DNA Microarrays

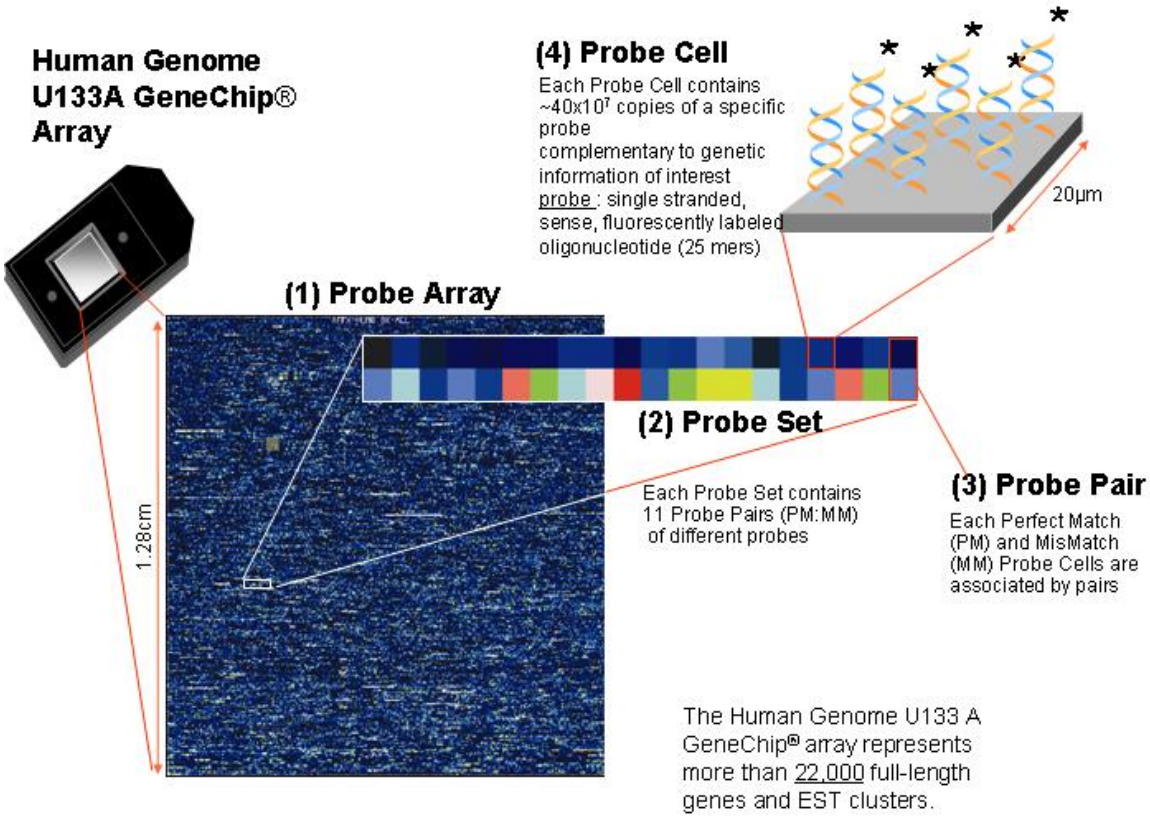
Guillermo Ayala Gallego

2025-02-27

Table of contents

Microarrays

DNA Microarrays



Un ejemplo

- gse21779

Affymetrix Genechip

- ¿Qué tipo de objeto tenemos? ¿Cuál es su clase?

```
class(gse21779raw)
```

```
[1] "AffyBatch"
attr(,"package")
[1] "affy"
```

- ¿Qué anotación tiene?

```
pacman::p_load(Biobase,affy)
annotation(gse21779raw)
```

```
[1] "hgu133plus2"
```

- ¿Cuántas sondas y muestras?

```
dim(exprs(gse21779raw))
```

```
[1] 1354896      18
```

La primera muestra

AffyID

```
head(probeNames(gse21779raw))
```

```
[1] "1007_s_at" "1007_s_at" "1007_s_at" "1007_s_at" "1007_s_at" "1007_s_at"
```

```
(ID = probeNames(gse21779raw)[400])
```

```
[1] "1552281_at"
```

```
sum(probeNames(gse21779raw) == ID)
```

```
[1] 11
```

- ¿Qué tamaños tienen los grupos de sondas?

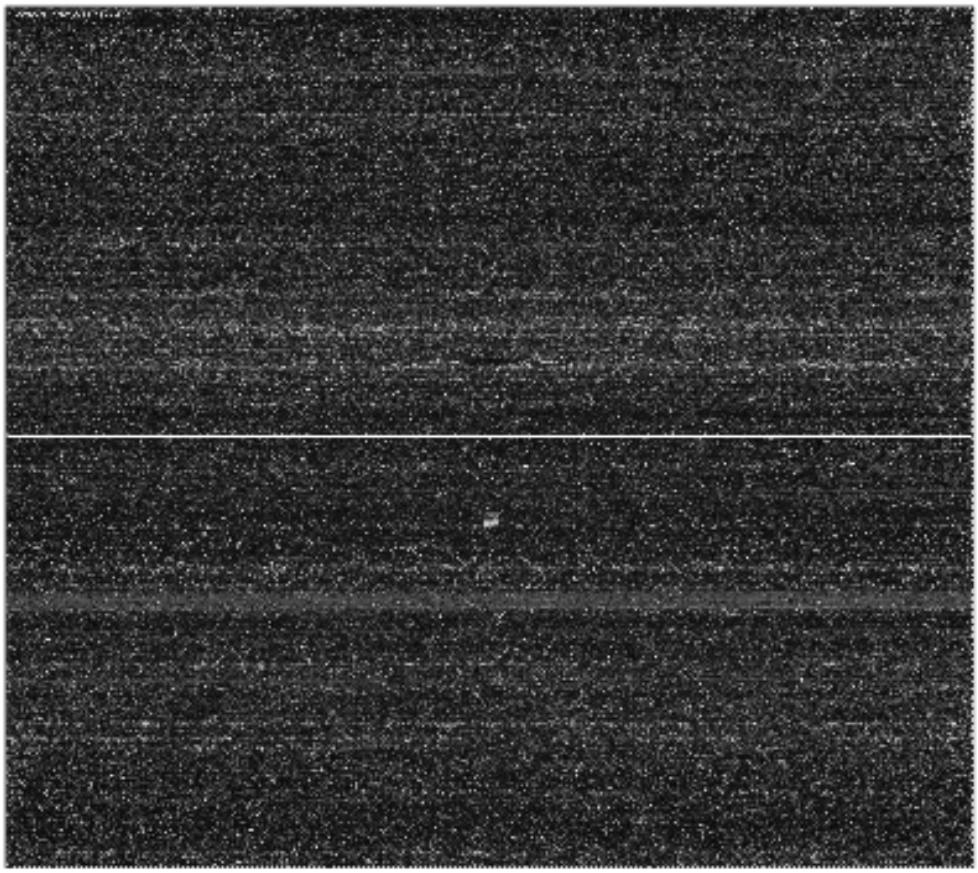


Figure 1: gse21779_1

```
counts = table(probeNames(gse21779raw))
table(counts)
```

```
counts
    8     9    10    11    13    14    15    16    20    69
    5     1     6 54130     4     4     2   482    40     1
```

Variabilidad intra y entre arrays

PM y MM: código

```
indexProbes(gse21779raw,"both",ID)[[1]]
```

```
 [1] 1088751  883561 1054203  366753  178855   68793  490689   62456 1123802
[10]  939573 1349651 1089915  884725 1055367  367917  180019   69957  491853
[19]   63620 1124966  940737 1350815
```

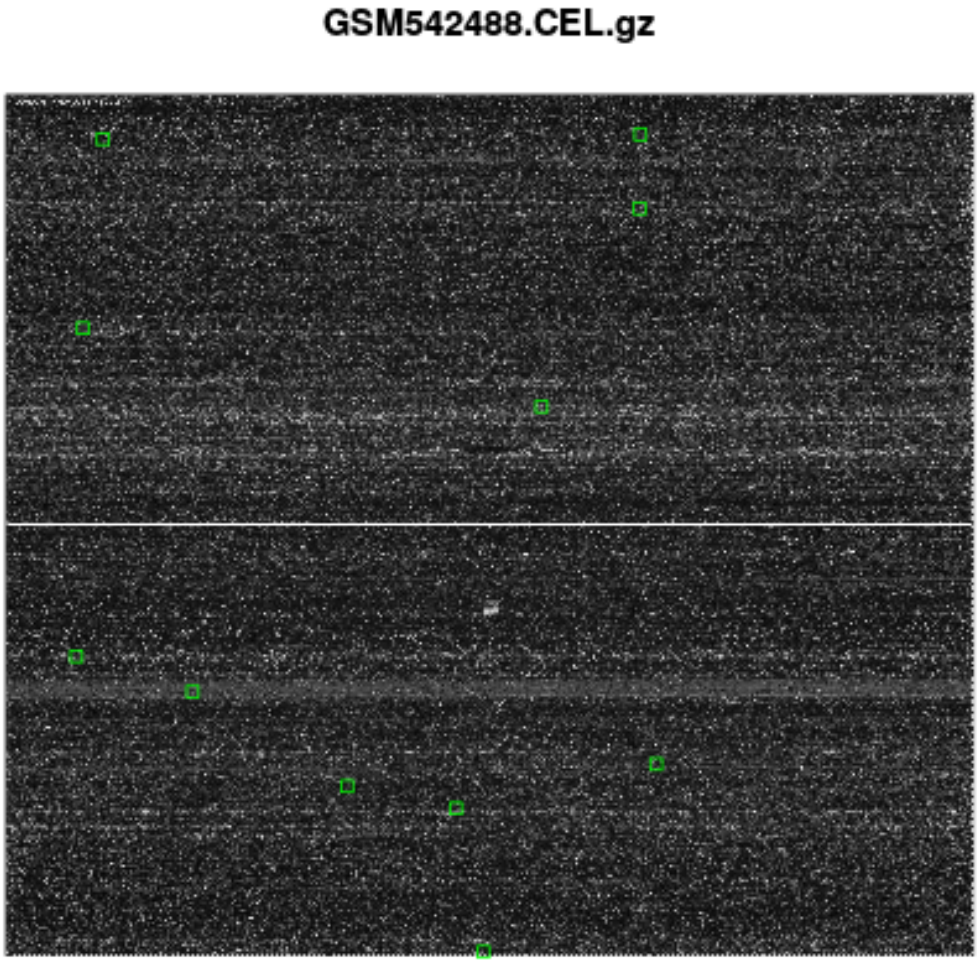
```
(posiciones = indexProbes(gse21779raw,"pm",ID)[[1]])
```

```
 [1] 1088751  883561 1054203  366753  178855   68793  490689   62456 1123802
[10]  939573 1349651
```

```
indexProbes(gse21779raw,"mm",ID)[[1]]
```

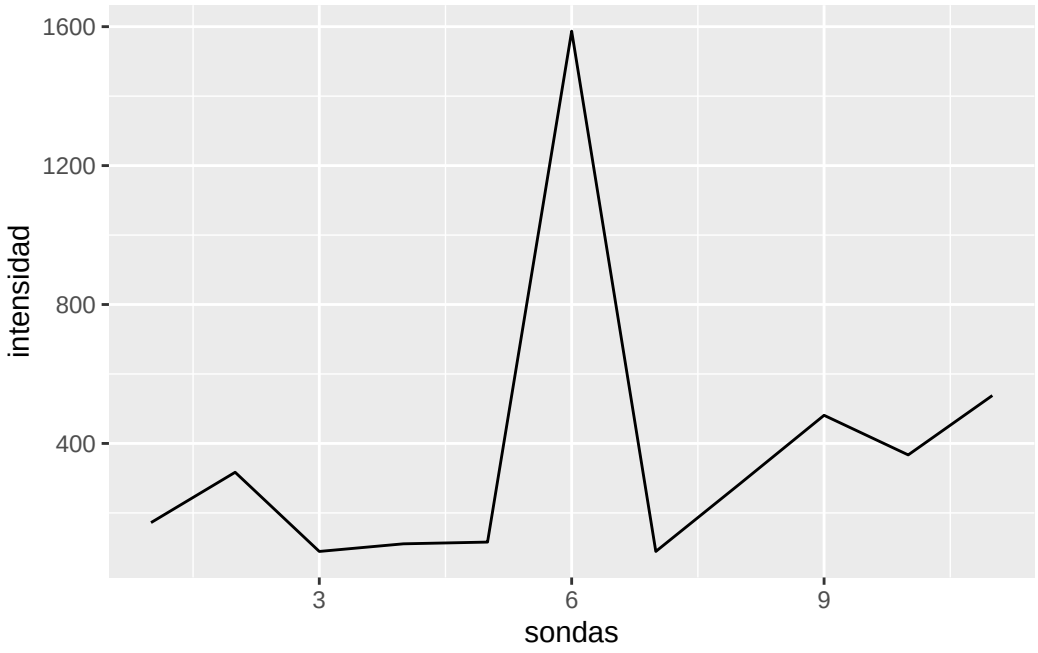
```
 [1] 1089915  884725 1055367  367917  180019   69957  491853   63620 1124966
[10]  940737 1350815
```

PM y MM: sondas



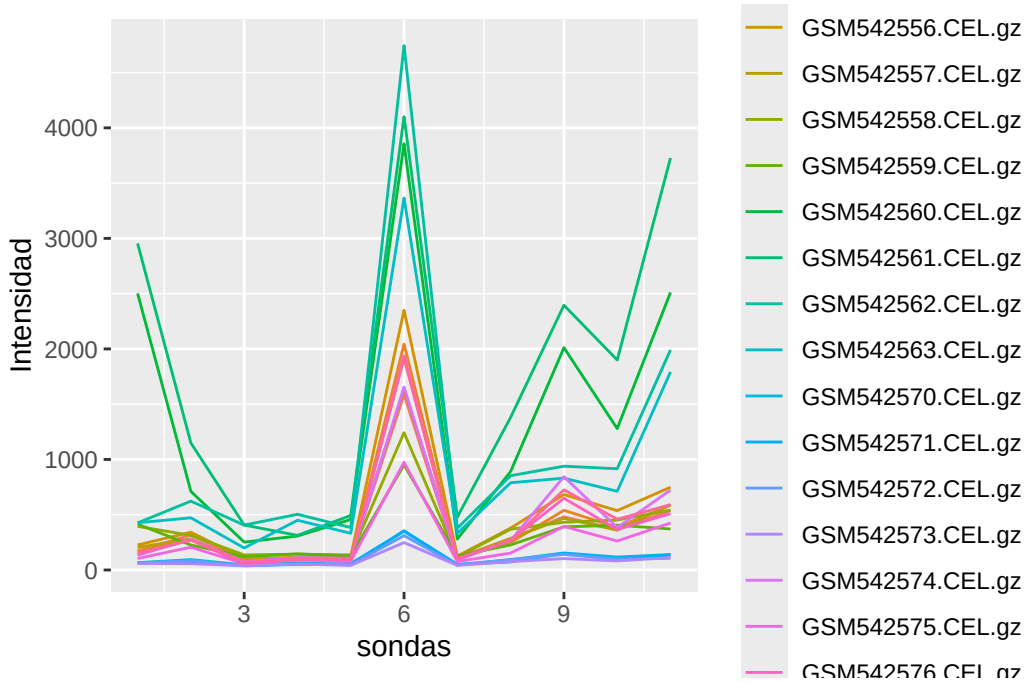
Intra

```
library(ggplot2)
pmID = probes(gse21779raw[,1],"pm",ID)
df1 = data.frame(sondas = 1:11,intensidad = pmID[,"GSM542488.CEL.gz"])
ggplot(df1,aes(x=sondas,y=intensidad))+geom_line()
```



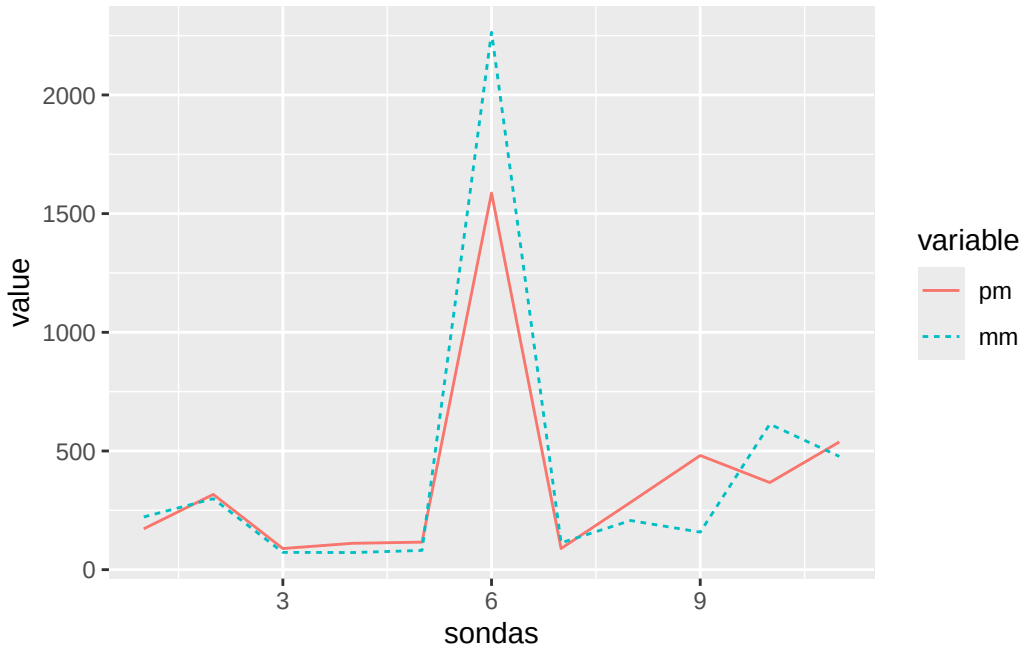
Entre

```
pm0 = probes(gse21779raw,"pm",ID)
pm0 = data.frame(pm0,sondas=1:11)
pm1 = reshape::melt(pm0,id="sondas")
ggplot2::ggplot(data=pm1,aes(x=sondas,y=value,colour=variable))+
    geom_line()+ylab("Intensidad")
```



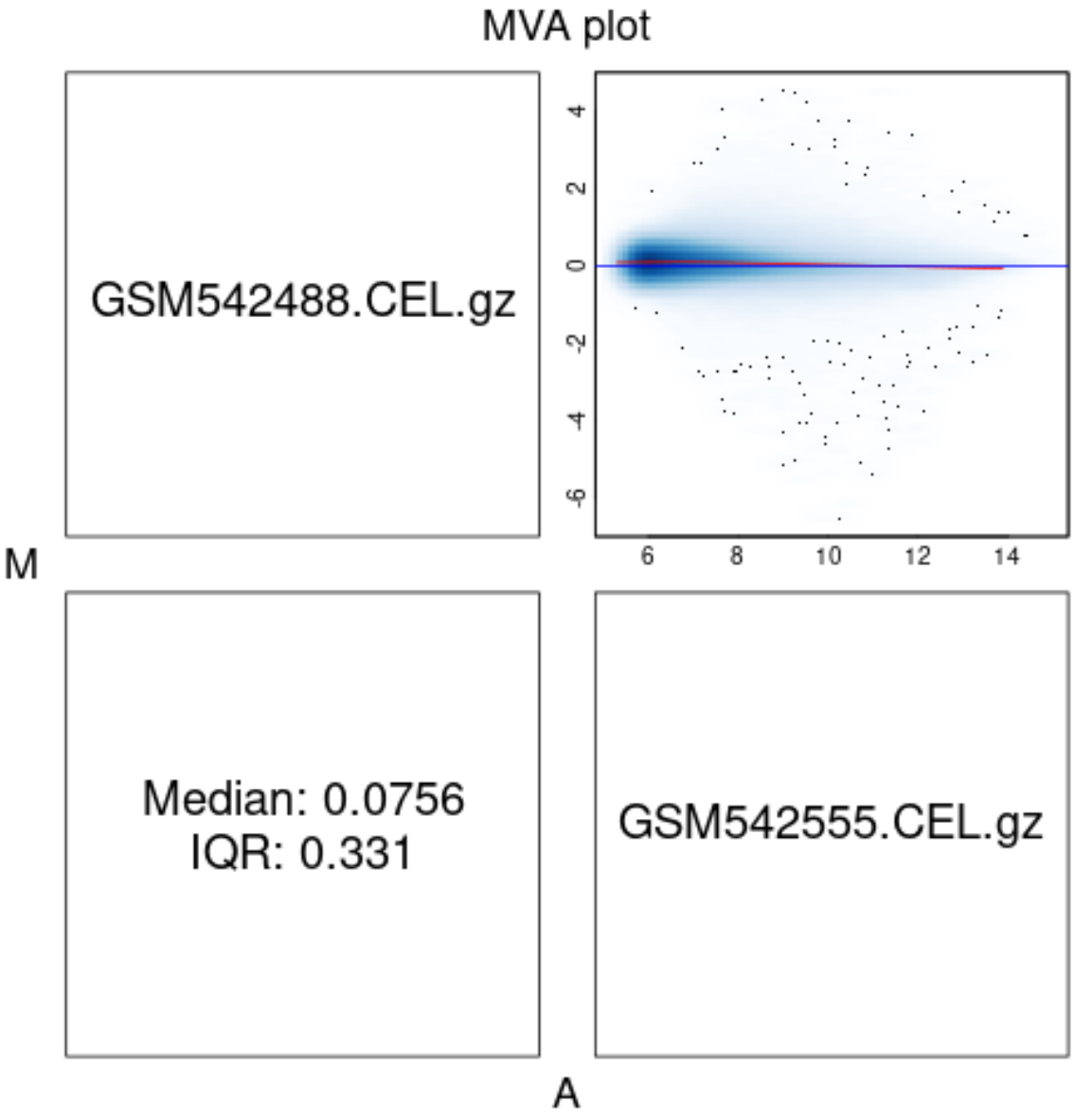
Comparando PM y MM

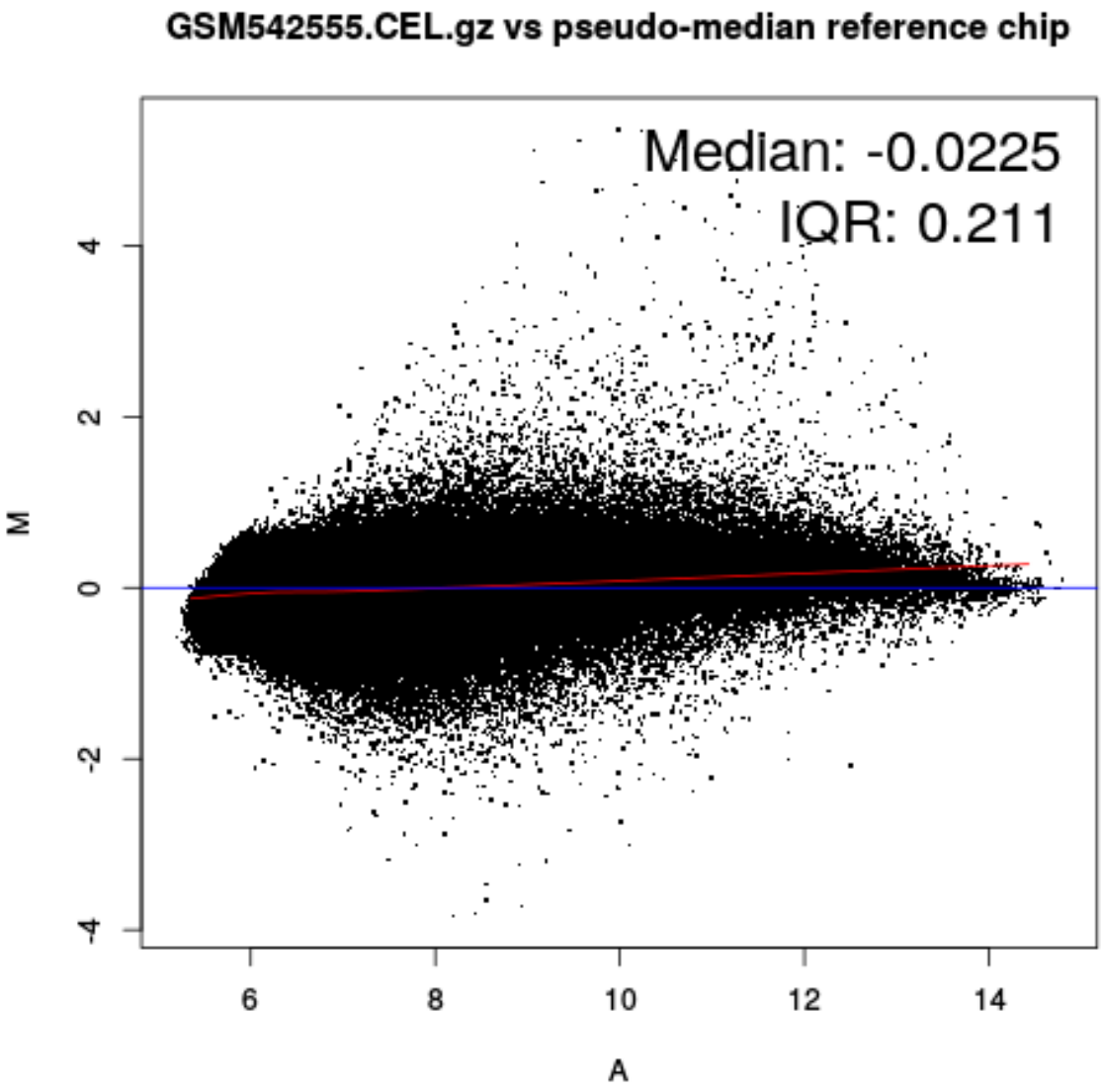
```
library(reshape)
pmm = data.frame(sondas=1:11,pm = probes(gse21779raw[,1],"pm",ID),
    mm = probes(gse21779raw[,1],"mm",ID))
df2 = reshape::melt(pmm,id="sondas")
levels(df2[,"variable"]) = c("pm","mm")
ggplot(df2,aes(x=sondas,y=value,colour=variable,linetype=variable))+geom_line()
```


2000
1500
1000
500
0
value
3
6
9
sondas
variable
pm
mm

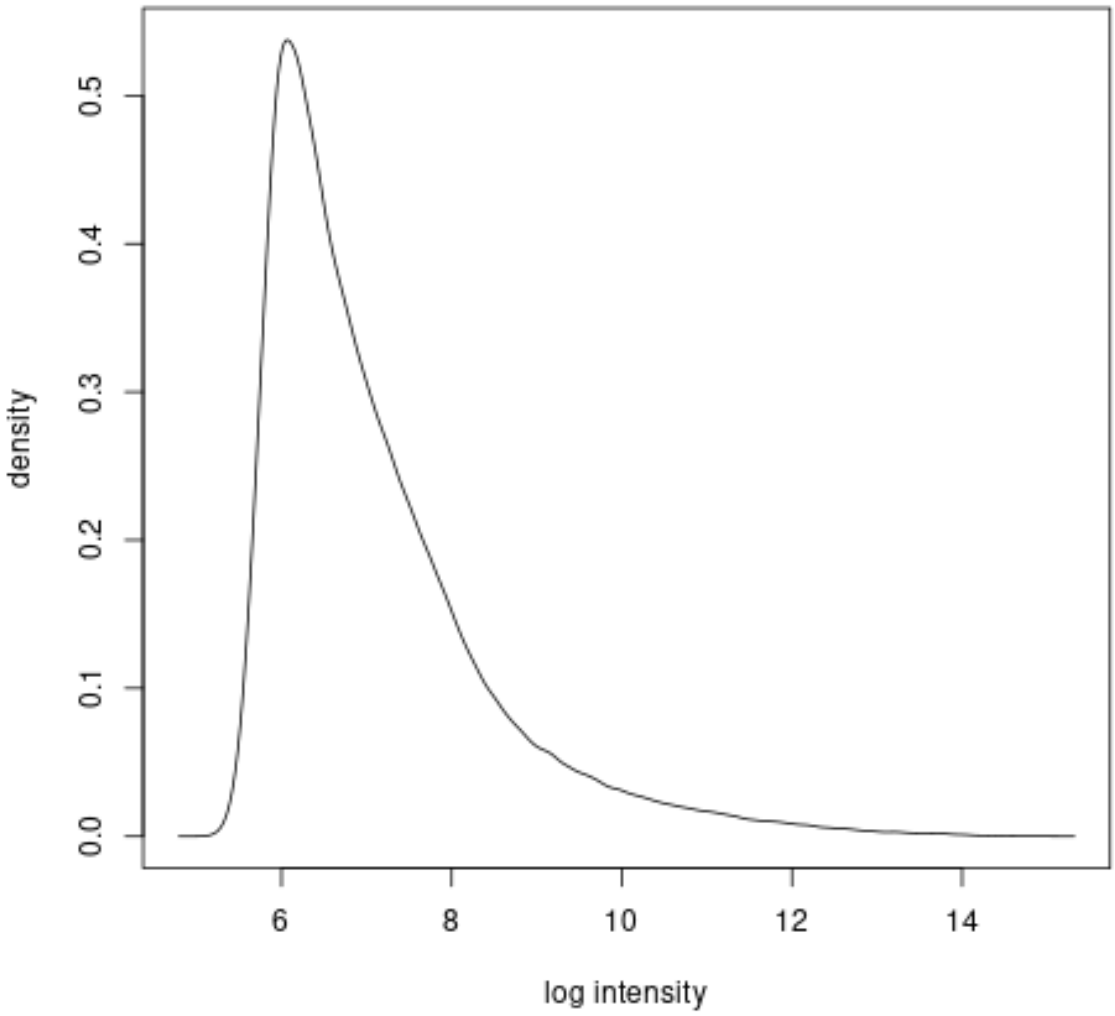
Control de calidad

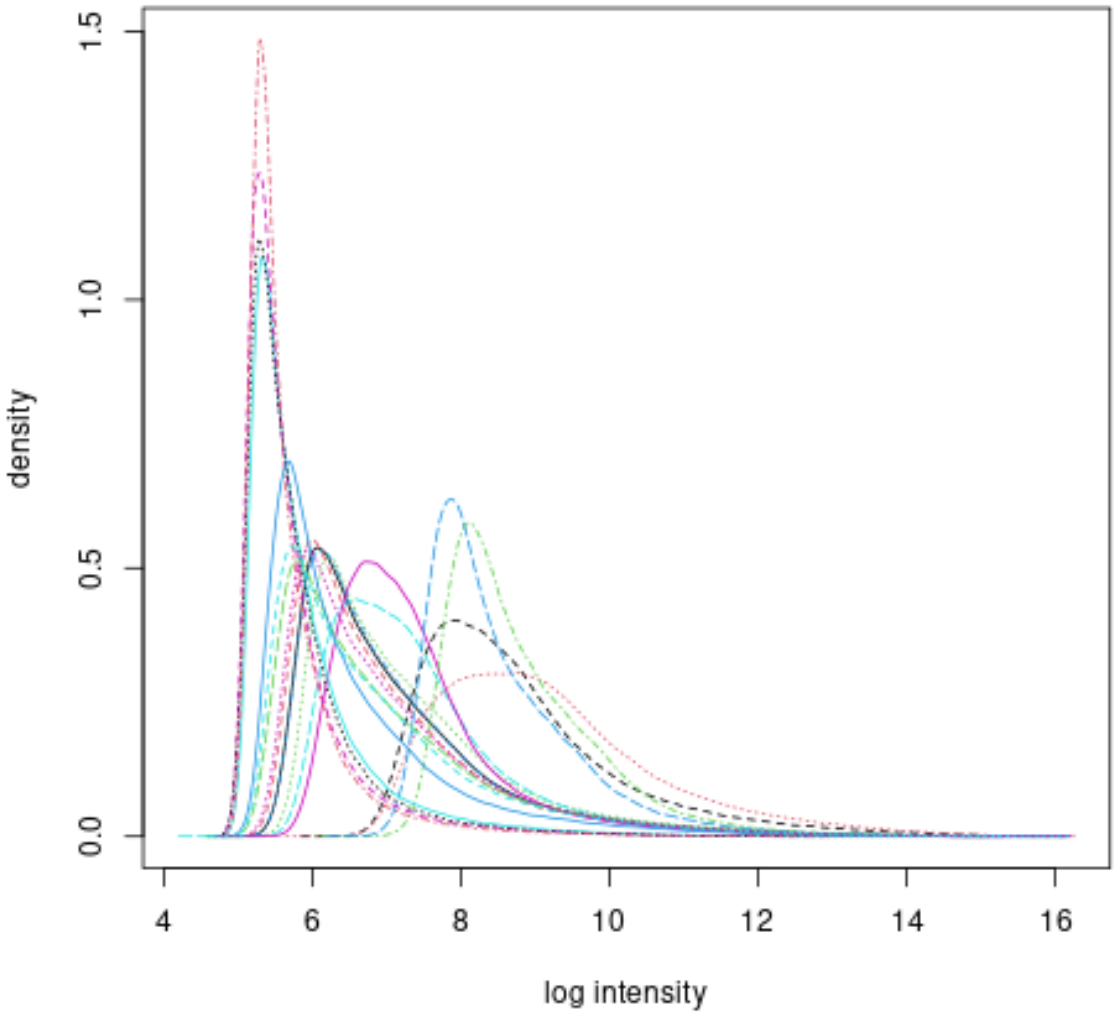
MA plot



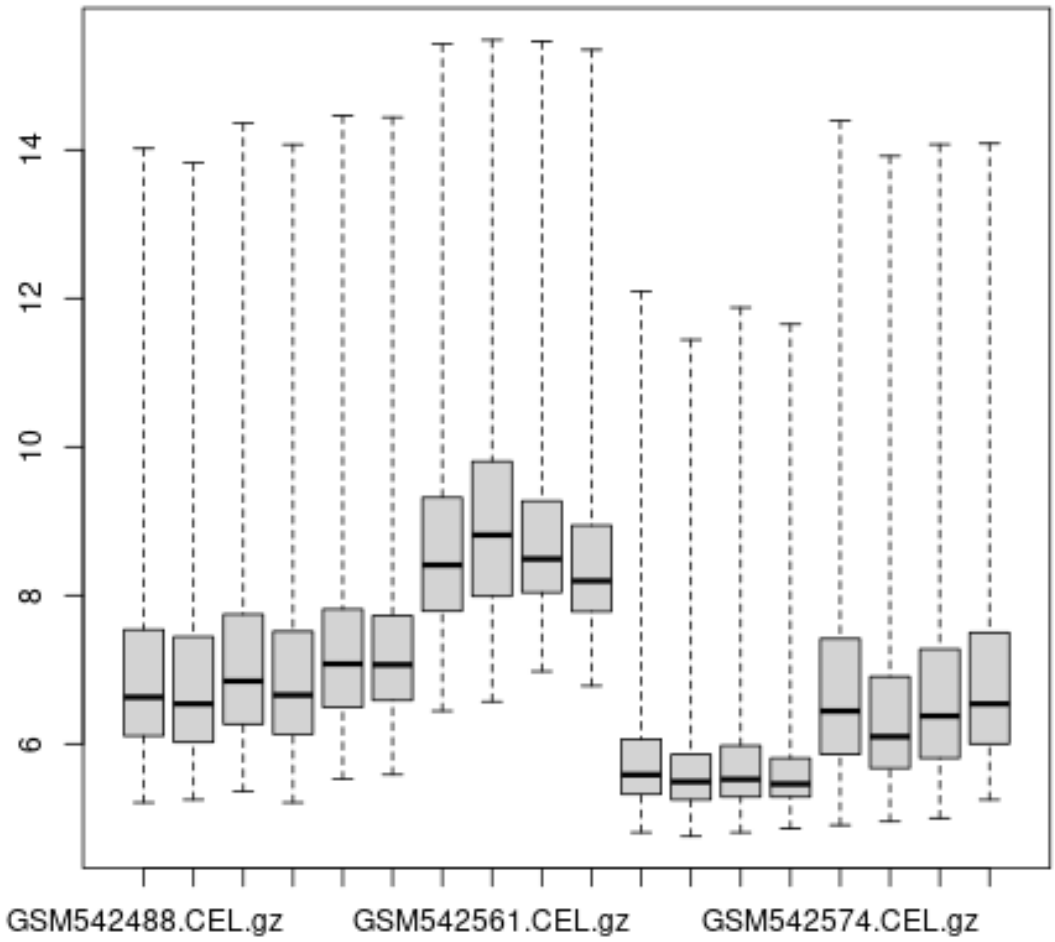

GSM542555.CEL.gz vs pseudo-median reference chip
Median: -0.0225
IQR: 0.211
M
A
4
2
0
-2
-4
6
8
10
12
14

Estimadores de densidad




density
0.0
0.5
1.0
1.5
4
6
8
10
12
14
16
log intensity

Diagramas de cajas



MAS5

¿Qué tenemos?

- En cada localización tendremos sondas que son oligonucleótidos específicos de un gen.

- De hecho son pares de sonda: una con la secuencia deseada y otra en la que se modifica la base en la posicion 13.
- La intensidad (o expresión) observada con el oligonucleótido correcto le llamamos PM (perfect matching).
- La expresión observado sobre el modificado le llamamos MM (miss matching).
- Una cierta proporción de localizaciones muestran un valor MM mayor que el valor PM.
- http://media.affymetrix.com/support/technical/whitepapers/sadd_whitepaper.pdf

Corrección de fondo

- Se divide el array en K zonas rectangulares: Z_k con $k = 1, \ldots, K$.
- En cada Z_k se ordenan las intensidades observadas en las distintas celdas y se considera el 2% con intensidades menores.
- Calculamos su media y desviación estándar muestrales: $b(Z_k)$ y $s(Z_k)$.
- Sea la celda localizada en (x, y).
- $d_k(x, y)$= la distancia desde (x, y) al centro de la región Z_k.
- Consideramos la siguiente cantidad

$$w_k(x, y) = \frac{1}{d_k^2(x, y) + s_0}$$

  donde s_0 nos evita que el cociente se pueda anular y por defecto $s_0 = 100$.
- Para la celda localizada en (x, y) consideramos

$$b(x, y) = \frac{1}{\sum_{k=1}^{K} w_k(x, y)} \sum_{k=1}^{K} w_k(x, y) b(Z_k)$$

Ruido local

- Vamos a restar a las intensidades una medida de ruido de fondo local.
- El problema es que nos encontremos con valores negativos.
- Este valor de ruido en la celda localizada en (x, y) se calcula como

$$n(x, y) = \frac{1}{\sum_{k=1}^{K} w_k(x, y)} \sum_{k=1}^{K} w_k(x, y) s(Z_k).$$

- La idea ahora es quitar a las intensidades originales el fondo ($b(x, y)$) que hemos calculado pero asegurándonos que no nos surja valores negativos.
- La intensidad original se denota como $I(x, y)$ en la localización (x, y). El valor ajustado de la intensidad viene dado por

$$A(x, y) = \max\{I(x, y) - b(x, y), f_0 n(x, y)\}$$

- El parámetro f_0 se suele fijar en $f_0 = 0.5$.

Cálculo del valor de expresión

- Por el procemiento descrito anteriormente podemos ajustar los valores de PM y de MM.
- En lo que sigue se supone que hemos ajustado estos valores.
- Lo esperable y correcto es que en una celda tengamos un valor de PM mayor que el correspondiente valor MM.
- Esto no es así en muchas ocasiones.
- Se define un valor IM (Ideal Mismatch).

Ideal Mismatch

- Denotamos: (PM_{ij}, MM_{ij}) el valor de PM y de MM de la j-ésima sonda perteneciente al i-ésimo conjunto de sondas.
- Se calcula un valor que llaman background específico para cada conjunto de sondas.

$$B_i = T_{bi}\{\log_2(PM_{ij}) - \log_2(MM_{ij}) : j = 1, \dots, n_i\},$$

donde T_{bi} denota el algoritmo bipesado en un solo paso.

- Si el valor de

$$IM_{ij} = \begin{cases} MM_{ij} & \text{si } MM_{ij} < PM_{ij} \\ \frac{PM_{ij}}{2^{B_i}} & \text{si } MM_{ij} \geq PM_{ij} \text{ y } B_i > \alpha \\ \frac{PM_{ij}}{2^{\frac{\alpha}{1+(\alpha - B_i)/\beta}}} & \text{si } MM_{ij} \geq PM_{ij} \text{ y } B_i \leq \alpha \end{cases}$$

- Por defecto: $\alpha = 0.03$ y $\beta = 10$.

Valor resumen

- Calculamos

$$V_{ij} = \max\{PM_{ij} - IM_{ij}, d\}$$

siendo por defecto $d = 2^{-20}$.
- Transformamos

$$\mathbb{V}_{ij} = \log_2(V_{ij})$$

con $j = 1, \dots, n_i$.
- Y volvemos a aplicar un estimador biponderado en un solo paso.

$$SignalLogValue_i = T_{bi}(\mathbb{V}_{i1}, \dots, \mathbb{V}_{in_i}).$$

- Fijamos una señal objetivo, Sc: por defecto, $Sc = 500$.
- Se calcula el siguiente factor de escala para el grupo de sondas i.

$$sf_i = \frac{Sc}{MediaAjustada\{2^{SignalLogValue_i}, 0.02, 0.98\}}$$

- **MediaAjustada** indica la media ajustada (media de los valores quitando una cierta proporción de los más grandes y de los más pequeños) en donde se quita el 2% de los más grandes y el 2% de los más pequeños.
- El valor final para el conjunto de sondas i es

$$ReportedValue(i) = sf_i \times 2^{SignalLogValue_i}$$

- Se incorpora una posibilidad de normalización que no indicamos.

Algoritmo biponderado de Tukey

- Tenemos unos datos $x_1, \dots, x_n$.
- Calculamos la mediana m_x de los datos $x_1, \dots, x_n$.
- Determinamos las distancias $d_i = |x_i - m_x|$.
- Calculamos la mediana m_d de d_i con $i = 1, \dots, n$ (MAD, median absolute deviation).
- Consideramos

$$u_i = \frac{x_i - m_x}{cm_d + \epsilon},$$

con $i = 1, \dots, n$. Los valores por defecto son $c = 5$ y $e = 0.0001$ (con el valor de e evitamos la división por cero).

- Consideremos una función bicuadrada (bisquare) definida como

$$w(u) = \begin{cases} (1-u^2)^2 & \text{si } |u| \leq 1 \\ 0 & \text{si } |u| > 1 \end{cases}$$

- Se define el estimador en un solo paso como

$$T_{bi}(x_1, \dots, x_n) = \frac{\sum_{i=1}^n w(u_i)x_i}{\sum_{i=1}^n w(u_i)}.$$

- Si sustituimos el valor m_x por T_b podríamos aplicar iterativamente el procedimiento hasta que se estabilice.

Robust Multichip Average (RMA)

RMA

- En este método trabajamos con todos los arrays simultáneamente.
- A diferencia del método anterior donde se procesaba cada array independientemente.
- El procedimiento solamente utiliza los valores PM y no utiliza para nada los valores MM.

Corrección de fondo

- El valor observado aleatorio S sería

$$S = X + Y$$

con $X \sim Exp(\alpha)$ e $Y \sim N(\mu, \sigma^2)$.

- Se tiene

$$E(X|S = s) = a + b\frac{\phi(\frac{a}{b})}{\Phi(\frac{a}{b})}$$

siendo $a = s - \mu - \sigma^2\alpha$ y $b = \sigma$ mientras que
ϕ y Φ denotan las funciones de densidad y de distribución de una normal estándar (con media 0 y varianza 1).

- Los parámetros del modelo (α, μ, σ^2) se estiman mediante un procedimiento no paramétrico.

Normalización de cuantiles

- Tomamos n vectores de longitud N y construimos la matriz X
$N \times n$ que los tiene por vectores columna.
- Ordenamos cada columna de X separadamente y tenemos X_s.
- Calculamos la media por filas de la matrix X_s y creamos X'_s una matriz con la misma dimensión que X, tal que en la fila j tenemos la media de la fila de X_s repetida n veces.
- Obtenemos X_t en donde cada columna de X'_s recupera el orden original.

Resumen

- En cada array tendremos N sondas.
- Suponemos que tenemos n arrays.
- La matriz de expresión a nivel de sonda será:

$$\mathbf{x} = [x_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$$

- La sonda i-ésima en el array j-ésimo tiene una expresión o intensidad x_{ij}.
- Denotamos el grupo k-ésimo de sondas con S_k.
- $S_k \subset \{1, \dots, N\}$
- $S_k = \{i_1, \dots, i_{|S_k|}\}$
- Denotamos

$$\mathbf{y} = [y_{ij}]_{i=1,\dots,|S_k|, j=1,\dots,n} = [x_{ij}]_{i \in S_k, j=1,\dots,n}.$$

Median polish

- Consideramos la matriz para **cada grupo de sondas**.

$$\begin{array}{ccc|c} \delta_{1,1} & \cdots & \delta_{1,n} & a_1 \\ \vdots & \ddots & \vdots & \vdots \\ \delta_{|S_k|,1} & \cdots & \delta_{|S_k|,n} & a_{|S_k|} \\ \hline b_1 & \cdots & b_n & m \end{array}$$

- Fijamos $\delta_{ij} = \log_2(y_{ij})$, $a_i = b_i = 0 \forall i, j$.
- Calculamos la mediana para cada fila a lo largo de las columnas ignorando la última columna (separada por la línea).
- Restamos a cada elemento de la fila la mediana correspondiente. Esta mediana se suma a la última columna (formada por $a_1, \dots, a_{|S_k|}, m$.
- Calculamos la mediana para cada columna de los valores correspondientes a las distintas filas sin considerar la última fila.
- Restamos la mediana calculada en el paso anterior a cada elemento de la columna exceptuando la última fila. A la última fila sumamos las medianas calculadas.
- El proceso descrito en los cuatro pasos anteriores continua iterando sucesivamente entre filas y columnas hasta que los cambios que se producen son nulos o muy pequeños.

Estimaciones

Una vez finalizado el proceso iterativo tendremos:

- $\hat{\mu} = m$
- $\hat{\theta}_j = b_j$
- $\hat{\alpha}_i = a_i$
- $\hat{\epsilon}_{ij} = \delta_{ij}$

- ¿Y cómo estimamos la expresión de un grupo de sondas en una array?

$$\hat{\mu} + \hat{\theta}_j$$

- **Y esto lo repetimos para cada grupo de sondas.**

affy

MAS5 y RMA

```
library(affy)
```

- El método **MAS5**

```
gse21779_mas5 = mas5(gse21779raw)
```

- El método **RMA**

```
gse21779_rma = rma(gse21779raw)
```

Densidades: MAS5

```
library(geneplotter)
```

```
Cargando paquete requerido: lattice
```

```
Cargando paquete requerido: annotate
```

```
Cargando paquete requerido: AnnotationDbi
```

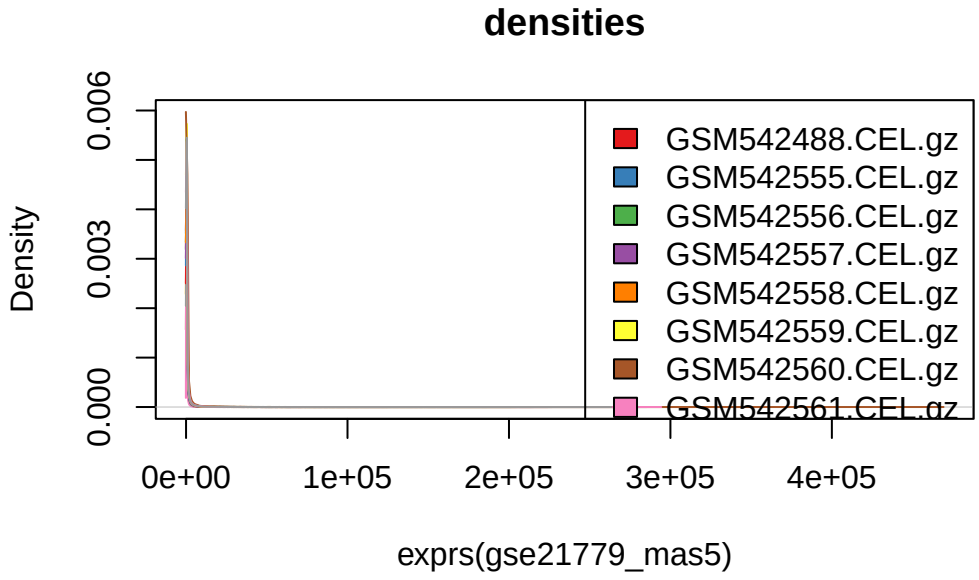
```
Cargando paquete requerido: stats4
```

```
Cargando paquete requerido: IRanges
```

```
Cargando paquete requerido: S4Vectors
```

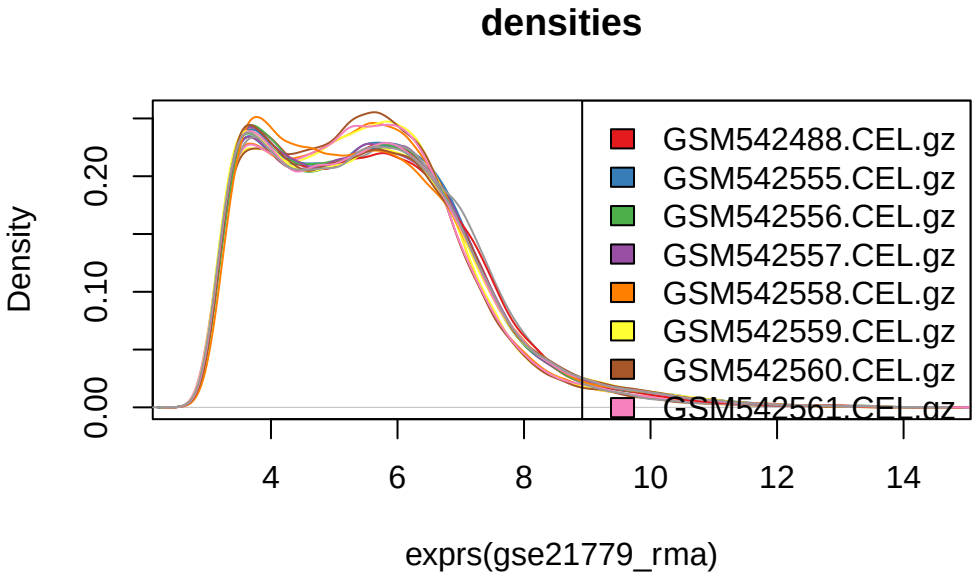
```
Adjuntando el paquete: 'S4Vectors'

The following objects are masked from 'package:reshape':

    expand, rename

The following object is masked from 'package:utils':

    findMatches

The following objects are masked from 'package:base':

    expand.grid, I, unname

Cargando paquete requerido: XML
```

```
geneplotter::multidensity(exprs(gse21779_mas5))
```



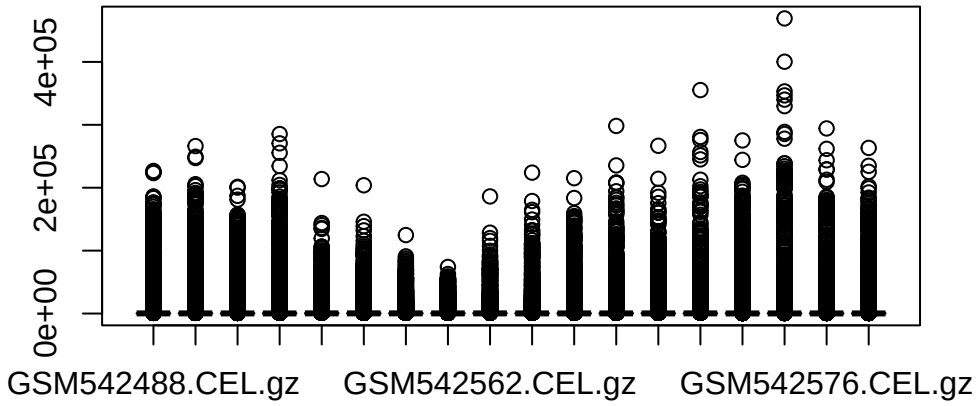
Densidades: RMA

```
geneplotter::multidensity(exprs(gse21779_rma))
```



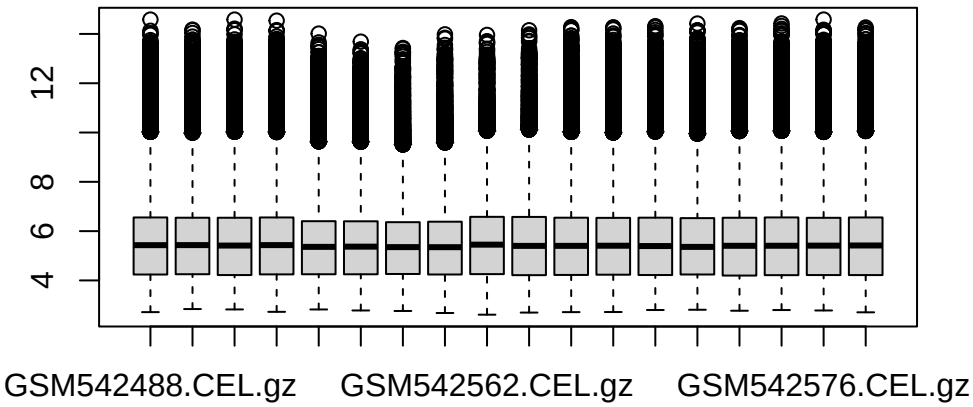
Diagramas de cajas: MAS5

```
graphics::boxplot.matrix(exprs(gse21779_mas5))
```



Diagramas de cajas: RMA

```
graphics::boxplot.matrix(exprs(gse21779_rma))
```



ExpressionSet

```
pacman::p_load(ALL)
```

```
data(ALL)
```

La clase del objeto la podemos ver con

```
class(ALL)
```

```
[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"
```

El método `print` nos da un breve resumen de la clase.

```
ALL
```

La matriz de expresión la obtenemos con el accessor `Biobase::exprs`.

```
             01005     01010    03002    04006    04007     04008     04010
1000_at   7.597323  7.479445 7.567593 7.384684 7.905312  7.065914  7.474537
1001_at   5.046194  4.932537 4.799294 4.922627 4.844565  5.147762  5.122518
1002_f_at 3.900466  4.208155 3.886169 4.206798 3.416923  3.945869  4.150506
1003_s_at 5.903856  6.169024 5.860459 6.116890 5.687997  6.208061  6.292713
1004_at   5.925260  5.912780 5.893209 6.170245 5.615210  5.923487  6.046607
1005_at   8.570990 10.428299 9.616713 9.937155 9.983809 10.063484 10.662059
             04016    06002     08001    08011    08012    08018    08024
```

```
1000_at    7.536119 7.183331  7.735545 7.591498 7.824284 7.231814 7.879988
1001_at    5.016132 5.288943  4.633217 4.583148 4.685951 5.059300 4.830464
1002_f_at  3.576360 3.900935  3.630190 3.609112 3.902139 3.804705 3.862914
1003_s_at  5.665991 5.842326  5.875375 5.733157 5.762857 5.770914 6.079410
1004_at    5.738218 5.994515  5.748350 5.922568 5.679899 6.044520 6.057632
1005_at   11.269115 8.812869 10.165159 9.381072 8.227970 7.627248 7.667445
              09008     09017    11005    12006    12007    12012     12019
1000_at    7.891793  7.756734 7.640012 7.759599 7.678636 7.464285  7.652719
1001_at    5.999496  4.987595 4.967288 4.770481 5.456332 4.785863  5.175609
1002_f_at  4.001606  4.048901 3.796550 3.912707 3.870893 3.930832  3.932360
1003_s_at  5.832952  6.097900 6.094379 6.235795 5.971466 6.037364  6.194623
1004_at    5.717497  6.210092 5.751805 5.883340 5.918456 5.725421  5.969027
1005_at   10.206353 10.015466 9.358516 8.824348 9.262478 7.232927 10.243610
             12026    14016     15001    15004    15005    16004    16009
1000_at   7.501591 7.570417  7.331509 7.366208 7.455451 7.328875 7.297313
1001_at   5.188992 5.258312  4.627955 4.733495 5.125098 5.332775 5.215707
1002_f_at 4.188444 4.028859  4.099497 3.831617 3.786741 4.648154 4.176687
1003_s_at 6.231228 6.348593  6.057790 6.043205 6.303592 5.905220 6.274415
1004_at   6.357476 6.173530  5.729543 5.671164 5.977084 5.721572 6.000902
1005_at   7.808452 7.557919 10.233185 9.749860 8.477219 9.884608 9.599149
             19005    20002    22009    22010    22011    22013    24001
1000_at   7.563561 7.541133 8.016818 7.862181 7.702580 7.412003 7.916169
1001_at   4.858392 4.964424 5.216252 5.135825 4.802946 5.222676 4.790170
1002_f_at 4.097332 3.789888 3.980839 3.954917 3.971934 4.109899 3.899038
1003_s_at 5.838829 6.160238 6.343042 6.195307 5.865581 6.243157 6.022905
1004_at   5.970021 6.054232 6.024327 6.114502 6.035582 5.896131 5.800271
1005_at   9.683996 9.221405 8.571812 8.841628 8.489550 8.998592 8.933302
             24005     24008    24010    24011     24017    24018     24019
1000_at   7.595848  7.296349 7.506236 7.144425  7.513972 7.815971  7.406135
1001_at   4.804743  5.002518 4.218220 5.228892  5.264158 4.899316  4.791335
1002_f_at 3.677240  3.906343 3.579385 3.829513  3.965467 4.058576  4.131234
1003_s_at 5.669771  5.668509 5.273965 5.817272  6.088179 6.387995  6.343673
1004_at   5.475892  5.437155 4.634124 5.552223  5.982065 5.874817  6.048209
1005_at   9.626267 10.826157 8.782330 7.881855 11.069535 9.102971 10.037809
             24022    25003     25006    26001    26003    26005    26008
1000_at   7.300980 7.845054  7.651229 7.376930 7.663977 7.250353 7.663612
1001_at   5.177703 5.250315  4.896195 5.123546 5.078104 4.945670 5.124591
1002_f_at 3.838198 4.046442  4.120495 4.131492 3.803233 4.105035 4.119660
1003_s_at 5.863318 6.205917  6.298788 6.118064 6.199316 5.782105 6.271931
1004_at   5.669051 5.931859  5.915944 6.002812 5.822230 5.930979 6.073099
1005_at   9.095296 8.670866 10.496309 9.046483 9.104846 9.756972 9.071812
             27003    27004    28001    28003    28005    28006     28007
1000_at   7.329996 7.360754 7.035203 7.705260 7.551734 7.538601  7.501531
```

```
1001_at   5.438098 4.757900 5.005279 5.009705 4.944978 4.511194  4.888814
1002_f_at 3.677207 3.638739 3.800893 4.238111 3.719482 3.788262  3.626149
1003_s_at 5.899308 5.664813 5.732956 6.379139 5.833428 5.362676  5.702537
1004_at   5.718582 5.595820 5.485361 5.881472 5.554058 4.986320  5.743743
1005_at   6.969776 8.867644 7.067019 9.080559 8.245585 7.807180 10.085771
             28019    28021    28023    28024    28028     28031    28032
1000_at   7.116676 7.107979 7.427808 6.549926 7.514761  7.377215 6.973861
1001_at   5.275964 4.865566 5.057619 5.185277 4.788468  4.778381 4.970430
1002_f_at 4.192648 3.979372 3.791415 3.943834 3.924333  3.657005 3.706332
1003_s_at 6.196541 5.804445 5.719376 5.943116 5.719659  5.939648 5.968072
1004_at   5.926093 5.768851 5.478333 5.756534 5.740607  5.770578 5.905965
1005_at   8.097072 8.661098 9.106441 8.804075 8.235771 10.607653 6.760202
             28035    28036    28037    28042    28043    28044    28047
1000_at   7.227516 7.407561 7.158049 7.235291 7.589310 7.988476 7.362458
1001_at   6.408157 5.042222 5.431469 4.686293 4.851805 4.894379 4.843868
1002_f_at 3.995074 3.714084 4.302001 3.677909 3.831514 3.690856 3.646990
1003_s_at 6.272305 5.733332 6.253362 6.098969 6.132159 6.130691 5.628370
1004_at   6.050495 5.651345 6.494545 6.109255 5.867806 5.592139 5.644372
1005_at   8.957027 8.764321 9.102879 7.008132 7.720932 7.861043 6.642728
             30001    31007    31011     33005     36001    36002    37013
1000_at   7.508667 7.147843 7.651676  7.486432  7.759074 7.473427 7.627685
1001_at   5.587029 4.943857 4.741654  4.642628  4.962544 4.953122 5.358236
1002_f_at 3.765444 3.789166 3.790688  3.682768  3.740679 3.688162 4.008891
1003_s_at 6.078532 5.858604 6.251328  5.961910  5.936883 5.642185 6.314849
1004_at   5.935291 5.526191 5.820374  5.810047  5.616831 5.678327 6.044299
1005_at   9.075910 9.588264 9.585180 11.609025 11.653732 8.893931 7.950866
             43001     43004    43007    43012     48001    49006     57001
1000_at   7.577529  7.600206 7.776844 7.585928  7.450666 7.004613  7.195206
1001_at   5.054157  4.879037 4.949908 5.057530  4.960382 4.836905  4.744006
1002_f_at 3.932435  4.028704 3.689141 3.891536  4.061201 3.699625  3.973128
1003_s_at 6.310934  6.086349 5.658127 6.363734  6.099140 5.616555  5.962672
1004_at   5.782177  5.817414 5.621938 5.975024  5.853644 5.704549  5.765632
1005_at   8.569400 10.693175 7.601647 9.377819 10.929231 9.193089 10.691061
             62001    62002    62003    63001    64001     64002    65005
1000_at   7.407351 7.756195 7.913324 7.270997 7.694588  7.583071 7.609538
1001_at   4.930312 5.238937 5.074681 4.513671 4.928159  4.804083 4.715693
1002_f_at 3.734818 3.945514 3.926906 3.639640 3.806746  4.104208 3.453649
1003_s_at 5.730142 6.061704 6.208286 5.519846 5.834263  6.340025 5.584102
1004_at   5.512776 5.956554 6.028228 5.041052 5.912557  6.056120 5.611407
1005_at   9.108345 9.507559 9.693530 7.758929 8.883131 10.320243 7.757368
             68001    68003    84004     LAL5    01003    01007    02020
1000_at    7.324502 7.545120 7.679603 7.604093 7.240252 7.676749 7.934247
1001_at    5.379102 4.650231 4.795495 4.988922 5.224752 5.129002 5.667907
```

```
1002_f_at  4.066075 3.626514 3.554142 3.960894 3.862458 4.169622 4.173444
1003_s_at  6.121059 6.347044 5.471594 5.761373 6.430906 6.202916 6.083160
1004_at    6.224473 5.884682 5.505538 6.048169 5.940335 5.979690 6.127251
1005_at   11.165801 8.986872 9.984865 9.741136 9.790736 8.671713 8.308241
             04018    09002    10005    11002    12008    15006    16002
1000_at   7.874448 7.404271 7.775253 7.771891 7.355677 7.388882 7.589734
1001_at   5.005420 5.127949 4.423445 4.476761 5.461252 5.330129 4.836986
1002_f_at 3.772763 3.979145 3.529124 3.656052 4.121548 4.146230 3.956283
1003_s_at 6.041962 6.013443 5.156087 5.528908 6.054917 6.028770 5.990144
1004_at   5.751041 6.138876 5.038015 5.396687 5.984467 6.282975 5.860502
1005_at   8.324171 6.379843 8.841847 7.536542 8.685069 9.657075 8.481350
             16007     17003    18001    19002    19008    19014     19017
1000_at   7.675929  7.662426 7.584008 7.840099 7.164922 7.843162  7.695714
1001_at   4.959669  5.743215 4.674920 5.208166 4.554529 5.718569  4.498515
1002_f_at 3.819324  4.142183 3.556962 3.942920 3.541456 4.000303  3.667304
1003_s_at 5.763215  6.307570 5.825432 6.181242 5.869161 6.131560  5.787218
1004_at   5.781565  5.857727 5.786847 5.935438 5.628791 5.613913  5.226277
1005_at   9.314393 10.336505 9.238966 8.941103 8.005422 9.629751 10.907283
             20005    24006    26009    28008    28009    31015     37001
1000_at   7.520867 7.836577 7.470524 7.520806 7.646947 7.727560  7.849455
1001_at   5.135697 5.129836 5.213340 4.690815 4.902946 4.866731  4.959450
1002_f_at 4.165152 4.002384 4.126231 3.655414 3.894576 3.970375  4.047428
1003_s_at 6.019615 6.110763 6.271189 5.513344 5.814061 5.779816  6.193790
1004_at   5.850529 5.847115 6.089797 5.211162 5.828099 5.738370  5.824930
1005_at   8.034295 7.967475 9.382400 7.727615 8.067144 9.211820 10.408645
             43006    43015    44001    49004    56007     64005    65003
1000_at   7.960842 8.188617 7.399999 7.813474 7.816922  7.913249 7.800199
1001_at   4.537677 5.154500 5.071885 4.874525 4.788699  5.403640 5.443827
1002_f_at 3.694877 3.949546 3.837517 3.767672 4.248075  4.119252 3.957468
1003_s_at 5.973162 6.033704 5.560886 5.930329 5.878533  6.193376 6.006443
1004_at   5.841095 5.870163 5.686555 5.793403 5.913503  6.053053 5.549087
1005_at   7.524865 8.381740 8.055760 7.234242 8.083834 10.163355 8.184442
             83001     LAL4
1000_at   8.030047 7.702217
1001_at   5.178633 5.029670
1002_f_at 4.053458 4.002306
1003_s_at 6.050129 6.297549
1004_at   5.932412 5.827131
1005_at   8.798521 7.687370
```

Los datos fenotípicos los tendremos con `Biobase::pData`.

```
pData(ALL)
```

¿Cuál fue chip que se utilizó para obtener estos datos?

```
annotation(ALL)
```

```
[1] "hgu95av2"
```

Los identificadores de las filas los tenemos con

```
featureNames(ALL)
```

Construyendo un ExpressionSet (Neve 2006)

Lo necesario

- Tenemos los datos de expresión en un fichero.
- En **otro** fichero tenemos información sobre los genes que tenemos en las filas de la matriz de expresión.
- Tenemos información sobre las distintas muestras que componen toda la experimentación.
- Y finalmente sabemos cosas sobre toda la experimentación realizada: autores, publicación principal que se derivó, objetivo de la experimentación.
- Y todo por separado, en distintos ficheros.
- ¿Cómo construimos el **ExpressionSet**?

Lectura datos

- Cargamos Biobase.

```
library(Biobase)
```

- Leemos datos de expresión.

```
library(tamidata)
finput = system.file("extdata","n06_expr.txt",package="tamidata")
exprs0 = read.table(file = finput,header = TRUE,sep="\t")
```

- Tiene que ser una matriz, `matrix`.

```
exprs0 = as.matrix(exprs0[,-1])
```

- Como matriz que es tendrá `rownames` y `colnames`.

```
colnames(exprs0)
```

```
 [1] "X600MPE"    "AU565"      "BT474"      "BT483"      "CAMA1"
 [6] "DU4475"     "HCC202"     "HCC1428"    "HCC1007"    "HCC2185"
[11] "LY2"        "MCF7"       "MDAMB415"   "MDAMB175"   "MDAMB361"
[16] "MDAMB435"   "MDAMB134"   "SUM185PE"   "SUM225CWN"  "SUM44PE"
[21] "SKBR3"      "T47D"       "UACC812"    "ZR75B"      "ZR751"
[26] "ZR7530"     "SUM52PE"    "BT20"       "HCC1937"    "HCC70"
[31] "HCC1187"    "HCC1008"    "HCC1599"    "HCC3153"    "HCC1569"
[36] "HCC2157"    "HCC1954"    "HCC38"      "HCC1500"    "HCC1143"
[41] "MCF12A"     "MCF10A"     "MDAMB468"   "SUM149PT"   "SUM190PT"
[46] "BT549"      "HS578T"     "HBL100"     "MDAMB231"   "MDAMB435_2"
[51] "MDAMB157"   "MDAMB436"   "SUM159PT"   "SUM1315"
```

```
rownames(exprs0)
```

```
NULL
```

- Como vemos está vacío el elemento `rownames`.

...

- Leemos variables fenotípicas.

```
finput = system.file("extdata","n06_sample_attributes.txt",package="tamidata")
pd0 = read.table(file = finput,header = TRUE,sep="\t")
```

- Podemos ver el número de muestras (que ha de coincidir con el número de columnas de la matriz de expresión) y de covariables.

```
dim(pd0)
```

```
[1] 54 14
```

```
dim(exprs0)
```

```
[1] 22215    54
```

- Es **necesario** que los nombres de las filas de **pd0** coincidan con los nombres de las columnas de **exprs0**.

```
rownames(pd0) = colnames(exprs0)
```

- Etiquetamos los metadatos: la columna `labelDescription` debe de estar presente.

```
metadatos = data.frame(labelDescription = c("Identificador",
"Número de muestra", "Tipo","En CGH","ALL","Método CDH1","SFRP1",
"HER2","KIT","Presencia de p53","BRCA","Orden","Adherencia",
"per3"),
row.names=colnames(pd0))
```

- Reunimos etiquetas de metadatos y metadatos en un **AnnotatedDataFrame**.

```
datosfenotipo = new("AnnotatedDataFrame", data = pd0,
    varMetadata = metadatos)
```

- Una vez tenemos este `AnnotatedDataFrame` podemos ver los nombres de las muestras.

```
sampleNames(datosfenotipo)
```

```
 [1] "X600MPE"    "AU565"      "BT474"      "BT483"      "CAMA1"
 [6] "DU4475"     "HCC202"     "HCC1428"    "HCC1007"    "HCC2185"
[11] "LY2"        "MCF7"       "MDAMB415"   "MDAMB175"   "MDAMB361"
[16] "MDAMB435"   "MDAMB134"   "SUM185PE"   "SUM225CWN"  "SUM44PE"
[21] "SKBR3"      "T47D"       "UACC812"    "ZR75B"      "ZR751"
[26] "ZR7530"     "SUM52PE"    "BT20"       "HCC1937"    "HCC70"
[31] "HCC1187"    "HCC1008"    "HCC1599"    "HCC3153"    "HCC1569"
[36] "HCC2157"    "HCC1954"    "HCC38"      "HCC1500"    "HCC1143"
```

```
[41] "MCF12A"     "MCF10A"     "MDAMB468"   "SUM149PT"   "SUM190PT"
[46] "BT549"      "HS578T"     "HBL100"     "MDAMB231"   "MDAMB435_2"
[51] "MDAMB157"   "MDAMB436"   "SUM159PT"   "SUM1315"
```

O también ver los valores de las covariables.

```
head(pData(datosfenotipo))
```

```
        IDENTIFIER sample_no Type in_CGH ALL CDH1_meth   SFRP1 HER2 KIT
X600MPE     600MPE         1    L      1   1        no 3.40987 <NA>  NO
AU565        AU565         2    L      1   1      <NA> 3.16902 <NA>  NO
BT474        BT474         3    L      1   1        no 3.16737  YES  NO
BT483        BT483         4    L      1   1        no 2.97456   NO  NO
CAMA1        CAMA1         5    L      1   1        no 3.16856 <NA>  NO
DU4475      DU4475         6    L      1   1        no 3.06074 <NA>  NO
        known.p53 BRCA order adherence per3
X600MPE        NO <NA>     1      <NA>    9
AU565         YES <NA>     2      <NA>    4
BT474          NO loss     3      <NA>   NA
BT483          NO   WT     4      <NA>   NA
CAMA1          NO   WT     5      <NA>   NA
DU4475         NO   WT     6      <NA>   NA
```

- Los experimentos los obtienen investigadores y hay que incluir esta información en los datos.

```
datosexperimento = new('MIAME',name='Neve et alt.',lab='Varios',
                       contact ='rmneve@lbl.gov',
                       title = 'A collection of breast cancer cell lines
                       for the study of
functionally distinct cancer subtypes',
abstract = 'An example ExpressionSet',
url = '10.1016/j.ccr.2006.10.008',
other = list(notes = 'Creado a partir  de ficheros de texto'))
```

El formato MIAME (Minimum information about a microarray experiment) puede consultarse en http://fged.org/projects/miame/

Y montamos las piezas

```
jcliment  = new("ExpressionSet",exprs=exprs0,phenoData = datosfenotipo,
  experimentData = datosexperimento,annotation = "hgu95av2")
```

Guardamos el `ExpressionSet`.

```
save(jcliment,file="jcliment.rda")
```

ArrayExpress

Otra vez Neve (2006)

- Hemos visto cómo construir un `ExpressionSet` cuando tenemos los datos de expresión, los fenotípicos y los de anotación.
- Los autores han subido toda la información del experimento a https://www.ebi.ac.uk/arrayexpress/.

```
library(ArrayExpress)
rawset = ArrayExpress("E-TABM-157")
```

Nos devuelve un objeto `affy::AffyBatch`.

GSE21779 (de dos formas)

gse21779: Gene Expression Omnibus (GEO)

- http://www.ncbi.nlm.nih.gov/geo/
- Bajamos los datos GSE21779.
- Los ficheros CEL los tenemos en http://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS4128
- Podemos usar **GEOquery**

```
library(GEOquery)
```

- Sabiendo el identificador podemos bajar los datos.

```
gcel = getGEOSuppFiles("GSE21779")
```

```
setwd("GSE21779/")
system("tar xvf GSE21779_RAW.tar")
library(affy)
gse21779raw = ReadAffy()
setwd("../")
system("rm -fr GSE21779")
gse21779rma = affy::rma(gse21779raw)
pData(gse21779rma)
gse = getGEO("GSE21779")
class(gse)
length(gse)
class(gse[[1]])
pData(gse[[1]])
rownames(pData(gse21779rma)) =
    sapply(rownames(pData(gse21779rma)),function(x)  unlist(strsplit(x,split=".CEL."))[1])
all(rownames(pData(gse21779rma)) == rownames(pData(gse[[1]])))
pData(gse21779rma) = pData(gse[[1]])
gse21779 = gse21779rma
```

geod21779: ArrayExpress

```
library(ArrayExpress)
geod21779raw = ArrayExpress("E-GEOD-21779")
geod21779 = rma(geod21779raw)
```

Correspondencias múltiples

El problema

- Necesitamos conocer la correspondencia entre sondas y genes.

- La opción más simple y primera a probar es utilizar un paquete de anotación.
- https://www.bioconductor.org/packages/release/data/annotation/

- En Aula Virtual tenemos los datos.

```
load("gse21779raw.rda")
```

- Normalizamos los datos.

```
library(Biobase)
dim(gse21779raw)
gse21779rma = affy::rma(gse21779raw)
save(gse21779rma,file=paste0(dirTamiData,"gse21779rma.rda"))
```

- ¿Cuántas sondas tenemos después de la normalización?

```
class(gse21779rma)
```

```
[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"
```

```
nrow(gse21779rma)
```

```
Features
   54675
```

```
ncol(gse21779rma)
```

```
Samples
     18
```

```
dim(gse21779rma)
```

```
Features  Samples
   54675       18
```

```
annotation(gse21779rma)
```

```
[1] "hgu133plus2"
```

- Establecemos las correspondencias con `Entrez` con `AnnotationDbi::select()`.

```
pacman::p_load(hgu133plus2.db)
a = AnnotationDbi::select(hgu133plus2.db,
                          keys=featureNames(gse21779rma),
                          columns=c("ENTREZID"),
                          keytype="PROBEID")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
dim(a)
```

```
[1] 57151     2
```

- Vemos que tenemos más filas en `a` que en `gse21779`.

```
dim(a)
```

```
[1] 57151     2
```

- Tenemos las correspondencias en forma de `data.frame`.

```
class(a)
```

```
[1] "data.frame"
```

- ¿Cuántas y qué sondas tienen correspondencia con más de un gen?

```
b = which(table(a[,1]) > 1)
length(b)
```

```
[1] 1528
```

```
sel = is.element(a[,1],names(b))
table(table(a[sel,1]))
```

```
   2    3    4    5    6    7    8    9   10   11   12   13   14   15   17   21
1199  152   73   30   16   14   11   11    3    4    2    1    2    3    2    1
  22
   4
```

- Una posibilidad puede ser quedarnos con la primera aparición.

```
c1 = match(unique(a[,1]),a[,1])
a1 = a[c1,]
fData(gse21779rma) = a1
```

- Vamos a ver si cada identificador ENTREZID tiene una sola correspondencia PROBEID.

```
table(table(a1[,2]) >1)
```

```
FALSE  TRUE
 9907 11451
```

- Por ello hemos de quedarnos con una sola sonda por identificador ENTREZID.

- Podemos repetir el código anterior para estos identificadores.

```
c2 = match(unique(a1[,2]),a1[,2])
a2 = a1[c2,]
dim(a2)
```

```
[1] 21359     2
```

```
gse21779 = gse21779rma[c2,]
```

- Podemos comprobar que ahora sí hay una correspondencia 1-1.

```
length(unique(a2[,1])) == length(unique(a2[,2]))
```

```
[1] TRUE
```

- Llevamos de nuestro atrevimiento queremos establecer la correspondencia de los identificadores PROBEID con ENTREZID y con ENSEMBL.

```
a = AnnotationDbi::select(hgu133plus2.db,
                          keys=featureNames(gse21779rma),
                          columns=c("ENTREZID","ENSEMBL"),
                          keytype="PROBEID")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
head(a)
```

```
    PROBEID ENTREZID         ENSEMBL
1 1007_s_at      780 ENSG00000204580
2 1007_s_at      780 ENSG00000229767
3 1007_s_at      780 ENSG00000234078
4 1007_s_at      780 ENSG00000137332
5 1007_s_at      780 ENSG00000223680
6 1007_s_at      780 ENSG00000215522
```

```
c1 = match(unique(a[,1]),a[,1])
a1 = a[c1,]
c2 = match(unique(a1[,2]),a1[,2])
a2 = a1[c2,]
```

- Sin embargo, hemos de tener en cuenta que no tenemos identificadores únicos `ENSEMBL`.

```
table(table(a1[,3]) >1)
```

```
FALSE  TRUE
 9013 11187
```

- Si queremos una correspondencia 1-1 hemos de decidir entre qué par de identificadores lo hacemos.

Un ejemplo Agilent

GSE104645

```
wd = getwd()
setwd(dirTamiData)
GEOquery::getGEOSuppFiles("GSE104645")
setwd("GSE104645")
system("tar xvf GSE104645_RAW.tar")
system("gzip -d *.gz")
GSE104645raw = limma::read.maimages(dir(".", "txt"), "agilent",
                                    green.only = TRUE,
                                    other.columns="gIsWellAboveBG")
```

```
save(GSE104645raw,file=paste0(dirTamiData,"GSE104645raw.rda"))
setwd(wd)
```

Eliminamos correspondencias múltiples.

```
pacman::p_load(hgug4112a.db)
a = AnnotationDbi::select(hgug4112a.db,
                  keys=GSE104645raw$genes[,"ProbeName"],
                  column=c("ENTREZID","ENSEMBL"),
                  keytype="PROBEID")
## Eliminamos sondas sin ENTREZID
a = a[!is.na(a[,"ENTREZID"]),]
c1 = match(unique(a[,1]),a[,1])
a1 = a[c1,]
c2 = match(unique(a1[,2]),a1[,2])
a2 = a1[c2,]
a2 = na.omit(a2)
GSE104645raw2 =
    GSE104645raw[match(a2[,1],
                       GSE104645raw$genes[,"ProbeName"]),]
dim(GSE104645raw2)
rownames(GSE104645raw2) = a2[,1]
save(GSE104645raw2,file=paste0(dirTamiData,
                               "GSE104645raw2.rda"))
```

Hacemos la corrección de fondo.

```
GSE104645el = limma::backgroundCorrect(GSE104645raw2,
                                       method="normexp")
```

Aplicamos una normalización de cuantiles.

```
GSE104645el = limma::normalizeBetweenArrays(GSE104645el,
                                            method="quantile")
```

Guardamos los datos.

```
save(GSE104645el,file=paste0(dirTamiData,"GSE104645el.rda"))
```

Contruimos el `Biobase::ExpressionSet`.

```
pd0 = data.frame(case = 1:ncol(GSE104645el))
rownames(pd0) = colnames(GSE104645el$E)
metadata = data.frame(labelDescription = c("case"),
                      row.names=colnames(pd0))
phenotypedata = new("AnnotatedDataFrame", data = pd0,
                    varMetadata = metadata)
experimentdata = new('MIAME',name="Okita A, Takahashi S,
 Ouchi K, Inoue M et al.",
 lab="Tohoku University Hospital",
 contact ="akira.okita.d8@tohoku.ac.jp",
 title = "Consensus molecular subtypes classification
 of colorectal cancer as a predictive factor for
chemotherapeutic efficacy against metastatic colorectal
cancer",
abstract = "Gene expression profile from 193
 formalin-fixed and paraffin embedded primary tumor
samples.",
url = "https://www.ncbi.nlm.nih.gov/pubmed/29721154",
other = list(notes = "An example of Agilent microarray"))

GSE104645 = new("ExpressionSet",exprs=GSE104645el$E,
                phenoData = phenotypedata,
                experimentData = experimentdata,
                annotation = "hgug4112a.db")
```

Añadimos el `ExpressionSet::fData` con los identificadores.

```
fData(GSE104645) = a2[,1:2]
```