitting chip, and probe effects to the \log_2 intensities within each probeset across an experiment in a robust way.

The output is a matrix of residuals, or weights for each chip which can be used as an additional diagnostic; systematically high residuals across an entire array, or a large fraction of an array is indicative of an outlier array. The main use of this tool is in deciding whether or not to keep an array in the down-stream data analysis.

Exercise 5: Create image plots (used to diagnose spatial artefacts) for the first two chips and compare them with the image plots you obtained previously. Make a boxplot of the normalised unscaled standard errors (NUSE) and relative log-expression (RLE). Does any chip have systematically high standard errors?

```
> library(affyPLM)
> plmset = fitPLM(rawdata)
> plmset
> image(plmset,1)
> image(plmset,2)
> NUSE(plmset, las=2)
> ?NUSE
> RLE(plmset, las=2)
> ?RLE
```

📖 To see the results of PLM quality analyses on other data sets, see Ben Bolstad's affyPLM image gallery at <http://plmimagegallery.bmbolstad.com/>.

Normalisation and summarisation

📖 Many normalisation and summarisation methods have been developed for Affymetrix data. These include MAS5.0 and PLIER which were developed by Affymetrix and RMA, GC-RMA, dChip and vsn (to name but a few) which have been developed by academic researchers.

Many of these methods are available in the affy package. For a comparison of some of these methods and assessment of their performance on different control data sets, see [2] or [3]. In this practical we will use the RMA (Robust Multichip Averaging) method described in [4]. This method applies a model based background adjustment followed by quantile normalisation and a robust summary method (median polish) on the \log_2 PM intensities to obtain probeset summary values.

Exercise 6: Normalize the Estrogen data using RMA. Plot the normalised data and assess whether it has been effective by comparing it to the non-normalised data.

```

> eset = rma(rawdata)
> eset
> boxplot(data.frame(exprs(eset)), col="blue", ylim=c(4,16))
> mva.pairs(pm(rawdata)[,3:6])
> x11()
> mva.pairs(exprs(eset)[,3:6], log.it=FALSE)

```

☞ We need to specify `log.it=FALSE` in `mva.pairs` as the expression measurements in `eset` have already been log-transformed as part of the RMA algorithm. You can also use the function `MAplot` to get individual MA plots.

☞ Many other pre-processing, normalisation and summarisation options are available using the `expresso` function. Type the following to find out what they are. Refer to the help page for further details.

```

> bgcorrect.methods()
> normalize.methods(rawdata)
> pmcorrect.methods()
> express.summary.stat.methods()

```

☞ For other normalisation methods, such as GC-RMA, which takes probe composition into consideration, and variance stabilising normalisation (`vsn`), see the `gcrrma` and `vsn` packages respectively.

Differential expression analysis

❏ Once the probe-level data has been normalised and summarised, we are left with an absolute measure of expression for each probe-set on each array. To assess differential expression between the conditions of interest, we need to further summarise the expression values from replicate arrays, and make contrasts between the experimental conditions of interest.

In this experiment we have four pairs of replicate arrays, which allows us to estimate four parameters in the linear model. The design matrix should have four columns which correspond to *EstAbsent10*, *EstPresent10*, *EstAbsent48* and *EstPresent48*, and eight rows, one for each array.

Exercise 7: Set up a design matrix to estimate the 4 coefficients from this experiment. Identify the controls in your data set. Fit the linear model to estimate the coefficients for each non-control probeset.

```

> library(limma)
> design = cbind(EstAbsent10=c(1,1,0,0,0,0,0,0), EstPresent10=c(0,0,1,1,0,0,0,0), EstAbsent48=c(0,0,0,0,1,1,0,0), EstPresent48=c(0,0,0,0,0,0,1,1))
> design
> controls = grep("AFFX",featureNames(eset))
> fit = lmFit(eset[-controls,],design=design)

```

❏ The experimenters are interested in learning which genes respond to estrogen at 10 hours and 48 hours. They are also interested in finding out which genes change in expression level over time in the absence of estrogen.

Exercise 8: Set up a contrasts matrix and obtain contrasts of interest for this experiment. Use the `eBayes` function to moderate the t -statistics obtained from your analysis.

```
> contrastsMatrix = makeContrasts("EstPresent10-EstAbsent10", "EstPresent48-EstAbsent48", "EstAbsent48-EstPresent48")
> contrastsMatrix
> fit2 = contrasts.fit(fit, contrasts=contrastsMatrix)
> fit2 = eBayes(fit2)
```

Exercise 9: Use the `hgu95av2` annotation package to obtain the chromosome, unigene identifier and gene name for each probeset. Add this information to your results.

```
> library(hgu95av2.db)
> hgu95av2()
> ids = fit2$genes$ID
> chr = mget(ids, hgu95av2CHR, ifnotfound=NA)
> unigene = mget(ids, hgu95av2UNIGENE, ifnotfound=NA)
> genenames = mget(ids, hgu95av2GENENAME, ifnotfound=NA)
> anno = cbind(AffyID=ids, Chr=as.character(chr), Unigene=as.character(unigene), Name=as.character(genenames))
> fit2$genes=anno
```

Exercise 10: Obtain a list of the top 10 ranking genes from your analysis that respond to estrogen at 10 hours. Obtain a similar list for those that respond to estrogen at 48 hours. Write the results out to file.

```
> topTable(fit2, coef="EstPresent10-EstAbsent10")
> topTable(fit2, coef="EstPresent48-EstAbsent48")
> write.fit(fit2, file="results.txt")
```

I We can now examine which genes respond to Estrogen at either time (10 hours or 48 hours) using the `decideTests` function in `limma`.

Exercise 11: Check the help page for `decideTests` and look at the arguments it accepts. How does it classify which genes are differentially expressed across multiple contrasts by default? Using its default settings and the `vennCounts` function, determine how many genes are up-regulated at 10 hours, and how many of those genes were still up-regulated at 48 hours. Repeat this analysis for down-regulated genes. Plot a venn diagram of the results for all differentially expressed genes in the contrasts we have been looking at.

```
> ?decideTests
> results = decideTests(fit2)
> ?vennCounts
> vennCounts(results, include="up")
> vennCounts(results, include="down")
> vennDiagram(results, include=c("up", "down"), counts.col=c("red", "green"))
```

I The version of R and the packages used to complete this tutorial are listed below. If you have further questions about using any of the functions in this tutorial, consult the help pages, or email the Bioconductor mailing list (bioconductor@stat.math.ethz.ch).

```
> sessionInfo()
```

```

R version 2.15.0 (2012-03-30)
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

locale:
[1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/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] estrogen_1.8.8      hgu95av2.db_2.7.1    org.Hs.eg.db_2.7.1
[4] RSQLite_0.11.1      DBI_0.2-5            hgu95av2cdf_2.10.0
[7] AnnotationDbi_1.18.0 limma_3.12.0         affyPLM_1.32.0
[10] preprocessCore_1.18.0 gcrma_2.28.0         affy_1.34.0
[13] Biobase_2.16.0      BiocGenerics_0.2.0    BiocInstaller_1.4.4

loaded via a namespace (and not attached):
[1] affyio_1.24.0      Biostrings_2.24.1  IRanges_1.14.2
[4] splines_2.15.0     stats4_2.15.0      tools_2.15.0
[7] zlibbioc_1.2.0

```

Annotation of Probesets

 The assignment of probes to probesets by Affymetrix is based on the best available data at the time the arrays are first released. However, the evolution of genomic and transcriptomic sequencing and annotations means that these assignments are out-dated and hence suboptimal. In practice, significantly better results will be obtained by using probeset annotations based on more recent sequencing [5]. Such annotations, and instructions for integrating them with R, can be found at:
http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/genomic_curated_CDF.asp.

Acknowledgements

James Wettenhall whose earlier lab was used as the basis for this tutorial, and Robert Gentleman and Wolfgang Huber for the `estrogen` package. Matt Ritchie and Tom Hardcastle for the initial version of this practical.

References

- [1] Scholtens D, Miron A, Merchant FM, Miller A, Miron PL, Iglehart JD, and Gentleman R. (2004) *Analyzing Factorial Designed Microarray Experiments*. Journal of Multivariate Analysis. 90(1):19–43.

- [2] Bolstad, B. M., Irizarry, R. A., Astrand, M., and Speed, T. P. (2003). *A comparison of normalization methods for high density oligonucleotide array data based on variance and bias*. Bioinformatics, 19(2):185–193.
- [3] Millenaar, F.F., Okyere, J., May, S.T., van Zanten, M., Voesenek, L.A. and Peeters, A.J. (2006). *How to decide? Different methods of calculating gene expression from short oligonucleotide array data will give different results*. BMC Bioinformatics, 7:137.
- [4] Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). *Exploration, normalization, and summaries of high density oligonucleotide array probe level data*. Biostat, 4(2):249–264.
- [5] Dai, M., Wang, P., Boyd, D., Kostov, G., Athey, B., Jones, E., Bunney, W., Myers, R., Speed, T., Akil, H., Watson, S. and Meng F. (2005). *Evolving gene/transcript definitions significantly alter the interpretation of GeneChip data*. Nucleic Acids Res, 33:175.