

# Bioinformatics for microarray analysis

*Joseph Chao-Chung Kuo*

*Feb 18, 2017*

We will show here a typical pipeline for analysis of gene expression data. This tutorial is based on the use of Monocytes and Macrophage data from the following paper:

van de Laar L, Saelens W, De Prijck S, Martens L et al. Yolk Sac Macrophages, Fetal Liver, and Adult Monocytes Can Colonize an Empty Niche and Develop into Functional Tissue-Resident Macrophages. *Immunity* 2016 Apr 19;44(4):755-68. PMID: 26992565

This tutorial contains the following steps.

1. Download the data from GEO
2. Process the expression table for analysis
3. DE Analysis by Limma
4. PCA (Principal component analysis)
5. Clustering by Heatmap (only top 250 genes)
6. GO (Gene Ontology) Enrichment Analysis
7. Combining other dataset for PCA and clustering

Steps 1 to 3 corresponds to code automatically generated by GEO2R from Gene Expression Omnibus (GEO), as shown during the course.

You can find 3 files given to you under **handouts** directory:

File	Description
handout.pdf	Step-by-step explanation of the tasks.
handout_script.R	Scripts extracted out of the handout.
Imun_gen.ex.Rdump	Preprocessed object from GEO for loading in the analysis.
pheno.table	A table storing the phenotype data for comparison.

## Set the working directory

Before you start to work on this handout, please set your working directory to the place where you save the directory. For example,

```
setwd("/projects/geneexpression/course_r_geo/handouts")  
# Please use your path instead of the above example
```

## Loading the libraries

Before running the analysis, all the necessary libraries need to be loaded. We assume you have successfully installed all libraries as described here (<http://costalab.org/r-installation/>)

```
library(Biobase)  
library(GEOquery)  
library(limma)  
library(gplots)  
library(RColorBrewer)  
library(gProfileR)  
library(sva)
```

**Biobase** contains base functions for Bioconductor; **GEOquery** is the bridge between GEO and BioConductor; **LIMMA** is a library for the analysis of gene expression microarray data, especially the use of linear models for analysing designed experiments and the assessment of differential expression. **gplots** is for making graphical figures; **RColorBrewer** can make colorful palettes; **gProfileR** is the package to perform gene enrichment analysis; and **sva** (ComBat) is used here to remove known batch effects from microarray data.

## 1 Download the data from GEO

Firstly, we need to download the data from GEO by the GEO number.

```
gset <- getGEO("GSE76999", GSEMatrix = TRUE, AnnotGPL = TRUE)
if (length(gset) > 1) idx <- grep("GPL6246", attr(gset, "names")) else idx <- 1
gset <- gset[[idx]]
```

**ExpressionSet** (*gset* here) is a Container for high-throughput assays and experimental metadata.

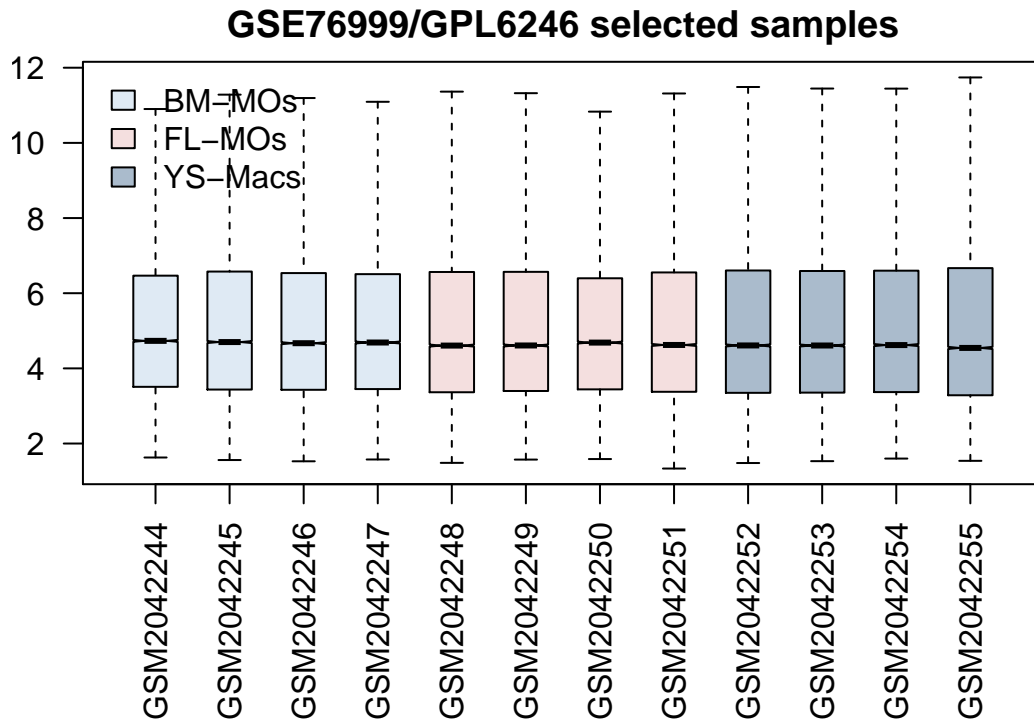
## 2 Process the expression table for analysis

Here we select only the first 12 samples for downstream analysis and eliminate others. These 12 samples are **BM-MOs** (bone marrow monocytes), **FL-MOs** (fetal liver monocytes), and **YS-Macs** (yolk sac-derived macrophages), with 4 replicates for each.

```
# make proper column names to match toptable
fvarLabels(gset) <- make.names(fvarLabels(gset))
# group names for all samples
gsms <- "000011112222XXXXXXXXXXXXXXXXXXXXX"
sml <- c()
for (i in 1:nchar(gsms)) { sml[i] <- substr(gsms,i,i) }
# eliminate samples marked as "X"
sel <- which(sml != "X")
sml <- sml[sel]
gset <- gset[,sel]
fl <- as.factor(sml)
labels <- c("BM-MOs", "FL-MOs", "YS-Macs")
```

Before we do any further analysis, we need to do quality control of the data. We use boxplot to check whether we need further normalization on the data.

```
palette(c("#dfeaf4", "#f4dfff", "#AABBCC"))
par(mar=c(2+round(max(nchar(sampleNames(gset)))/2), 4, 2, 1))
title <- paste ("GSE76999", '/', annotation(gset), " selected samples", sep='')
boxplot(exprs(gset), boxwex=0.6, notch=T, main=title, outline=FALSE, las=2, col=fl)
legend("topleft", labels, fill=palette(), bty="n")
```



The boxplot indicates all samples have the same distribution, therefore there is no need for any normalization.

### 3 Differential Expression Analysis by Limma

Next, we want to find genes, which are differentially expressed between BM-MOs, FL-MOs and YS-Macs using **Limma**. First, we perform a log transformation of the data and next we create a design matrix defining the groups to compare. Here we show you how to get the top 250 DE genes.

```
# transform into log2 scale
ex <- exprs(gset)
exprs(gset) <- log2(ex)
# set up the data and proceed with analysis
sml <- paste("G", sml, sep="") # set group names
fl <- as.factor(sml)
gset$description <- fl
design <- model.matrix(~ description + 0, gset)
colnames(design) <- levels(fl)
fit <- lmFit(gset, design)
cont.matrix <- makeContrasts(G2-G0, G1-G0, G2-G1, levels=design)
fit2 <- contrasts.fit(fit, cont.matrix)
fit2 <- eBayes(fit2, 0.05)
tT <- topTable(fit2, adjust="fdr", sort.by="B", number=250)
```

Above script is from GEO2R and only gets top 250 DE genes. In order to get all significant DE genes, we use the script below:

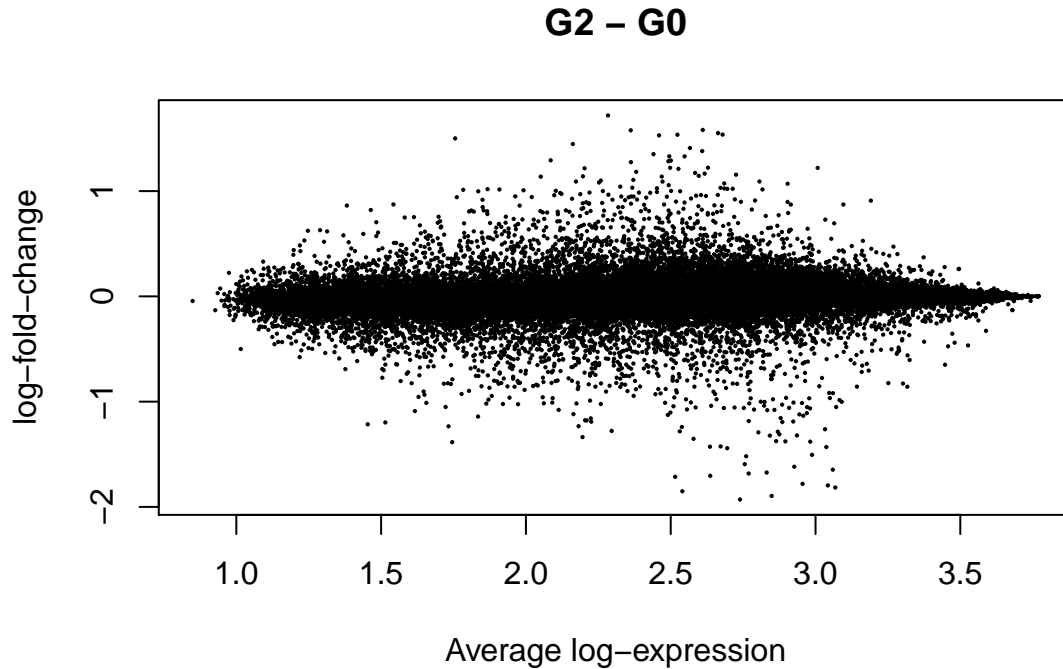
```
write.table(tT, file="de_genes.csv", row.names=F, sep="\t")
tT <- topTable(fit2, adjust="fdr", sort.by="B", p.value=0.05, number=Inf)
```

Please open the output table *de\_genes.csv* and try to understand each column.

### MA plot

The MA plot visualises the differences between measurements taken in two samples, by transforming the data onto M (log ratio) and A (mean average) scales, then plotting these values. Here we take **BM-MOs** (G0) v.s. **YS-Macs** (G2) as an example.

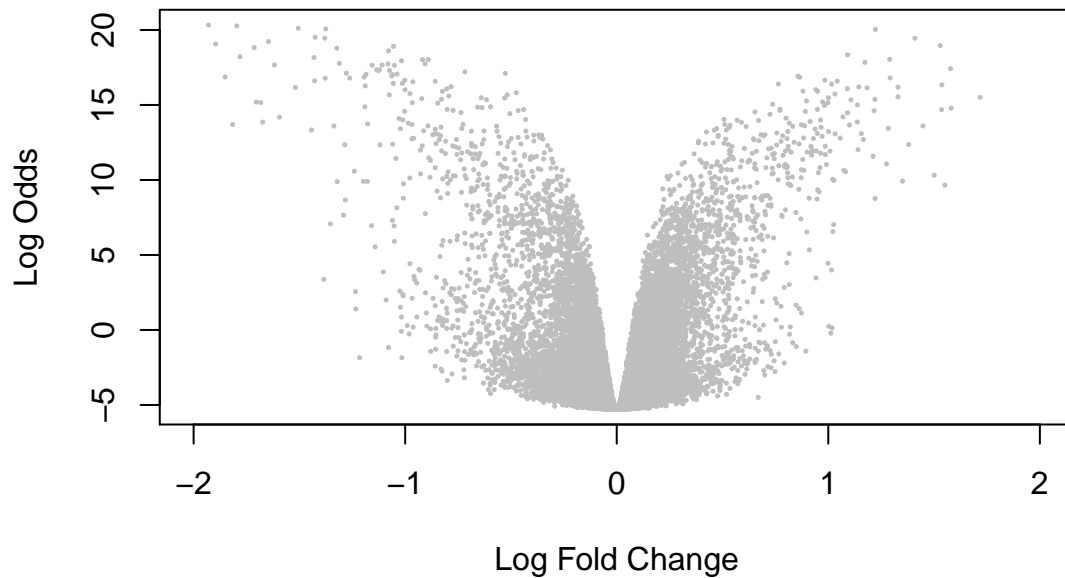
```
ma <- fit2[, "G2 - G0"]  
plotMA(ma)
```



### Volcano plot

A volcano plot is a type of scatter-plot that is used to quickly identify changes in large data sets. It plots significance versus fold-change on the y and x axes, respectively.

```
colnames(fit2)  
# pdf("volcanoplot.pdf", width=4, height=4) # Save the plot into PDF format  
volcanoplot(fit2, coeff=1, xlim=c(-2,2), col="gray")
```



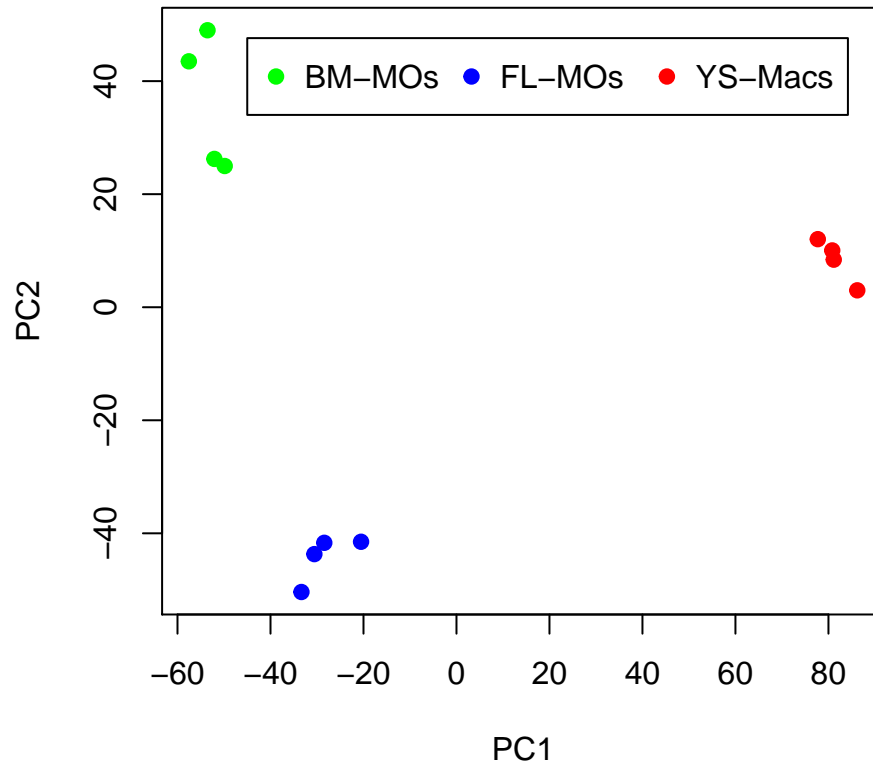
```
# dev.off() # close the graphical device
```

```
## [1] "G2 - G0" "G1 - G0" "G2 - G1"
```

## 4 PCA (Principal component analysis)

Principal component analysis (PCA) is a mathematical algorithm that reduces the dimensionality of the data while retaining most of the variation in the data set<sup>1</sup>. It accomplishes this reduction by identifying directions, called principal components, along which the variation in the data is maximal. By using a few components, each sample can be represented by relatively few numbers instead of by values for thousands of variables. Samples can then be plotted, making it possible to visually assess similarities and differences between samples and determine whether samples can be grouped (see Nature Biotechnology 26, 303 - 304 (2008)).

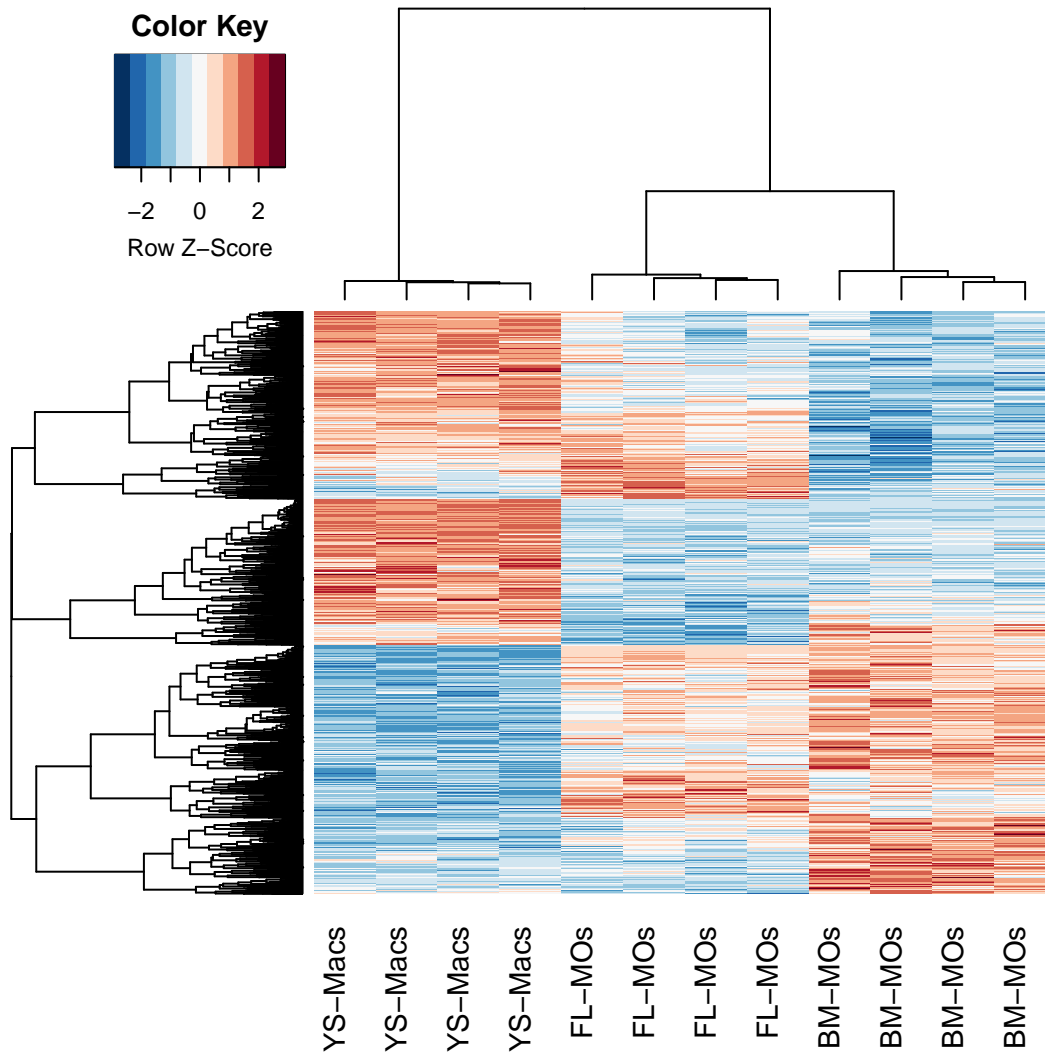
```
palette(c("Green", "Blue", "Red"))
pca=prcomp(t(ex))
plot(pca$x[,1:2], col=f1, pch=19)
legend("topright", inset=.05, labels, pch=19, col=1:3, horiz=TRUE)
```



## 5 Clustering by Heatmap (only top 250 genes)

Another way to visualize the similarity and the difference of the samples is to perform hierarchical clustering and plot results as a heatmap.

```
hmcol=rev(brewer.pal(50,"RdBu"))
hp <- heatmap.2(ex[match(tT$ID, rownames(ex)),], dendrogram = "both", col=hmcol, scale="row",
  labRow=F, density.info="none", trace="none", margins = c(8,3),
  labCol=rep(labels,each=4),
  hclust=function(x) hclust(x,method="complete"),
  distfun=function(x) as.dist((1-cor(t(x)))/2))
```



### Get genes from clustering

If we want to get the clusters from the dendrogram on the left hand side, we can use the script below:

```
clusters <- cutree(as.hclust(hp$rowDendrogram),k=5)
clustered_genes <- list(tT$Gene.symbol[match(names(clusters)[clusters==1],tT$ID)],
                        tT$Gene.symbol[match(names(clusters)[clusters==2],tT$ID)],
                        tT$Gene.symbol[match(names(clusters)[clusters==3],tT$ID)],
                        tT$Gene.symbol[match(names(clusters)[clusters==4],tT$ID)],
                        tT$Gene.symbol[match(names(clusters)[clusters==5],tT$ID)])
write.table(clustered_genes[[1]],file="cluster1_genes.txt",quote=F,row.names=F,col.names=F)
write.table(clustered_genes[[2]],file="cluster2_genes.txt",quote=F,row.names=F,col.names=F)
write.table(clustered_genes[[3]],file="cluster3_genes.txt",quote=F,row.names=F,col.names=F)
write.table(clustered_genes[[4]],file="cluster4_genes.txt",quote=F,row.names=F,col.names=F)
write.table(clustered_genes[[5]],file="cluster5_genes.txt",quote=F,row.names=F,col.names=F)
```

## 6 GO (Gene Ontology) Enrichment Analysis

GO analysis requires a list of genes to find which GO terms are over-represented. Here we take **BM-MOs** (G0) v.s. **YS-Macs** (G2) as an example.

```
go_enrich_up <- gprofiler(as.vector(tT$Gene.symbol[tT$G2...G0 > 0]),
                          organism="mmusculus")
go_enrich_down <- gprofiler(as.vector(tT$Gene.symbol[tT$G2...G0 < 0]),
                            organism="mmusculus")
write.table(go_enrich_up, "GO_BM-MOs_YS-Macs_up_genes.csv")
write.table(go_enrich_down, "GO_BM-MOs_YS-Macs_down_genes.csv")
```

## 7 Combining other dataset for PCA and clustering

Finally, as an advanced example of data combined, we have selected Monocyte and Macrophage populations from the ImmGenn data. See the script XXX to see how this was done.

For simplicity, we will load the pre-processed data and a table containing details on the experiments (pheno data). To make these data comparable, we need to remove the batch effects caused by different labs, by using a tool called combat.

```
# get imm gene data from saved object
ex_imun_gen <- dget("Imun_gen.ex.Rdump")
# get all pheno data from tab sep. file
phenoData_table <- read.table(file="pheno.table.csv", header=T, sep=",",
                              quote="")

ex_all <- merge(ex_imun_gen, ex, by="row.names")
rownames(ex_all) <- ex_all$"Row.names"
ex_all$"Row.names" <- NULL
ex_all <- as.matrix(ex_all)

modcombat = model.matrix(~1, data=phenoData_table)
combat_edata = ComBat(dat=ex_all, batch=phenoData_table$batch,
                      mod=modcombat, par.prior=TRUE)
```

### PCA on merged data

Once the data has been combined, we can perform PCA and clustering analysis in both data sets.

```
pdf("PCA_int.pdf", width=4, height=4)
pca_im=prcomp(t(ex_all))
palette(rainbow(length(levels(phenoData_table[,1]))))
plot(pca_im$x[,1:2], pch=19, col=phenoData_table[,1])
text(pca_im$x[,1]+1, pca_im$x[,2]+3, phenoData_table[,2], cex=0.5)
```

### Clustering of merged data

```
hmcol=rev(brewer.pal(50, "RdBu"))
sel <- match(tT$ID, rownames(combat_edata))
sel <- sel[!is.na(sel)]
heatmap.2(combat_edata[sel,], dendrogram = "both", col=hmcol, scale="row",
```



```
labRow=F, density.info="none", trace="none", margins = c(9,3),
labCol=phenoData_table[,2])
```

