Jack Zhan

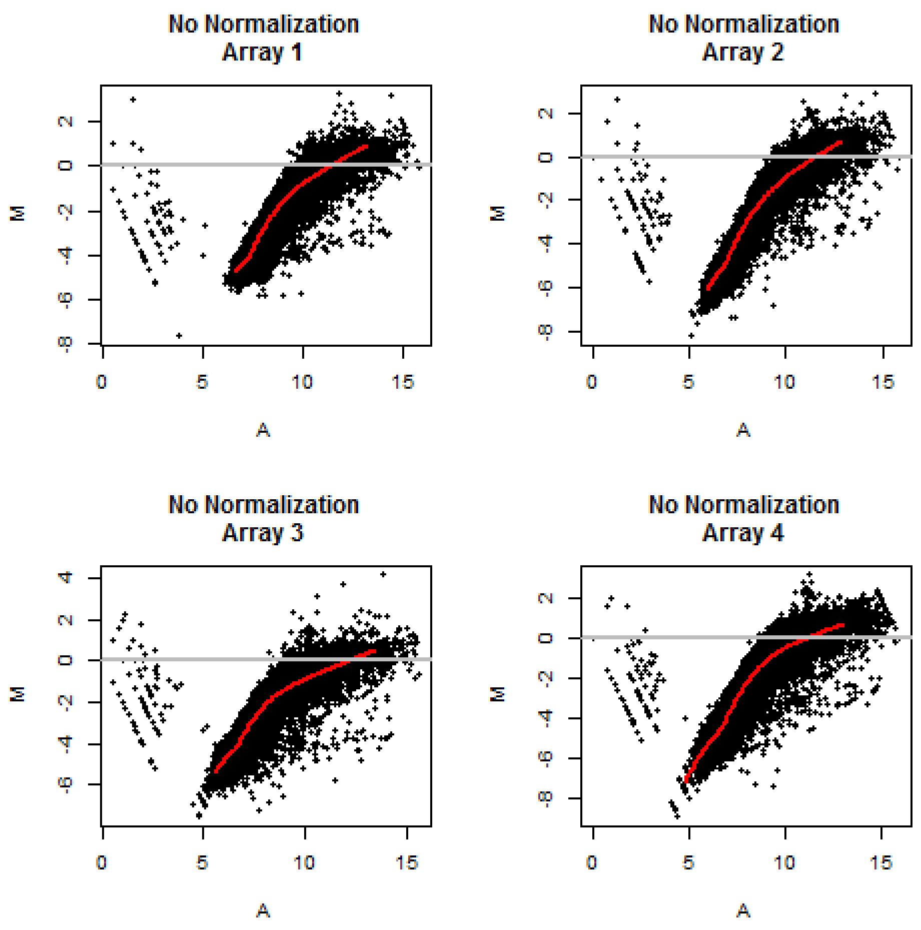
Homework 2

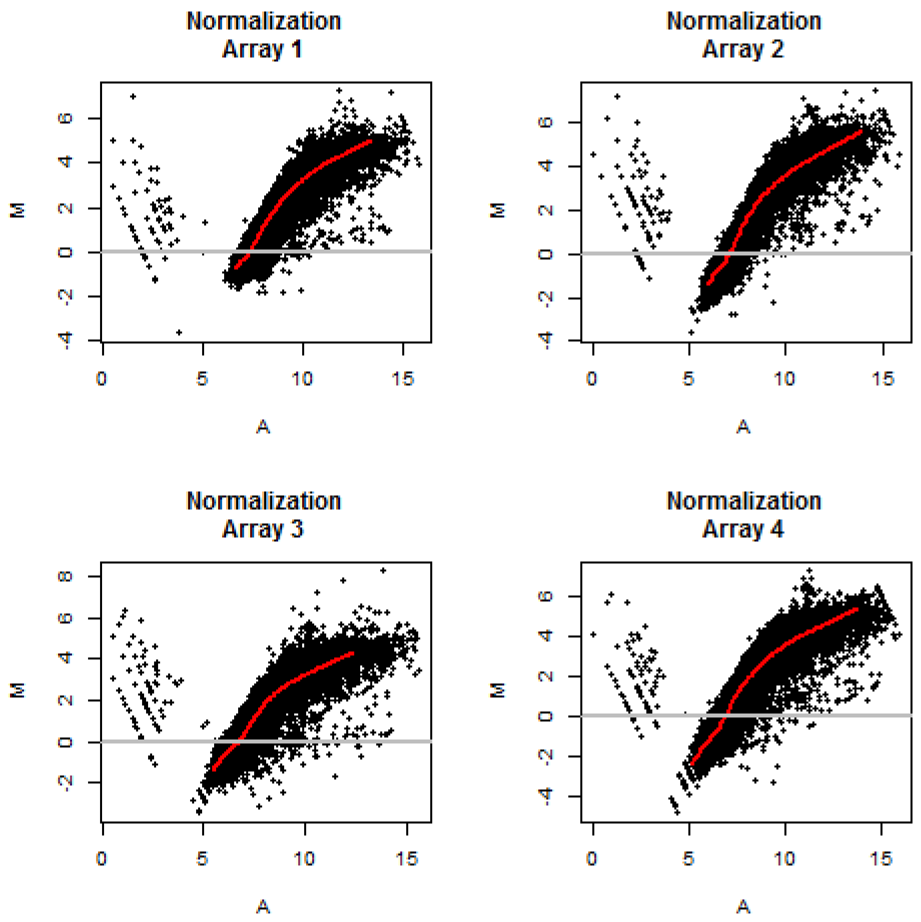
1.

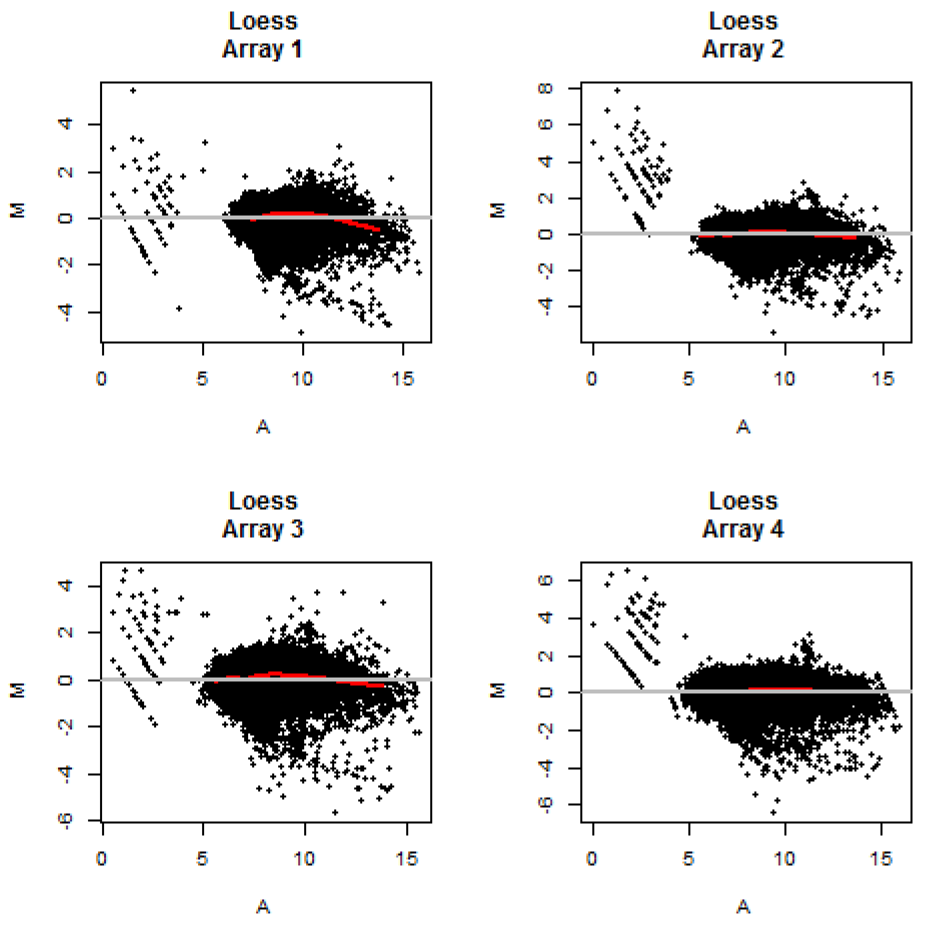
```
#Homework 2
#Author: Jack Zhan

#This is the code for Homework 2

#Loading the library marray
library(marray)
#Setting the working directory
setwd("C:\\Users\\Jack\\Desktop\\R_Lab\\HW2")
file_location <- "C:\\Users\\Jack\\Desktop\\R_Lab\\HW2"

#loading the genepix_files
genepix_files <- read.GenePix(path=file_location)
```

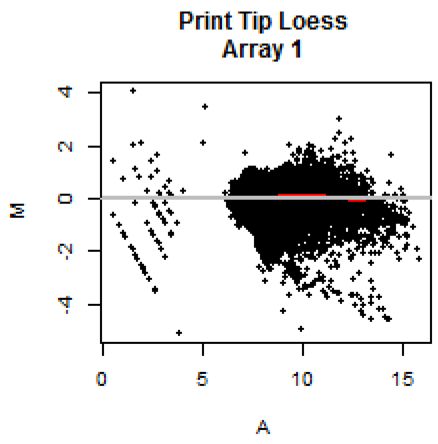
2.

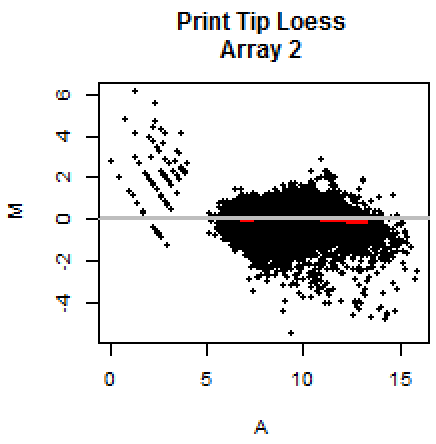
```
#Normalize each array using median global, loess, and print-tip-group loess methods
genepix_normal <- maNorm(genepix_files,norm="median")
genepix_loess <- maNorm(genepix_files,norm="loess")
genepix_tip <- maNorm(genepix_files,norm="printTipLoess")
genepix_none <- maNorm(genepix_files,norm="none")


# 4 figures arranged in 2 rows and 2 columns
par(mfrow=c(2,2))
#Plot each of the 4 No Normalization
maPlot(genepix_none[,1],main='No Normalization\nArray 1',legend.func=NULL)
maPlot(genepix_none[,2],main='No Normalization\nArray 2',legend.func=NULL)
maPlot(genepix_none[,3],main='No Normalization\nArray 3',legend.func=NULL)
maPlot(genepix_none[,4],main='No Normalization\nArray 4',legend.func=NULL)
dev.copy(png,'genepix_none.png')
dev.off()

# 4 figures arranged in 2 rows and 2 columns
par(mfrow=c(2,2))
#Plot each of the 4 Normalization
maPlot(genepix_normal[,1],main='Normalization\nArray 1',legend.func=NULL)
maPlot(genepix_normal[,2],main='Normalization\nArray 2',legend.func=NULL)
maPlot(genepix_normal[,3],main='Normalization\nArray 3',legend.func=NULL)
maPlot(genepix_normal[,4],main='Normalization\nArray 4',legend.func=NULL)
dev.copy(png,'genepix_normal.png')
dev.off()

# 4 figures arranged in 2 rows and 2 columns
par(mfrow=c(2,2))
#Plot each of the 4 Loess
maPlot(genepix_loess[,1],main='Loess\nArray 1',legend.func=NULL)
maPlot(genepix_loess[,2],main='Loess\nArray 2',legend.func=NULL)
maPlot(genepix_loess[,3],main='Loess\nArray 3',legend.func=NULL)
maPlot(genepix_loess[,4],main='Loess\nArray 4',legend.func=NULL)
dev.copy(png,'genepix_loess.png')
dev.off()

# 4 figures arranged in 2 rows and 2 columns
par(mfrow=c(2,2))
#Plot each of the 4 Print Tip Loess
maPlot(genepix_tip[,1],main='Print Tip Loess\nArray 1',legend.func=NULL)
maPlot(genepix_tip[,2],main='Print Tip Loess\nArray 2',legend.func=NULL)
maPlot(genepix_tip[,3],main='Print Tip Loess\nArray 3',legend.func=NULL)
maPlot(genepix_tip[,4],main='Print Tip Loess\nArray 4',legend.func=NULL)
dev.copy(png,'genepix_tip.png')
dev.off()
```

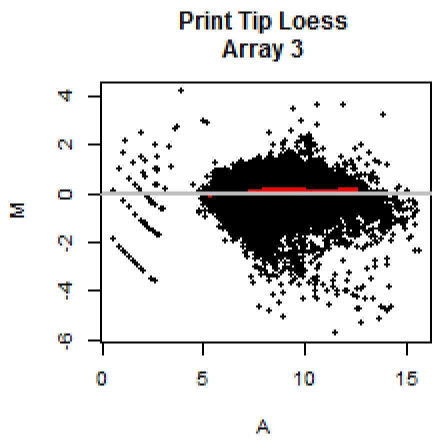


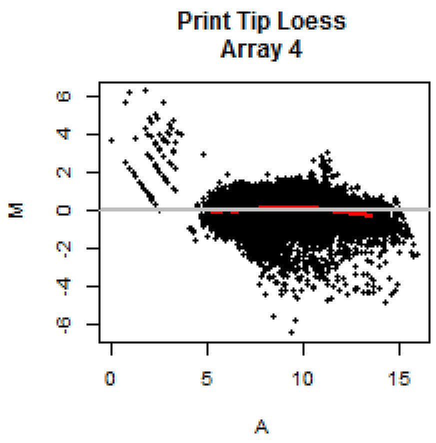

Normalization
Array 1
Normalization
Array 2
Normalization
Array 3
Normalization
Array 4
M
A
-4
-2
0
2
4
6
8
0
5
10
15


Loess
Array 1
Loess
Array 2
Loess
Array 3
Loess
Array 4
M
A



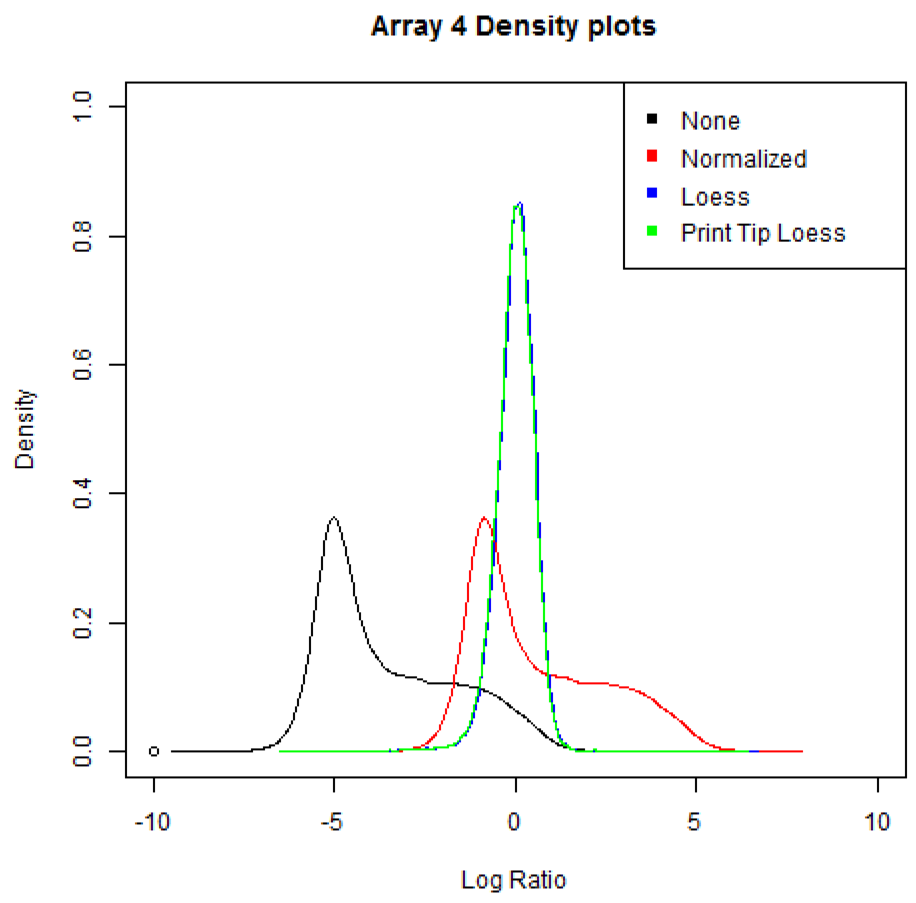






3.

```
#Density plot with log ratio values
#Remove all missing values for calculation
density_none <-density(maM(genepix_none)[,4],na.rm=T)
density_normal <-density(maM(genepix_normal)[,4],na.rm=T)
density_loess <-density(maM(genepix_loess)[,4],na.rm=T)
density_tip <-density(maM(genepix_tip)[,4],na.rm=T)

plot(c(-10,10),c(0,1),main="Array 4 Density plots",xlab="Log Ratio",ylab="Density")
lines(density_none, lwd=1,col="black")
lines(density_normal, lwd=1, col="red")
lines(density_loess, lwd=1,col="blue")
lines(density_tip, lwd=1,col="green")
legend("topright",c("None","Normalized","Loess","Print Tip Loess"),pch=15,col=c("black","red","blue","green"))
dev.copy(png,'density_plot.png')
dev.off()
```



4. I think Loess is the best. Print Tip Loess is basically the same as Loess based on the graphs produced. They both have a log ratio close to 0 and a cluster that most resembles a circle around 0.

5.

```
#Extracting the Cy5 foreground and background values
foreground <- maRf(genepix_files)
background <-maRb(genepix_files)
#subtract the background from the foreground values, then log2 transform
cy5 <- log2(foreground-background)
print (cy5)
#Calculating global median normalization
cy5_normalized <-cy5
for (i in 1:ncol(cy5)) {
  cy5_normalized[,i] = cy5_normalized[,i]/median(cy5_normalized[,i],na.rm = T)
}
```

6.

```
# Spearman Correlation calculation on cy5_normalized
cy5.cor <- cor(cy5_normalized,method="spear",use="pairwise.complete.obs")
rownames(cy5.cor) <- c()
colnames(cy5.cor) <- c()
print(cy5.cor)
```

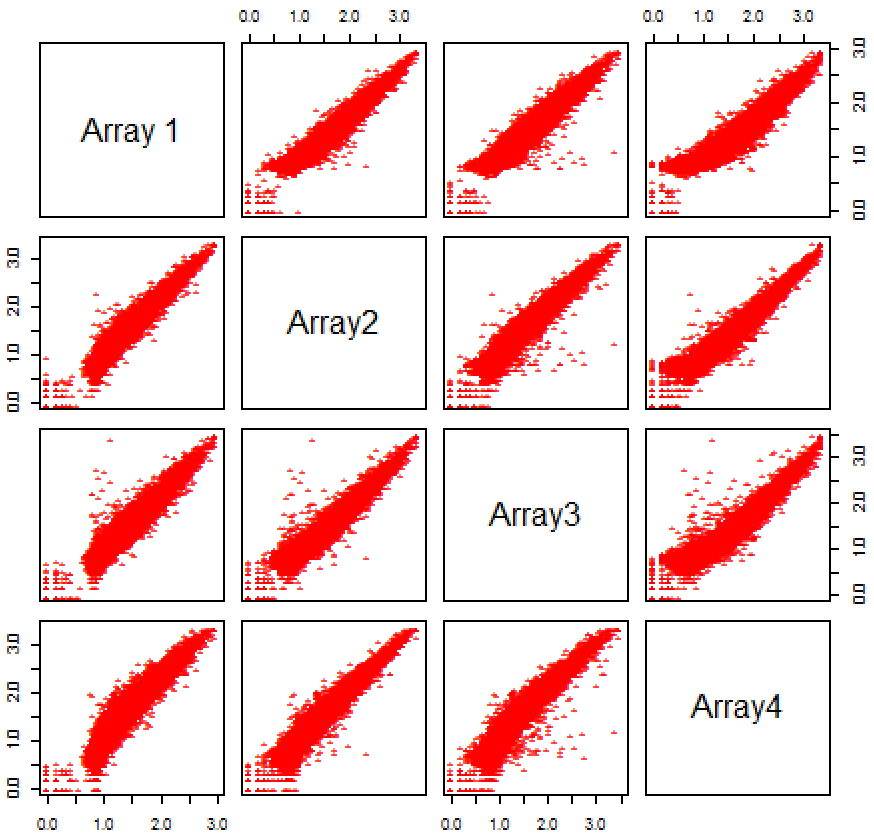
```
          [,1]      [,2]      [,3]      [,4]
[1,] 1.0000000 0.8962752 0.8790831 0.8990696
[2,] 0.8962752 1.0000000 0.8766078 0.9080043
[3,] 0.8790831 0.8766078 1.0000000 0.8854758
[4,] 0.8990696 0.9080043 0.8854758 1.0000000
```

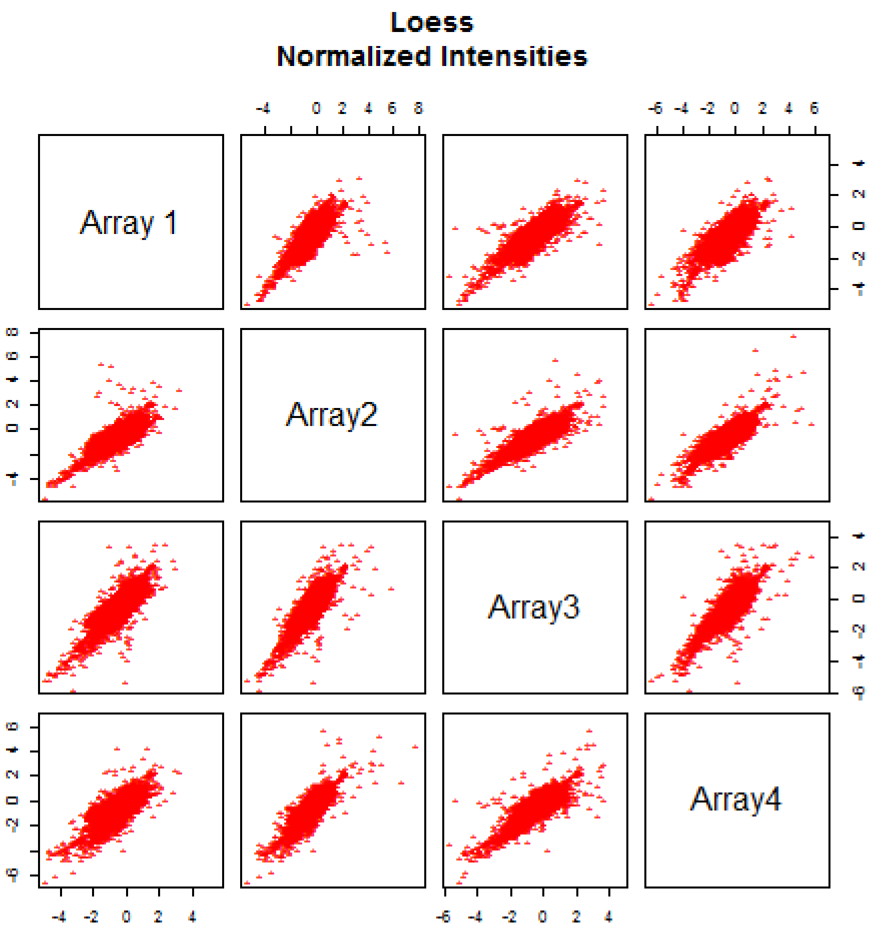
```
# Spearman Correlation calculation on loess data
loess_normalized <- maM(genepix_loess)
loess.cor <- cor(loess_normalized,method="spear",use="pairwise.complete.obs")
rownames(loess.cor) <- c()
colnames(loess.cor) <- c()
print(loess.cor)
```

```
          [,1]      [,2]      [,3]      [,4]
[1,] 1.0000000 0.6882265 0.7587292 0.6916965
[2,] 0.6882265 1.0000000 0.7217613 0.7061138
[3,] 0.7587292 0.7217613 1.0000000 0.7467916
[4,] 0.6916965 0.7061138 0.7467916 1.0000000
```

```
#Create matrix plot using pairs function
pairs(cy5_normalized,labels=c("Array 1","Array2","Array3","Array4"),main="Cy5 background Subtract\nNormalized
Intensities",pch="*",col="red")
dev.copy(png,'matrix_cy5.png')
dev.off()

pairs(loess_normalized,labels=c("Array 1","Array2","Array3","Array4"),main="LnNormalized Intensities",pch="*",col="red")
dev.copy(png,'matrix_loess.png')
dev.off()
```

Cy5 background Subtract Normalized Intensities





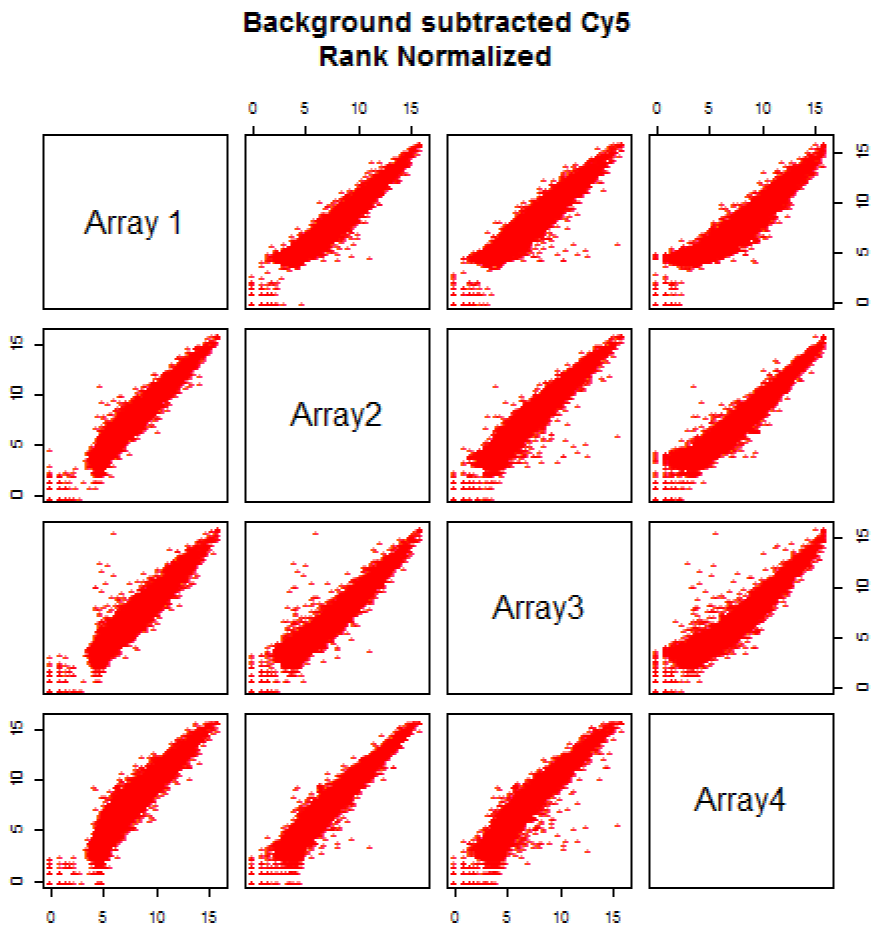
7.

```
#Get unlogged version of background subtract
cy5_unlog <- foreground-background
#put values in matrix
cy5_matrix <- matrix(cy5_unlog,nrow=nrow(cy5_unlog),ncol=ncol(cy5_unlog))
print(cy5_matrix)
#sort the columns
cy5_sort <- apply(cy5_matrix,2,sort)
#Get mean on sorted data
print(cy5_sort)
cy5_mean <- apply(cy5_sort,1,mean)
print(cy5_mean)
#Apply row mean to a new matrix
cy5_mean_matrix <- matrix(cy5_mean,nrow=nrow(cy5_unlog),ncol=ncol(cy5_unlog))
print(cy5_mean_matrix)
#Rank the inital matrix with the argument ties="first"
cy5_rank <- apply(cy5_matrix,2,rank,ties="first")

#Create empty matrix
cy5_final<-matrix(, nrow=nrow(cy5_unlog),ncol=ncol(cy5_unlog))
#Rank the columns independently on the original background subtracted matrix
for(i in 1:ncol(cy5_rank)) {
 cy5_final[,i] = cy5_sort[cy5_rank[,i],i]
}
```

```
#verifying the rankings with histograms of each column
par(mfrow=c(2,2))
hist(cy5_final[,1])
hist(cy5_final[,2])
hist(cy5_final[,3])
hist(cy5_final[,4])
dev.off()
```

8.

```
#Spearman Correlation calculation on cy5_final
log_cy5 <- log2(cy5_final)
log_cy5.cor <- cor(log_cy5,method="spear",use="pairwise.complete.obs")
print(log_cy5.cor)
          [,1]      [,2]      [,3]      [,4]
[1,] 1.0000000 0.8962752 0.8790831 0.8990696
[2,] 0.8962752 1.0000000 0.8766078 0.9080043
[3,] 0.8790831 0.8766078 1.0000000 0.8854758
[4,] 0.8990696 0.9080043 0.8854758 1.0000000
airs(log_cy5,labels=c("Array 1","Array2","Array3","Array4"),main="Background subtracted Cy5\nRank Normalized",pch="*",col="red")
dev.copy(png,'matrix_rank_cy5.png')
dev.off()
```



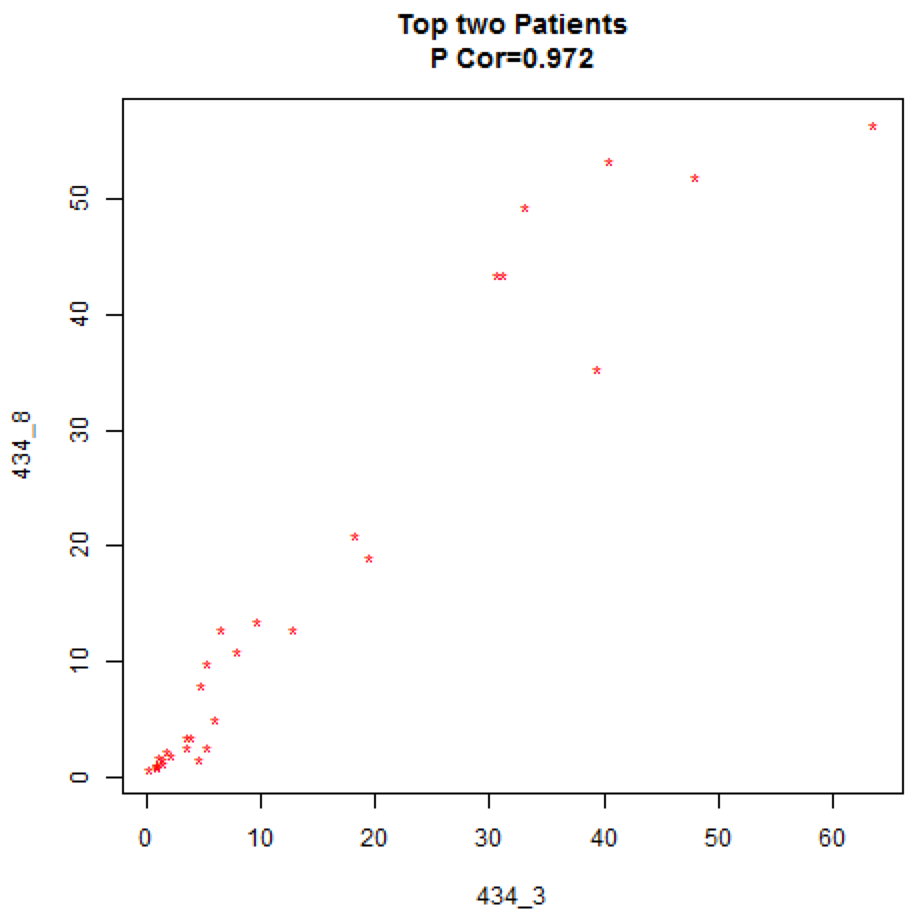
9. I would go with background select rank normalized because the image in the matrix plots are most consistent throughout the arrays. We also did a lot of work to create the plot as well.

10.

```
f.parse <- function(path=pa,file=fi,out=out.fi) {
  d <- read.table(paste(path,file,sep=""),skip=11,sep=",",header=T)
  u <- as.character(unique(d$Name))
  u <- u[u!=""]; u <- u[!is.na(u)];
  ref <- unique(as.character(d$Name[d$Type=="Reference"]))
  u <- unique(c(ref,u))
  hg <- c("B-actin","GAPH","18S")
  hg <- toupper(hg)
  p <- unique(toupper(as.character(d$Name.1)))
  p <- sort(setdiff(p,c("",hg)))

  mat <- matrix(0,nrow=length(u),ncol=length(p))
  dimnames(mat) <- list(u,p)
  for (i in 1:length(u)) {
    print(paste(i,": ",u[i],sep=""))
    tmp <- d[d$Name %in% u[i],c(1:3,6,9)]
    g <- toupper(unique(as.character(tmp$Name.1)))
    g <- sort(setdiff(g,c("",hg)))

    for (j in 1:length(g)) {
      v <- tmp[toupper(as.character(tmp$Name.1)) %in% g[j],5]
      v <- v[v!=999]
      v <- v[((v/mean(v))<1.5) & ((v/mean(v))>0.67)]          #gene j vector

      hv3 <- NULL
      for (k in 1:length(hg)) {      #housekeeping gene vector (each filtered by reps)
        hv <- tmp[toupper(as.character(tmp$Name.1)) %in% hg[k],5]
        hv <- hv[hv!=999]
        hv3 <- c(hv3,hv[((hv/mean(hv))<1.5) & ((hv/mean(hv))>0.67)])
      }

      sv <- mean(as.numeric(v)) - mean(as.numeric(hv3))  #scaled value for gene j

      if(i==1) { #reference sample only
        mat[u[i],g[j]] <- sv
        next
      }

      mat[u[i],g[j]] <- sv - mat[u[1],g[j]]
    }
  }

  mat[1,][!is.na(mat[1,])] <- 0
  fc <- 2^(-1 * mat)
  write.table(t(c("Subject",dimnames(mat)[[2]])),paste(path,out,sep=""),quote=F,sep="\t",col.names=F,row.names=F)
  write.table(round(fc,3),paste(path,out,sep=""),quote=F,sep="\t",append=T,col.names=F)
}
f.parse(file_location,"\\Inflammation_qRT-PCR.csv","\\output.txt")
```

11.

```
#Convert to matrix
pcr_matrix <- t(as.matrix(qrt_pcr))
pcr_matrix = pcr_matrix[,-14]
print(pcr_matrix)
qrt_pcr.cor <- cor(pcr_matrix,method="spear",use="pairwise.complete.obs")
print(qrt_pcr.cor)
matrix_cor_qrt <- as.matrix(qrt_pcr.cor)
print(matrix_cor_qrt)
matrix_cor_qrt = matrix_cor_qrt
#Remove unwanted data
matrix_cor_qrt[is.na(matrix_cor_qrt)] <- 0
matrix_cor_qrt[matrix_cor_qrt > .9999999] <- 0
#Get max value
print(max(matrix_cor_qrt))
#Max value of 0.9724138 for 434_3 and 434_8
#grab the  434_3 and 434_8
p_434_3 <-as.numeric(qrt_pcr["434_3",])
print(p_434_3)
p_434_8 <-as.numeric(qrt_pcr["434_8",])
print(p_434_8)
#Plot graph
plot(p_434_3,p_434_8,xlab="434_3",ylab="434_8",main="Top two Patients\nP Cor=0.972",pch="*",col="red")
dev.copy(png,'qrt_plot.png')
dev.off()
```


Top two Patients
P Cor=0.972
434_8
434_3