ANÁLISIS DE DATOS DE MICROARRAYS DE DOS CANALES

UN ACERCAMIENTO TEÓRICO - PRÁCTICO

Ricardo Bello
Mauro Colombini
Eugenia Takeda

Tabla de Contenidos

¿Qué es un microarray?

Es un soporte sólido, generalmente de vidrio o silicio, al que se le han adherido, mediante un robot, en forma ordenada sondas (***probes***) con diferentes cadenas conocidas de material genético (DNA, cDNA, oligos) (cubriendo parte o toda la secuencia de un genoma-transcriptoma de un organismo), en forma de matriz de miles de puntos (10000 – 40000) equiespaciados. Cada secuencia se asocia con un único gen. Cada punto contiene millones de secuencias clonadas "idénticas".

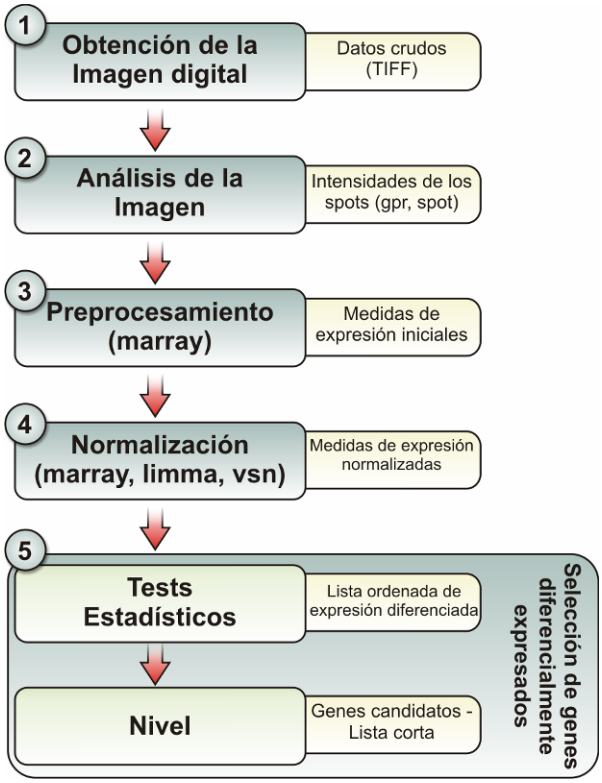
¿Para que sirve un microarray?

Estos permiten medir simultáneamente la actividad y la interacción de cientos de genes. Pueden ser aplicados en investigaciones para descubrir genes, diagnóstico de enfermedades y prognosis, en farmacocinética, en toxicología, etc.

La aplicación típica de los microarreys radica en la utilización de los mismos para la identificación de genes expresados diferencialmente.

Esquema general para el análisis de datos de Microarrays

En el siguiente esquema se resumen en 5 pasos el procedimiento necesario para hallar genes diferencialmente expresados utilizando microarrays de dos canales.



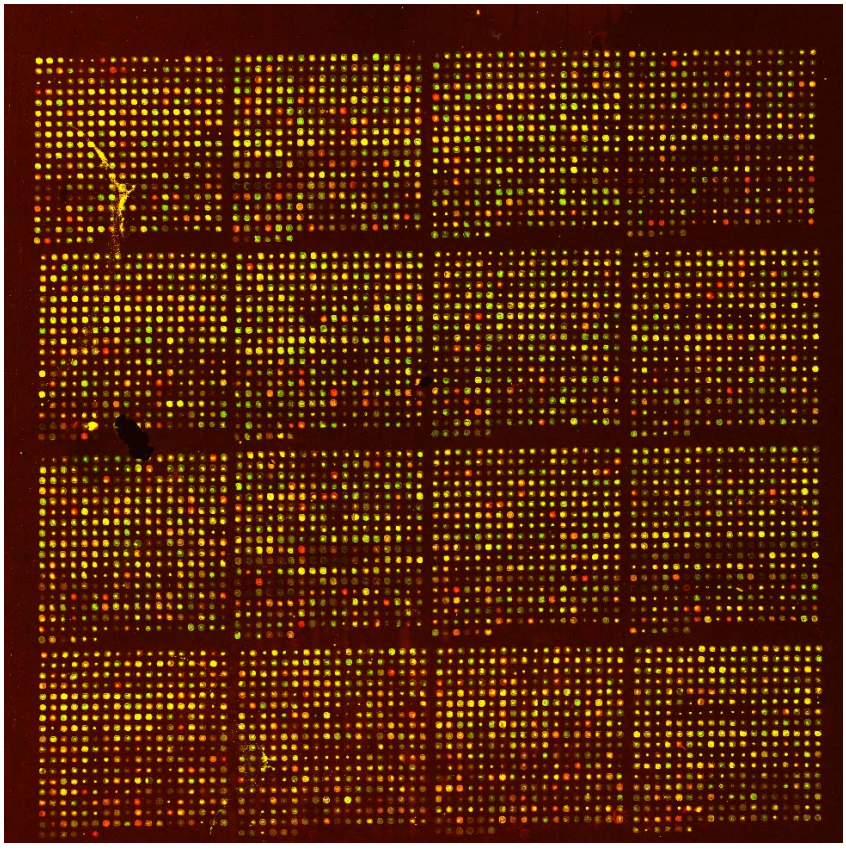
A continuación se procederá a describir cada uno de los pasos, ejemplificando con un caso real.

1. Obtención de la Imagen digital

La imagen puede ser obtenida por medio de dos instrumentos, ***Scanners*** o ***Imagers***. Ambos actúan excitando cada tinte fluorescente de cada target mediante una luz monocromática producida por un laser (para el caso de los scanners) o luz blanca (imagers) y colectando la luz de emisión (fluorescencia) convirtiendo la corriente de fotones en valores digitales que pueden ser almacenados en una computadora como un archivo de imagen (.TIFF). A cada tinte le corresponde una longitud de onda de excitación y una longitud de onda de emisión diferente. Los tintes más usados son los de cianina, Cy3 (verde) y Cy5 (rojo), los cuales tienen emisiones en los rangos de 510-550 nm y 630-660 nm, respectivamente.

Para un típico experimento de microarray, se producen dos archivos de imagen, una para cada tinte fluorescente que en general para resumir, se presentan en una sola imagen superponiendo ambas imágenes. Para el caso en el que las intensidades de las señales de un spot en ambas imágenes sean similares, este spot se mostrará de color amarillo (ver figura 1).

Figura 1: imagen de un microarray de dos canales



2. Análisis de la Imagen

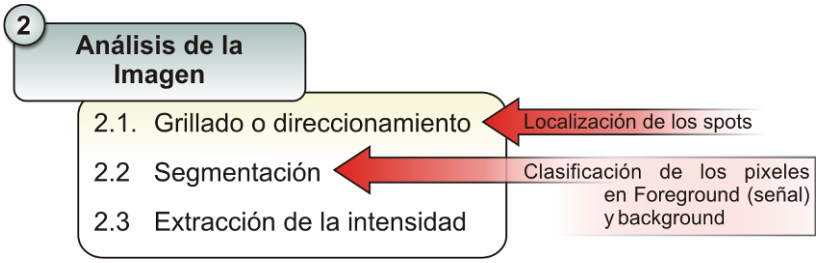
El primer propósito del análisis de la imagen es obtener las intensidades del background[1] y el foreground[2] para los canales rojo y verde de cada spot en el microarray.

[1] Intensidad lumínica de fondo. Esta intensidad no corresponde a un Spot.
[2] Intensidad correspondiente al spot.

Un propósito secundario del análisis de la imagen es obtener medidas cualitativas para cada spot que podría ser usado para marcar spots o arrays defectuosos o para evaluar la reproducibilidad de cada spot.

Luego del análisis de la imagen obtenemos archivos del tipo .gpr, .spot, etc dependiendo del software utilizado. Estos archivos contienen, entre otras cosas, las intensidades de cada canal correspondientes a los spots (Foreground), background, medidas cualitativas (medidas de variabilidad, tamaño del spot, medidas de circularidad, entre otras).

El análisis de la imagen puede separarse en tres etapas, como lo demuestra el siguiente esquema:



2.1 Grillado o direccionamiento

Antes de segmentar la imagen primero se debe identificar la localización de cada spot. Este proceso es llamado grillado o direccionamiento. Bajo condiciones ideales los spot en cada bloque se encuentran equiespaciados y en el mismo lugar donde fueron impresos por las agujas. Sin embargo pequeñas variaciones durante la impresión del array puede causar irregularidades significantes en la imagen.

Para localizar los spots se debe localizar un rectángulo o cuadrado que contenga al spot. Estos rectángulos pueden obtenerse por medio de métodos de grillado automático o manual. Para el segundo caso es necesario estimar ciertos números de parámetros como la separación entre filas y columnas; pequeñas traslaciones individuales de los spots, etc.

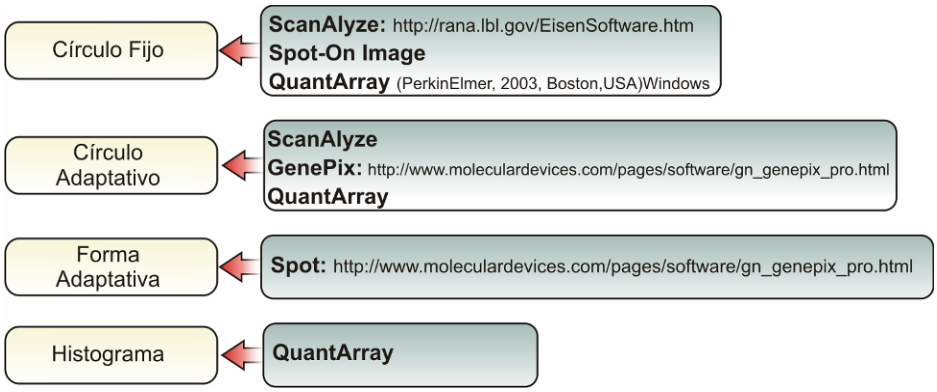
2.2 Segmentación

La segmentación de la imagen puede ser definida como el proceso de particionamiento en dos diferentes regiones, cada una con propiedades diferentes. La segmentación permite la clasificación de pixeles como foreground o background, utilizando una spot mask.

Existen diferentes métodos de segmentación que pueden ser categorizados en cuatro grupos de acuerdo a la geometría de los spots que ellos producen.

Segmentación						
Círculo Fijo		**Círculo Adaptativo**		**Forma Adaptativa**	**Histograma**	
Ventajas	***Desventajas***	***Ventajas***	***Desventajas***	***Ventajas***	***Ventajas***	***Desventajas***
Fácil de implementar. Funciona bien si todos los spots son circulares y del mismo tamaño.	Resultados imprecisos cuando los spots tiene forma invariable.	*Fácil de implementar. Funciona bien si todos los spots son circulares y del mismo tamaño.*	Una máscara circular tiene un mal ajuste, ya que rara vez los spots son circulares en la práctica.	*Uno de estos métodos es el llamado Región Sembrada Creciente (SRG).Este método puede detectar con más precisión a los spots ya que se adapta a sus formas reales.*	*No necesita información espacial.*	Puede producir spots no conexos.

Cada uno de los métodos antes mencionados se implementan en los siguientes software:



2.3 Extracción de las intensidades

Se extraen por un lado las intensidades del foreground calculando el promedio de las intensidades de los pixeles sobre el spot mask. Para el cálculo de la intensidad del background se pueden utilizar el método del **valle** o el denominado **Morphological opening**.

Luego de estimar el background, era común en la práctica, corregir las intensidades del foreground restando las intensidades del background, $R = R_f - R_b$ y $G = G_f - G_b$. La motivación para ajustar el background estaba dada por la creencia de que las intensidades medidas de los spots incluyen una contribución no específica.

En la actualidad existen gran cantidad de papers, en los cuales desestiman la sustracción del background, debido a que producen gran cantidad de errores. Por ejemplo uno de los efectos indeseados de la corrección del background es que se pueden obtener intensidades negativas al sustraer el background. Esto sucede debido a que la intensidad del background es mayor que la del foreground adjudicado, quizás, a errores en la estimación de las intensidades del background. Cuando se calcula el logaritmo de las intensidades, estos valores se pierden, resultando en una pérdida de información.

Como se ha mencionado anteriormente luego de realizar el análisis de la imagen se obtienen archivos con las intensidades, del background y foreground,

entre otras. Estos archivos tendrán diferente extensión dependiendo del software de análisis de la imagen.

Software	Extensión típica
GenePix	`*.gpr`
ScanAlyze	`*.dat`
QuantArray	`*.txt`
Spot	`*.spot, *.dat`

3. Preprocesamiento

Desde el preprosesamiento hasta el punto 5 mostrado en el esquema de análisis se utilizará el software estadístico llamado R que lo podrá bajar de la página web: http://www.r-project.org/

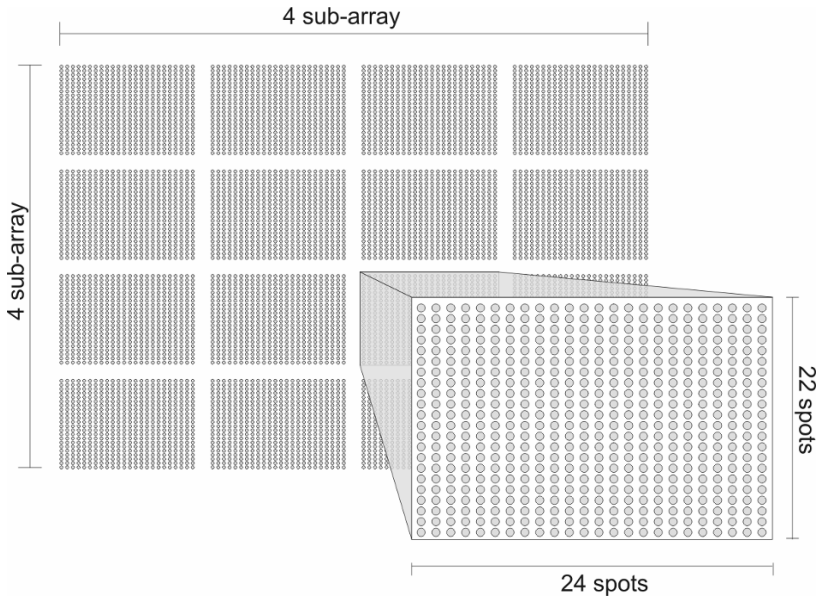
Experimento: Swirl

Para ejemplificar el preprocesamiento de los datos de un experimento de microarray se utilizarán datos de un experimento denominado swirl zebrafish. El experimento se realizó usando peces denominados zebrafish para estudiar el desarrollo temprano en vertebrados. Swirl es una mutación puntual en el gen BMP2 que causa defectos en la organización de los embriones en desarrollo. Un objetivo de este experimento es identificar genes con expresión alterada en el mutante swirl comparado con los zebrafish wild-type. Se utilizaron 4 replicaciones, dos sets de pares dye-swap. Cada ADNc target del mutante fue marcado usando uno de los colorantes Cy3 o Cy5 y el ADNc target wild-type fue marcado usando el otro colorante.

Figura 2: pez zebrafish adulto.

En este caso estudiado, el ADNc target fue hibridizado a los microarrays conteniendo 8848 probes. Los microarrays fueron impresos usando un set de 4x4 agujas, produciendo así matrices de 4x4 subarrays. Cada subarray consistía de 22x24 spots (ver figura 3).

Figura 3: diseño del microarray para el experimento swirl



Para cada una de las cuatro replicaciones produce un par de imágenes (uno para cada canal, rojo y verde), las cuales fueron procesadas usando el software Spot. Luego del análisis de las imágenes se obtiene un grupo de archivos denominados `swirl.1.spot`, `swirl.2.spot`, `swirl.3.spot`, y `swirl.4.spot`. Cada uno de estos archivos contienen 8448 filas y 30 columnas; las filas corresponden a los spots y las columnas a diferentes estadísticos de los spots. También se crea un archivo denominado `SwirlSample.txt` que contiene información sobre la muestra y otro `fish.gal`, el cual almacena información sobre el nombre de los genes y la estructura del arreglo.

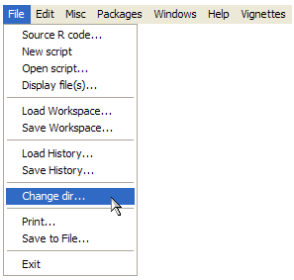
¿Como acceder a los datos?

Para el caso particular de los datos de swirl zebrafish, estos se encuentran incluidos en la librería de marray. Por lo tanto primero se debe cargar la librería marray de la siguiente manera.

```
library(marray)
```

Luego deberá crear un solo objeto en el cual se almacenará toda la información; es decir se deberá guardar toda la información almacenada en los archivos antes mencionados, para luego procesar dicha información.

Primero debemos posicionarnos en el directorio en donde se encuentran almacenados los archivos, para ello deberá seleccionar del menú *file*, la opción *change dir..*, se abrirá una ventana en la cual podrá seleccionar el directorio en donde se encuentran los datos del experimento (`C:/ARCHIVOS DE PROGRAMA/R/ R-2.2.1/library/marray/swirldata/`).



Como segundo paso se deberá leer los datos del target y almacenarlos en la variable `swirl.targets`.

```
swirl.targets<-read.marrayInfo(file.path(file.choose()))
```

Crearemos el objeto `swirl1` y almacenaremos en su interior las intensidades y la información de los targets.

```
swirl1<-read.Spot(path=".", targets=swirl.targets)
```

Luego leeremos la información sobre los probes almacenados en el archivo `fish.gal`.

```
swirl.probes<- read.Galfile(galfile="fish.gal")

swirl1@maLayout<-swirl.probes$layout

swirl1@maGnames<-swirl.probes$gnames
```

Tipee el siguiente comando:

```
Swirl1@maTargets
```

R mostrará los siguientes datos:

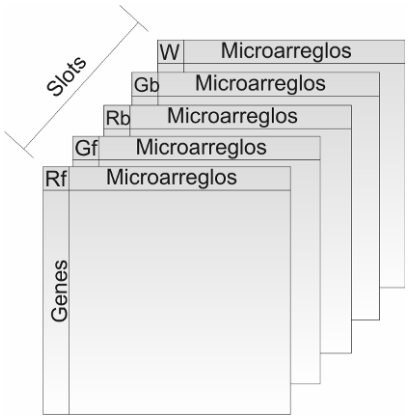
```
An object of class "marrayInfo"
@maLabels
[1] "swirl.1.spot" "swirl.2.spot" "swirl.3.spot" "swirl.4.spot"

@maInfo
         Names slide number experiment Cy3 experiment Cy5      date comments
1 swirl.1.spot           81          swirl      wild type 2001/9/20       NA
2 swirl.2.spot           82      wild type          swirl 2001/9/20       NA
3 swirl.3.spot           93          swirl      wild type 2001/11/8       NA
4 swirl.4.spot           94      wild type          swirl 2001/11/8       NA
```

```
@maNotes
[1] "C:\\Archivos de programa\\R\\R-2.2.1\\library\\marray\\swirldata\\SwirlSample.txt"
```

Esta acción nos muestra los 4 microarray y el diseño del experimento con respecto a los tinetes.

Para entender mejor cual es la estructura de datos en el paquete marray, y como se almacena la información observemos el siguiente esquema.



Cada intensidad es almacenada en un slot (una matriz) en la cual los genes se encuentran en las filas y cada uno de los microarreglos en las columnas.

Para poder comprobar cuales son los slots se debería ingresar los siguientes comandos en R.

```
slotNames(swirl1)
```

R nos mostrará todos los nombres de los slots:

```
[1] "maRf"     "maGf"     "maRb"     "maGb"     "maW"      "maLayout"
[7] "maGnames"  "maTargets" "maNotes"
```

Si se quiere leer, por ejemplo, los datos de las intensidades del foreground para los primeros 4 spots del canal rojo se debe tipear en R lo siguiente:

```
> swirl1@maRf[1:4,]
```

El símbolo @ permite acceder a los slots por su nombre.

Slots	Información
maRf	Intensidad del foreground para el canal rojo
maGf	Intensidad del foreground para el canal verde
maRb	Intensidad del background para el canal rojo
maGb	Intensidad del background para el canal verde
maGnames	Nombre de los genes
maTargets	Información sobre el ADNc target

Para poder observar un resumen de todos los datos del experimento se debería ingresar el siguiente comando.

```
> summary(swirl1)
```

Presentación gráfica

Es muy útil en la práctica realizar gráficos que permitan examinar los resultados de cualquier experimento de microarray. Un gráfico puede ayudar a

evaluar el éxito de un experimento, puede guiar la elección de herramientas de análisis y puede resaltar problemas específicos.

- **MA-plot**

El gráfico más común es el denominado MA-plot de Dudoit et al. En este se grafican los valores M (también conocido como log-ratio) en el eje vertical y los valores A en el horizontal.

¿Cómo se calculan dichos valores?

Se definen los valores M y A de la siguiente manera.

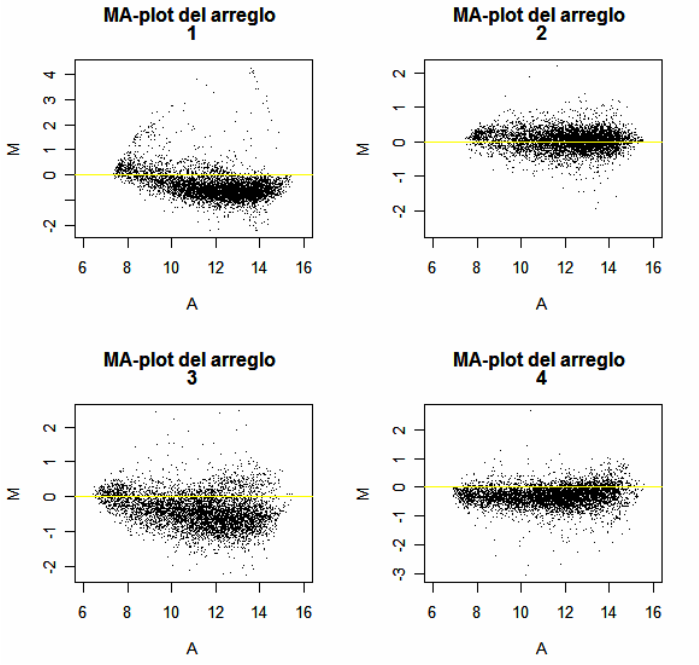
$$M = \log_2 \frac{R_f}{G_f}$$

$$A = \frac{\log_2 R_f + \log_2 G_f}{2}$$

Para construir un gráfico MA-plot, para cada uno de los arrays, en R se debe ingresar los siguientes comandos:

```
par(mfrow=c(2,2))
for(i in 1:4) {
   plot(0.5*(log(slot(swirl1,"maRf")[,i],2)+log(slot(swirl1,"maGf")[,i
   ],2)),log(slot(swirl1,"maRf")[,i],2)-
   log(slot(swirl1,"maGf")[,i],2),xlab="A",ylab="M",pch=".",xlim=c(6,
   16),main=c("MA-plot del arreglo ",i))
abline(h=0,col=7)
}
```

Se obtendrán los siguientes gráficos:



Un gráfico MA-Plot ideal es aquel en el cual toda la nube de puntos se encuentra sobre el valor 0 del eje M.

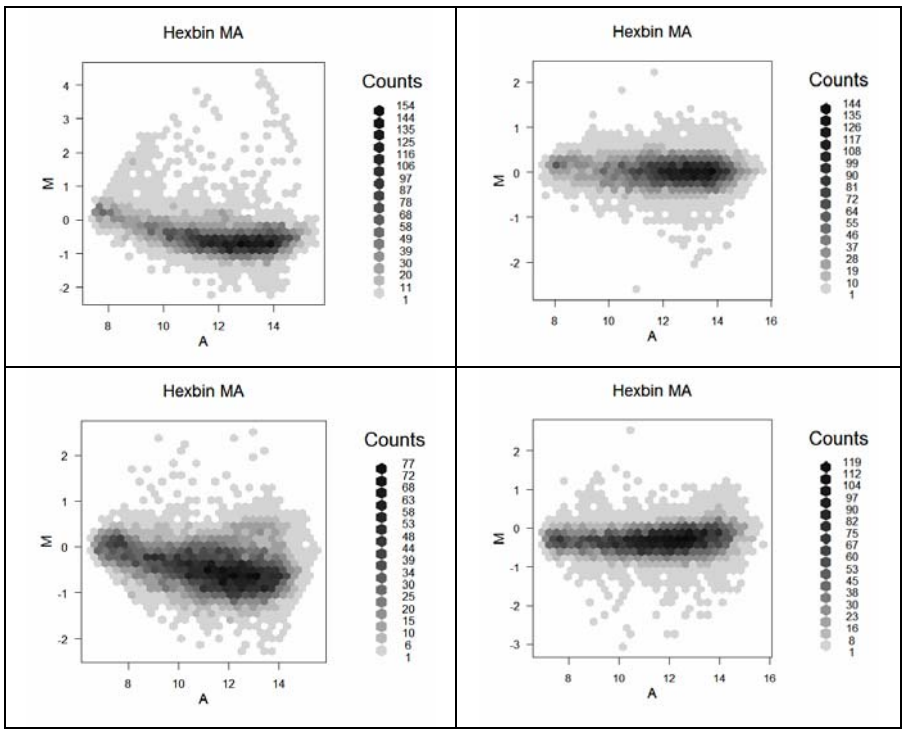
Todas aquellas desviaciones con respecto al ideal se deben principalmente a errores sistemáticos vinculados con varios factores, como ser: diferencias en la eficiencia de la incorporación de los tintes, diferencias en la cantidad de ARNm,

diferencias en los parámetros de escaneado, diferencias entre los grupos de agujas, efectos espaciales y efectos debido al plato.

Un gráfico MA-plot que es muy útil y que difiere del antes visto es el gráfico Hexbin MA-plot. Este gráfico tiene en cuenta la superposición de puntos.

```
library(arrayQuality)
for(i in 1:4) {
 Rf<-swirl1@maRf[,i]
 Gf<-swirl1@maGf[,i]
 M<-log2(Rf/Gf)
 A<-0.5*log2(Rf*Gf)
 X11()
 plot(hexbin(A,M),main="Hexbin MA")
}
```

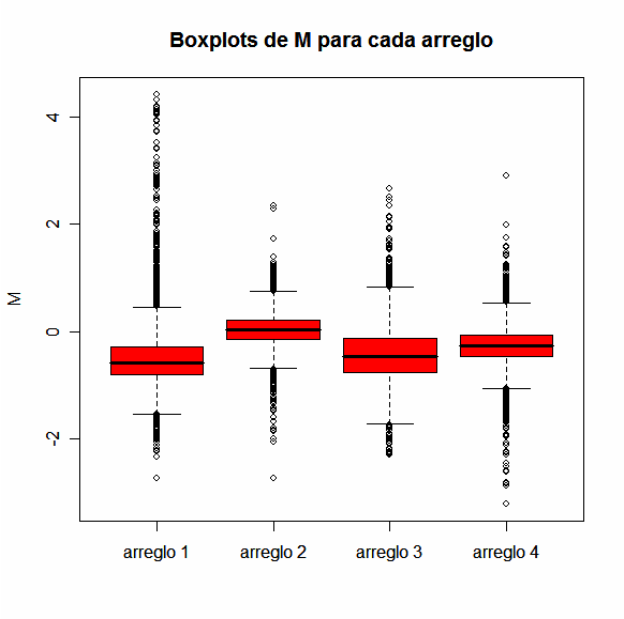
Estos comandos generan los siguientes gráficos.



- **Boxplots**

Otros de los gráficos utilizados en el análisis de datos de microarray son los Boxplots, que pueden ser útiles para comparar los valores M entre los arrays.

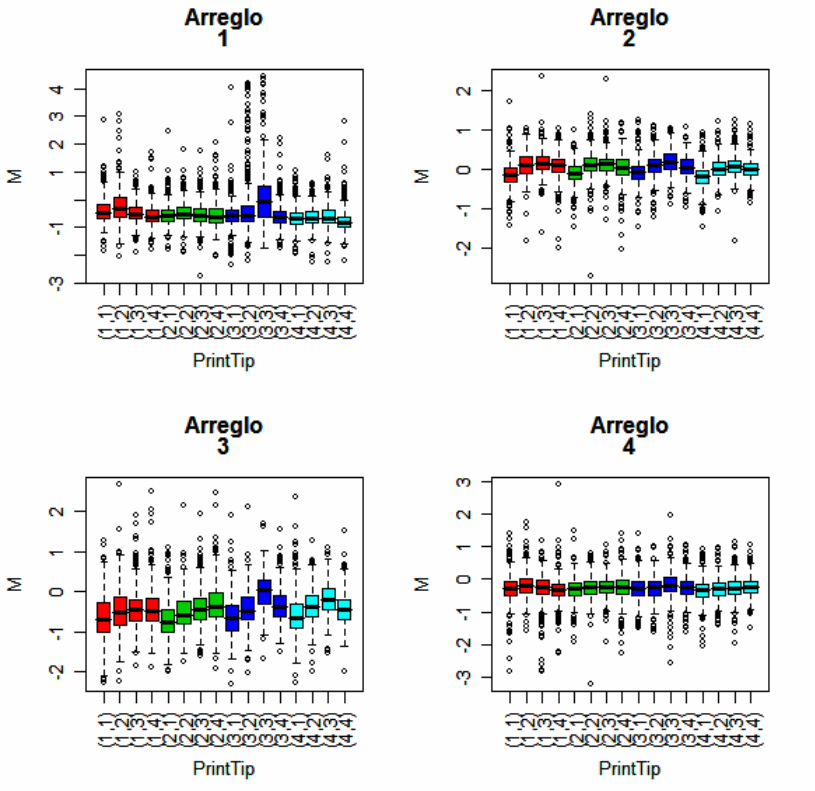
```
 par(mfrow=c(1,1))
 boxplot(swirl1,names=c("Arreglo 1","Arreglo 2","Arreglo 3","Arreglo
4"))
 title("Boxplots de M para cada arreglo")
```



Los Boxplot muestran gráficamente 5 números resumen, los tres cuartiles, el máximo y el mínimo. La caja central del gráfico que va desde el primer cuartil hasta el tercero encierra el 50% de los datos.

Una de las opciones al realizar un boxplot es graficar el valor M en función de las agujas.

```
par(mfrow=c(2,2))
for(i in 1:4) {
    boxplot(swirl1[,i],xvar="maPrintTip",yvar="maM",main=c("Arreglo",i
))
}
```



Al analizar los gráficos anteriores, estos no deberían presentar ninguna tendencia, entre los distintos grupos de agujas (print-tip group). Si se observa detenidamente el primer array se puede observar una pequeña tendencia decreciente entre los print-tip group. Los valores correspondientes a las agujas de la primera fila son superiores a los de la última.

- **Gráficos Espaciales**

Los gráficos espaciales son gráficos de dos dimensiones en el cual cada punto representa un estadístico del spot del microarreglo. Estos gráficos se utilizan para explorar la calidad de hibridación de cada arreglo.

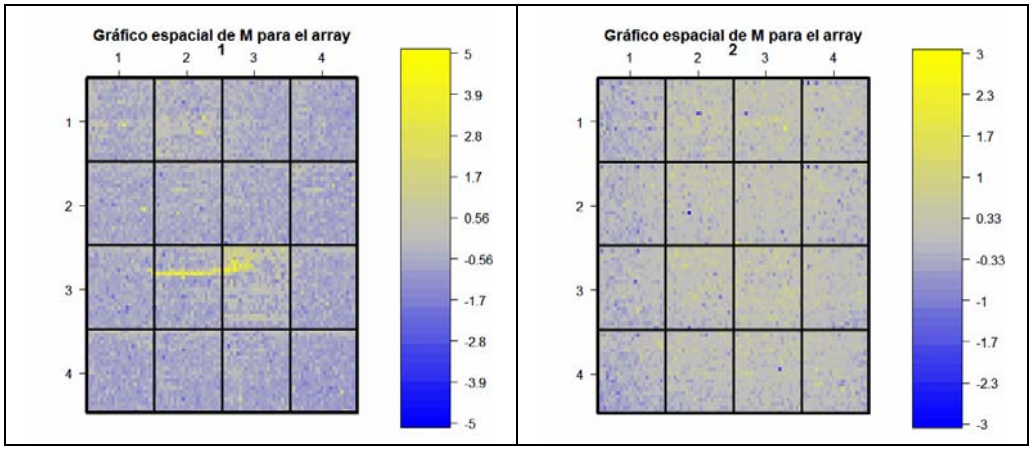
Ingrese los siguientes comandos para obtener los gráficos espaciales para los valores de M y para el background de cada canal:

```
for(i in 1:4) {
   X11()
   image(swirl1[,i], main=c("Gráfico espacial de M para el array",i))
}


for(i in 1:4) {
   X11()
   image(swirl1[,i],xvar="maRb", main=c("Gráfico espacial de Rb para
el array",i))
}


for(i in 1:4) {
   X11()
   image(swirl1[,i],xvar="maGb", main=c("Gráfico espacial de Gb para
el array",i))
}
```

El resultado es el siguiente:



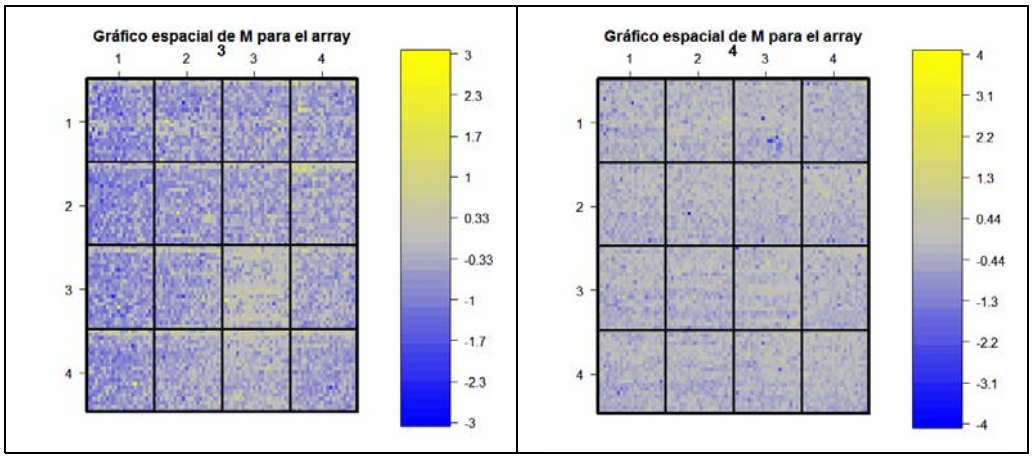


Gráfico espacial de M para el array
3
3
2.3
1.7
1
0.33
-0.33
-1
-1.7
-2.3
-3
Gráfico espacial de M para el array
4
4
3.1
2.2
1.3
0.44
-0.44
-1.3
-2.2
-3.1
-4

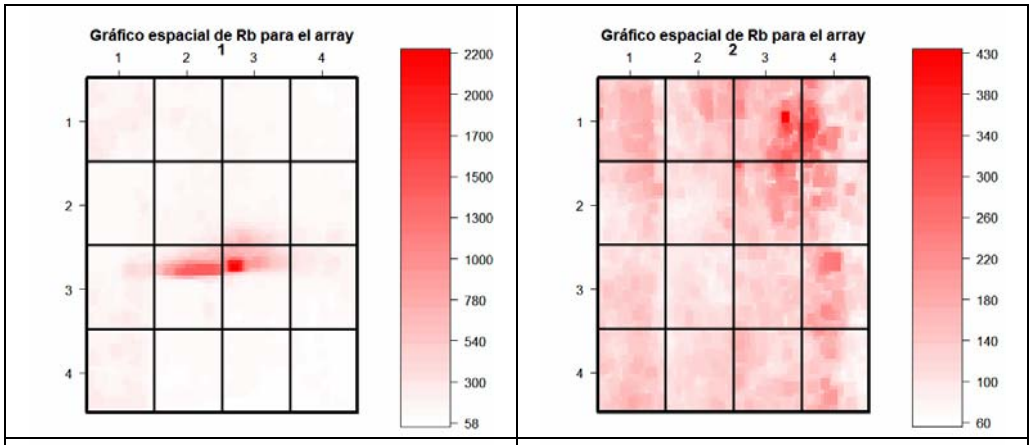


Gráfico espacial de Rb para el array
1
2200
2000
1700
1500
1300
1000
780
540
300
58
Gráfico espacial de Rb para el array
2
430
380
340
300
260
220
180
140
100
60

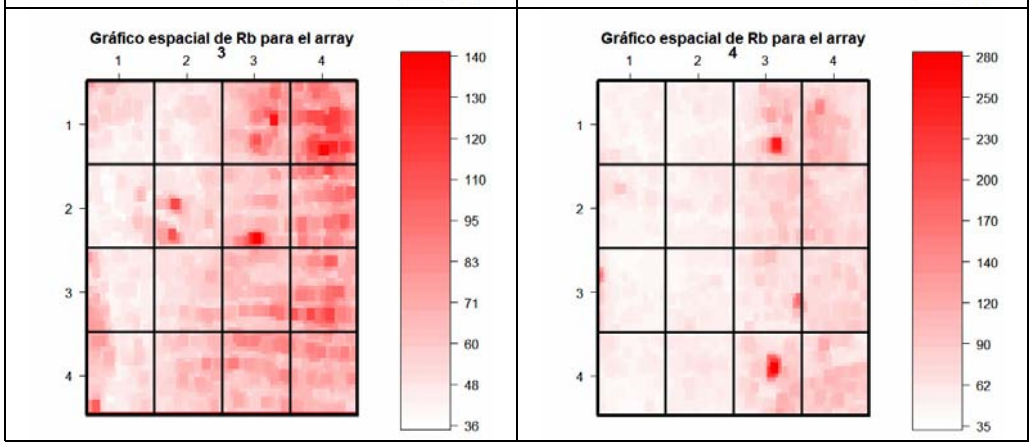


Gráfico espacial de Rb para el array
3
140
130
120
110
95
83
71
60
48
36
Gráfico espacial de Rb para el array
4
280
250
230
200
170
140
120
90
62
35

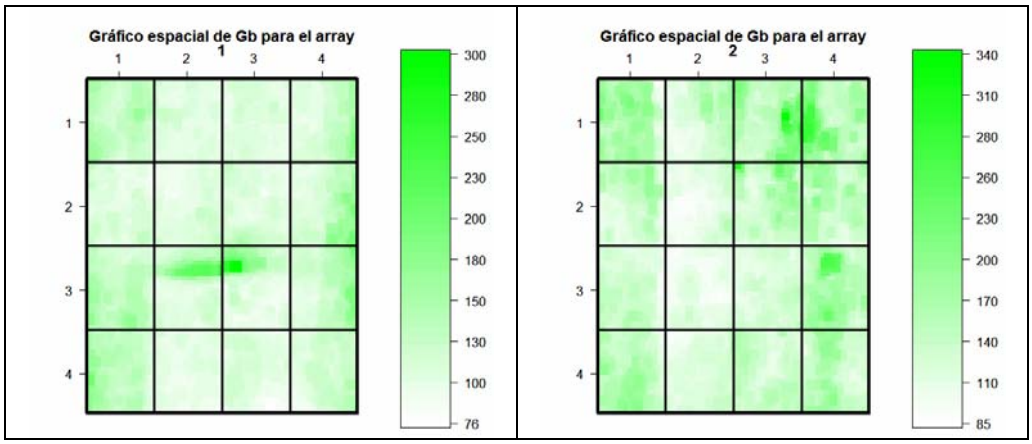
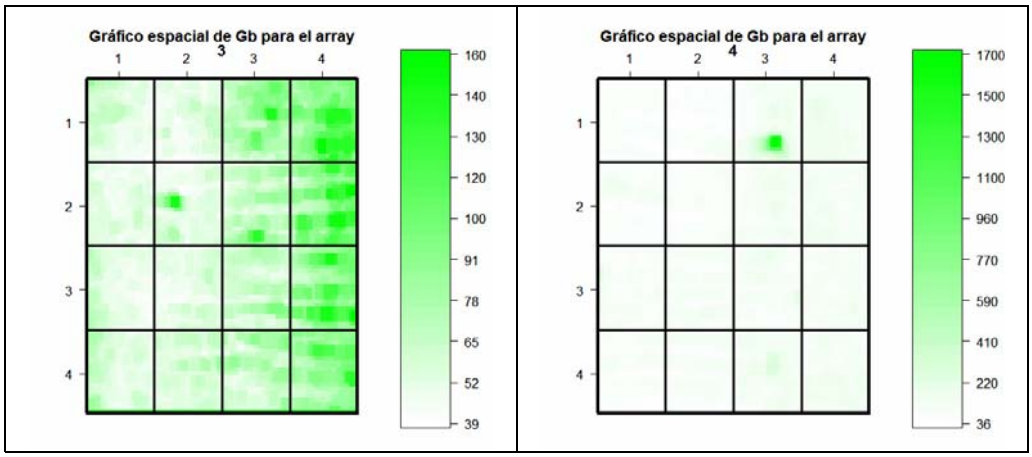


Gráfico espacial de Gb para el array
1
300
280
250
230
200
180
150
130
100
76
Gráfico espacial de Gb para el array
2
340
310
280
260
230
200
170
140
110
85

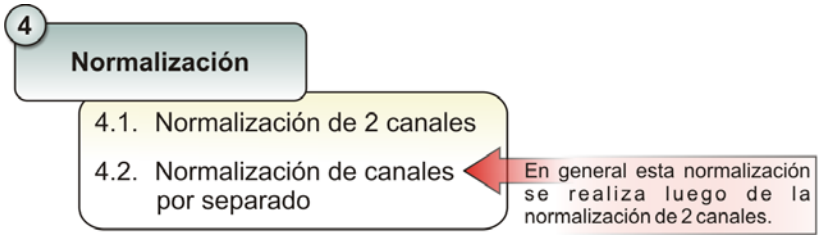


Lo que se pretende observar en cada gráfico es una homogeneidad de colores en todo el microarray. Como se puede observar en el gráfico espacial para las intensidades del background del canal verde del 3° microarray, existe un efecto espacial, ya que las máximas intensidades se agrupan a la derecha del microarray.

4. Normalización

El propósito de la normalización es solucionar ciertas variaciones en la tecnología del microarray que pueden provenir de diferencias biológicas entre las muestras de ARN o los probes impresos. Se pretende solucionar algunos errores sitemáticos, descriptos en la página 8.

La necesidad de normalización puede verse más claramente en experimentos self-self[3]. Aunque no hay expresión diferencial, la intensidad del canal rojo a menudo tiende a ser más baja que las intensidades del canal verde. Más aún estas intensidades son usualmente no constantes entre los spots dentro y entre arrays.



4.1 Normalización de 2 canales

Este tipo de normalización se realiza en R con la función maNorm(). Esta función permite realizar diferentes tipos de normalizaciones, especificando los siguientes parámetros.

[3] Experimentos en el cual dos muestras idénticas de ARNm son marcadas con diferentes colorantes e hibridizadas al mismo microarray.

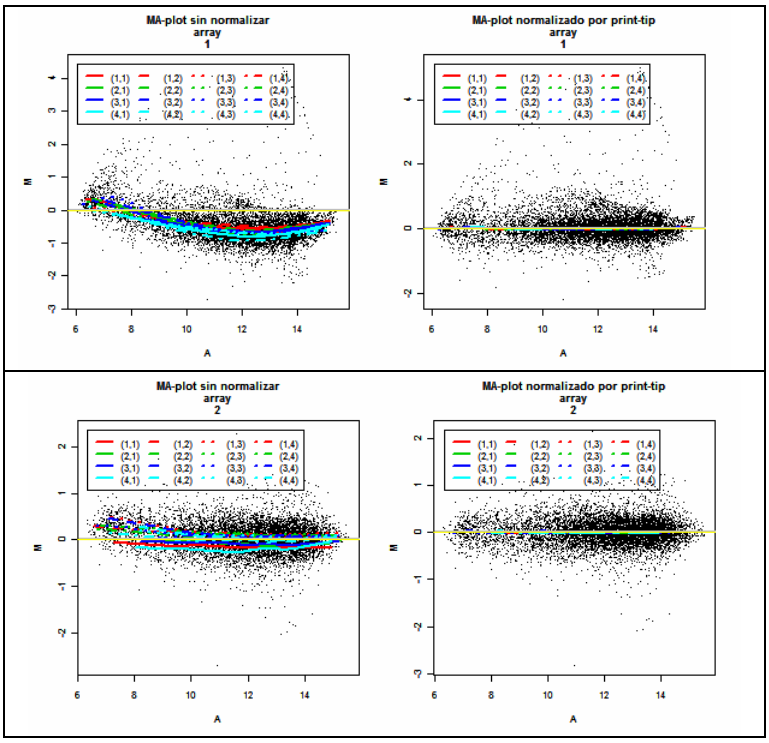
Nombre	Descripción
`none`	No realiza ninguna normalización
`median`	Normalización global por mediana
`loess`	Normalización global dependiente de A
`printTipLoess`	Normalización A dependiente, dentro del grupo que determina cada aguja, utilizando la función loess
`twoD`	Normalización espacial utilizando la función loess
`scalePrintTipMAD`	Normalización A dependiente, dentro del grupo de cada aguja utilizando la función loess, seguida de una normalización de escala dentro de cadagrupo utilizando la función MAD

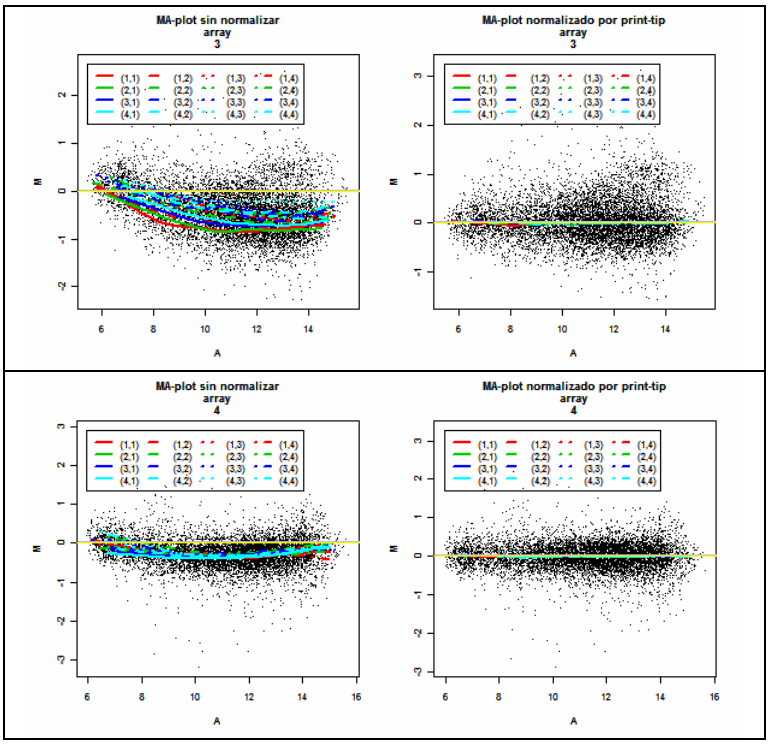
La siguiente instrucción permite realizar una normalización Print-tip-Loess.

```
swirl.normP<-maNorm(swirl1, norm="p")
```

Para observar el efecto de la normalización se realizarán los siguientes gráficos comparando con los datos no normalizados:

```
X11(width = 6, height = 3, pointsize = 7)
par(mfrow=c(1,2))
for(i in 1:4) {
    plot(swirl1[,i], main=c("MA-plot sin
normalizar","array",i),pch="." )
    abline(h=0,col=7)
    plot(swirl.normP[,i],main=c("MA-plot normalizado por print-
tip","array",i),pch=".")
    abline(h=0,col=7)
    X11(width = 6, height = 3, pointsize = 7)
    par(mfrow=c(1,2))
}
```

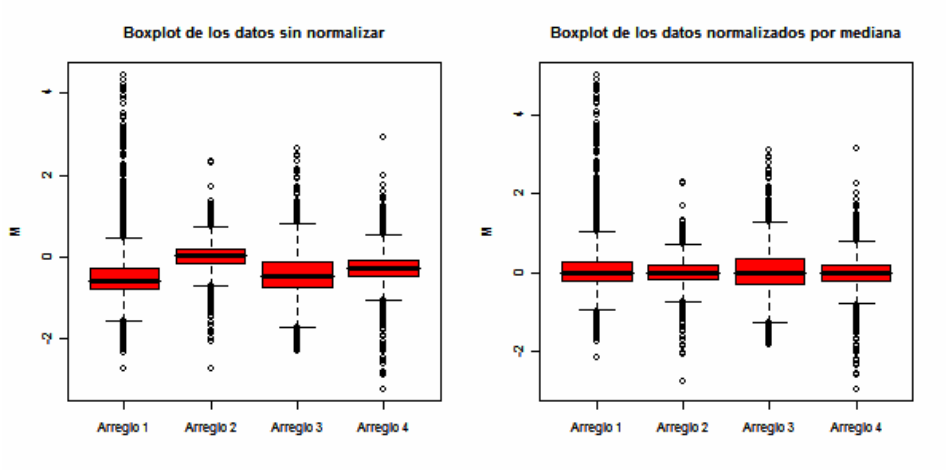




Como se pueden observar en los gráficos la normalización por Print-tip moviliza toda la nube de puntos hacia el valor 0 de M, solucionando los errores sistemáticos debido a las agujas.

La normalización anterior se realizó dentro de cada array, pero también puede hacer una normalización entre arrays, por ejemplo normalizando por las medianas.

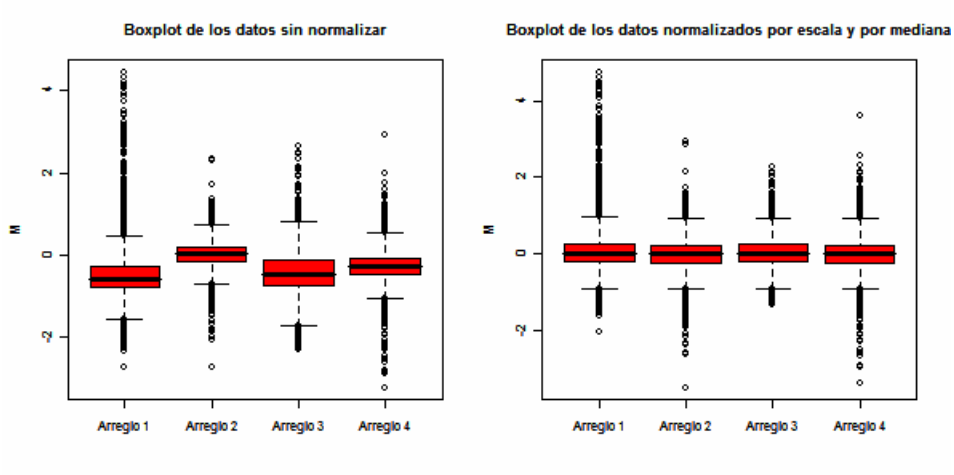
```
  swirl.normM<-maNorm(swirl1,norm="m")
  X11(width = 6, height = 3, pointsize = 7)
  par(mfrow=c(1,2))
  boxplot(swirl1, main=("Boxplot de los datos sin
normalizar"),names=c("Arreglo 1","Arreglo 2","Arreglo 3","Arreglo 4"))
  boxplot(swirl.normM,main=("Boxplot de los datos normalizados por
mediana"),names=c("Arreglo 1","Arreglo 2","Arreglo 3","Arreglo 4"))
```



La normalización por mediana modifica las medianas de cada array, dándole valor 0 a cada una como se observa en los gráficos anteriores.

Existe una normalización más drástica que modifica la escala entre los arrays. Se recomienda evaluar la realización de esta normalización, es decir si no existen diferencias muy marcadas entre los arrays no se recomienda aplicar esta normalización. La función que permite realizar esta normalización es `maNormScale` del paquete marray.

```
 swirl.normS<-maNormScale(swirl.normM)
 X11(width = 6, height = 3, pointsize = 7)
 par(mfrow=c(1,2))
 boxplot(swirl1, main=("Boxplot de los datos sin
normalizar"),names=c("Arreglo 1","Arreglo 2","Arreglo 3","Arreglo 4"))
 boxplot(swirl.normS,main=("Boxplot de los datos normalizados por
escala y por mediana"),names=c("Arreglo 1","Arreglo 2","Arreglo
3","Arreglo 4"))
```



En este caso no solo se igualan las medianas sino que también se igualan los cuartilos.

4.2 Normalización de canales por separado

Dentro de este tipo de normalizaciones se encuentra la normalización por cuantiles (Quantile Normalization). Se realiza utilizando la función normalizeBetweenArrays del paquete limma. Como los datos del objeto `swirl1` son del tipo marray, se deberá realizar una conversión de formatos, entre marray a RGlist (formato de limma).

Para la conversión se debe cargar primero la librería `convert`.

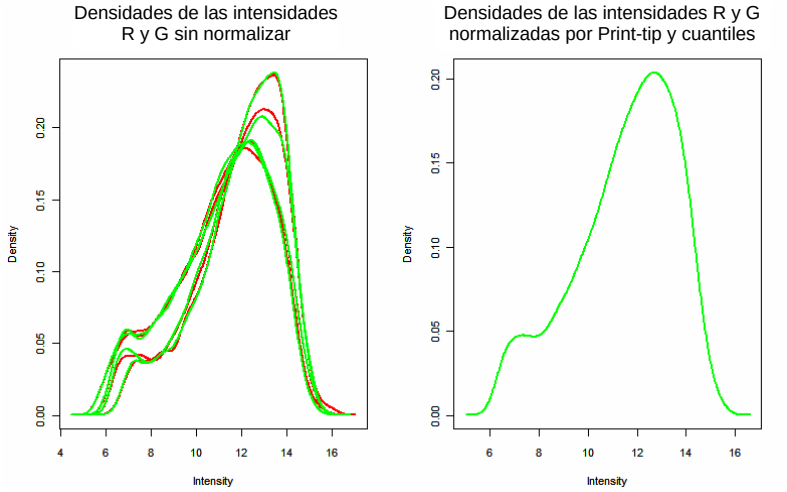
```
library (convert)
```

Luego se realizan las conversiones y la normalización.

```
swirl.SN<- as(swirl1,"RGList")
swirl.P<- as(swirl.normP, "MAList")
swirl.PQ<- normalizeBetweenArrays(swirl.P,method="quantile")
```

Para observar el efecto de la normalización se graficará de la siguiente manera:

```
X11(width = 8, height = 5, pointsize = 8)
par(mfrow=c(1,2))
plotDensities(swirl.P)
plotDensities(swirl.PQ)
```

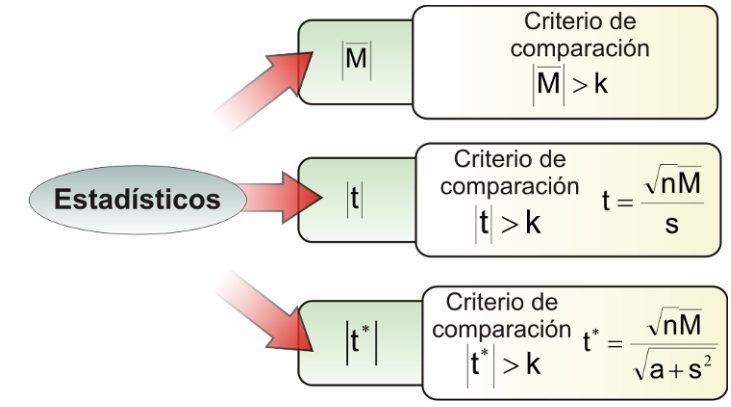


Mediante el método de normalización por cuantiles, las distribuciones de las intensidades de los canales rojo y verde de todos los arreglos quedan idénticas, como se puede observar en las gráficas anteriores.

5. Selección de genes diferencialmente expresados (DE)

Uno de los principales objetivos del análisis de datos de microarray es identificar cuales de los genes muestra una buena evidencia de de estar diferencialmente expresado. La selección de los genes DE puede separarse en dos pasos.

5.1. El primer paso es seleccionar un estadístico el cual permitirá rankear a los genes en función de su expresión diferencial, desde una evidencia más fuerte hasta una más débil.



$\overline{M}$ = es el promedio de los valores M entre cada array para cada uno de los genes
n = tamaño de la muestra
s = desvío estandar de los M valores
a = parámetro que se estima por medio de un método bayesiano

5.2. El segundo es elegir un valor crítico para el ordenamiento anterior de los genes por encima del cual cualquier valor resulta significativo, define la cantidad de genes que se considerarán DE.

Seguiremos trabajando con el ejemplo del experimento Swirl.

Primero deberá asegurarse de que el directorio de trabajo sea el directorio en donde se encuentran los archivos del experimento.

Luego deberemos leer la lista de ARN target hibridado a cada canal de cada arreglo, y los nombres de los archivos que contienen la información de las intensidades. Para ello utilizaremos la función `readtargets` de la librería `limma`.

```
library (limma)
targets <- readTargets("SwirlSample.txt")
```

Para leer los datos de intensidades utilizaremos la función `read.maimages` y lo almacenaremos en un objeto denominado RG de clase RGList. En RG se almacenarán las intensidades del R y G del foreground con sus correspondientes intensidades para el background.

```
RG <- read.maimages(targets$Names, source="spot")
```

Agregaremos a RG la información de los nombres de los genes y también la información de la estructura del arreglo.

```
RG$genes <- readGAL("fish.gal")
RG$printer<-getLayout(RG$genes)
```

Luego debemos proceder a normalizar los datos almacenados en RG, utilizando la función de `limma` denominada `normalizeWithinArrays`. Esta función realiza primero una corrección por background. Por default realiza una normalización por print-tip y calcula los valores M y A.

```
MA<-normalizeWithinArrays(RG)
```

Podemos realizar gráficos MA-plot con la función `plotMA` de la siguiente manera:

```
par(mfrow=c(1,2))
for(i in 1:4) {
    X11(width = 6, height = 3, pointsize = 7)
    par(mfrow=c(1,2))
    plotMA(RG, array=i, main=c("MA-plot sin normalizar array",i))
    abline(h=0,col=7)
    plotMA(MA, array=i,main=c("MA-plot normalizado array",i))
    abline(h=0,col=7)
}
```

Luego de normalizar los datos, se ajusta un modelo lineal a los valores de M de cada gen, que permitirá obtener un estadístico $\hat{M}$. Para el ejemplo que estamos utilizando tendremos 4 ecuaciones, una para cada array. Las ecuaciones tendrán la siguiente forma:

$$Y = X\beta + \varepsilon$$

Donde X es una matriz de diseño que estará dado por el tipo de experimento. El experimento swirl fue realizado con intercambio de tintes (dye-swap) por lo tanto la matriz resultará:

$$X = \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}$$

Los valores negativos corresponden a los arreglos con dye-swap.

El modelo se ajusta usando la función `lmFit`. Primeramente se creará la matriz de diseño.

```
X<-c(1,-1,1,-1)
```

Luego se ajustará al modelo lineal:

```
fit1 <- lmFit(MA, design=X)
summary(fit1)
```

Al resumir el contenido de `fit1` se puede observar lo siguiente:

```
                Length Class      Mode
coefficients    8448   -none-     numeric
rank               1   -none-     numeric
assign             0   -none-     NULL
qr                 5   qr         list
df.residual     8448   -none-     numeric
sigma           8448   -none-     numeric
cov.coefficients   1   -none-     numeric
stdev.unscaled  8448   -none-     numeric
pivot              1   -none-     numeric
method             1   -none-     character
design             4   -none-     numeric
genes              5   data.frame list
Amean           8448   -none-     numeric
```

En `coefficients` se almacenan los $\overline{M}$ para cada gen.

Este estadístico presenta problemas, por lo tanto se calculará un estadístico denominado t * (t moderado).

Se utilizará la función eBayes que calculará el estadístico t moderado para cada gen.

```
fit1.bayes<-eBayes(fit1)
summary(fit1.bayes)
```

Se obtiene la siguiente salida:

```
               Length Class      Mode
coefficients    8448   -none-     numeric
rank               1   -none-     numeric
assign             0   -none-     NULL
qr                 5   qr         list
df.residual     8448   -none-     numeric
sigma           8448   -none-     numeric
cov.coefficients   1   -none-     numeric
stdev.unscaled  8448   -none-     numeric
pivot              1   -none-     numeric
method             1   -none-     character
design             4   -none-     numeric
genes              5   data.frame list
Amean           8448   -none-     numeric
df.prior           1   -none-     numeric
s2.prior           1   -none-     numeric
```

```
var.prior           1   -none-     numeric
proportion          1   -none-     numeric
s2.post          8448   -none-     numeric
t                8448   -none-     numeric
p.value          8448   -none-     numeric
lods             8448   -none-     numeric
F                8448   -none-     numeric
F.p.value        8448   -none-     numeric
```

En **`lods`** se almacenan los estadísticos t moderados.

Para ver los 10 primeros genes expresados diferencialmente, ordenados por el estadístico B, se utilizará la función `topTable`.

```
topTable(fit1.bayes,coef=1,number=10,genelist=fit1.bayes$genes,adjust.method="BH",sort.by="B",resort.by=NULL)
```

Se obtiene:

```
     Block Row Column      ID   Name         M        A         t      P.Value
2961     6  14      9 fb85d05 18-F10  2.657165 10.34821  20.78021 1.332187e-07
3723     8   2      3 control   Dlx3  2.190034 13.27927  17.55944 4.296156e-07
1611     4   2      3 control   Dlx3  2.189233 13.48893  16.08003 7.898498e-07
7649    15  11     17 fb58g10 11-L19  1.597736 13.51875  14.12596 1.926656e-06
515      1  22     11 fc22a09 27-E17 -1.264942 13.16892 -13.62240 2.470493e-06
7491    15   5      3 fb24g06  3-D11 -1.319002 13.64346 -13.56782 2.539284e-06
4454     9  10     14 fb54e03  10-K5  1.199001 13.15679  13.05461 3.304493e-06
319      1  14      7 fb85a01  18-E1  1.287253 12.54610  12.96990 3.454511e-06
7036    14   8      4 fb40h07  7-D14 -1.350680 13.84314 -12.66262 4.067523e-06
8295    16  16     15 fb94h06 20-L12 -1.276615 12.04342 -12.50151 4.437776e-06
       adj.P.Val        B
2961 0.001125432 7.615431
3723 0.001814696 6.807814
1611 0.002224217 6.347681
7649 0.003159151 5.628394
515  0.003159151 5.418860
7491 0.003159151 5.395487
4454 0.003159151 5.169090
319  0.003159151 5.130538
7036 0.003159151 4.987744
8295 0.003159151 4.910989
```

En donde se puede observar los nombres de los genes que se encuentran DE en la columna `Name`.

Si queremos obtener todos los genes cuyo p-valor < 0.05, primero debemos almacenar en una variable todos los genes ordenados por el estadístico B.

```
tabla8448<-
topTable(ajuste,coef=1,number=8448,genelist=ajuste$genes,adjust.method
="BH",sort.by="B")

menores <- subset(tabla8448, P.Value < 0.005, c(ID, M, A, t, P.Value,
B))
menores[1,]
```

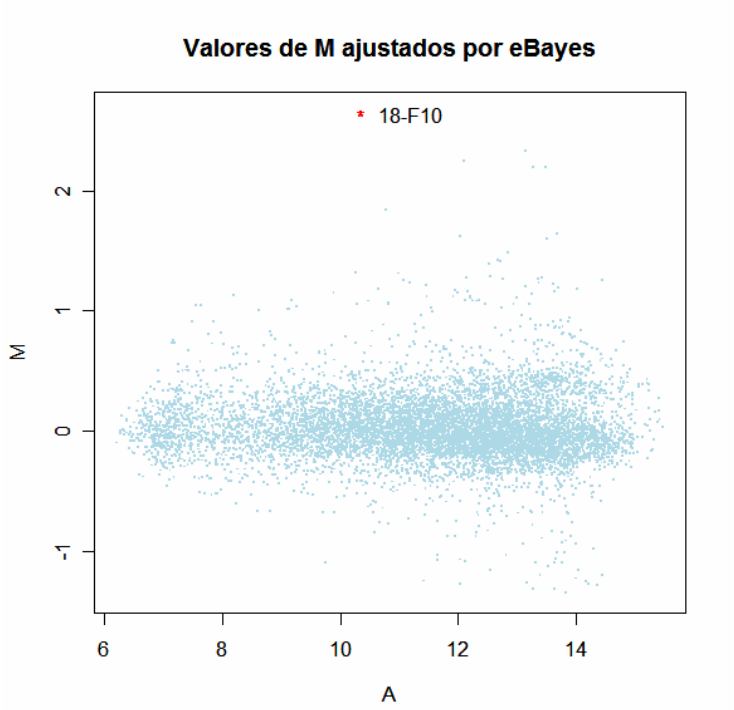
Podemos observar:

```
         ID       M        A        t      P.Value        B
2961 fb85d05 2.657165 10.34821 20.78021 1.332187e-07 7.615431
```

Luego podemos observar visualmente el gen de mayor expresión diferencial de la siguiente forma:

```
M<- fit1.bayes$coefficients
A<- fit1.bayes$Amean
plot (A, M, pch=".", col="lightblue", cex=2, main="Valores de M
  ajustados por eBayes")
points(tabla8448$A[1], tabla8448$M[1], pch="*", col=2,
  cex=1);text(11.2,2.64,tabla8448[1,5])
```

Observamos el siguiente gráfico:



Bibliografía

Kelmansky, D. ***Apuntes del curso: Análisis Exploratorio y Confirmatorio de Datos de Experimentos de Microarrays***. Buenos Aires, 2006.

Li, Q.; Fraley, C.; Bumgarner, R.; Yeung, K.; Raftery, A. ***Donuts, Scratches and Blancks: Robust Model-Based Segmentation of Microarray Images***. Technical Report no. 473. USA, 2005.

Yang, Y.; Buckley, M.; Dudoit, S.; Speed, T. ***Comparison of methods for image analysis on cDNA microarray data***. Technical Report no. 473. USA, 2000.

Smyth, G.; Yang, Y.; Speed, T. ***Statistical Issues in cDNA Microarray Data Analysis***. Methods in Molecular Biology series, Humana Press, Totowa, NJ, 2002.

Speed, T. ***Statistical Analysis Of Gene Expression Microarray Data***. Chapman & Hall/CRC CRC Press LLC, 2003.

Berrar, D.; Dubitzky, W.; Granzow, M. ***A Practical Approach To Microarray Data Analysis***. Kluwer Academic Publishers, 2003.