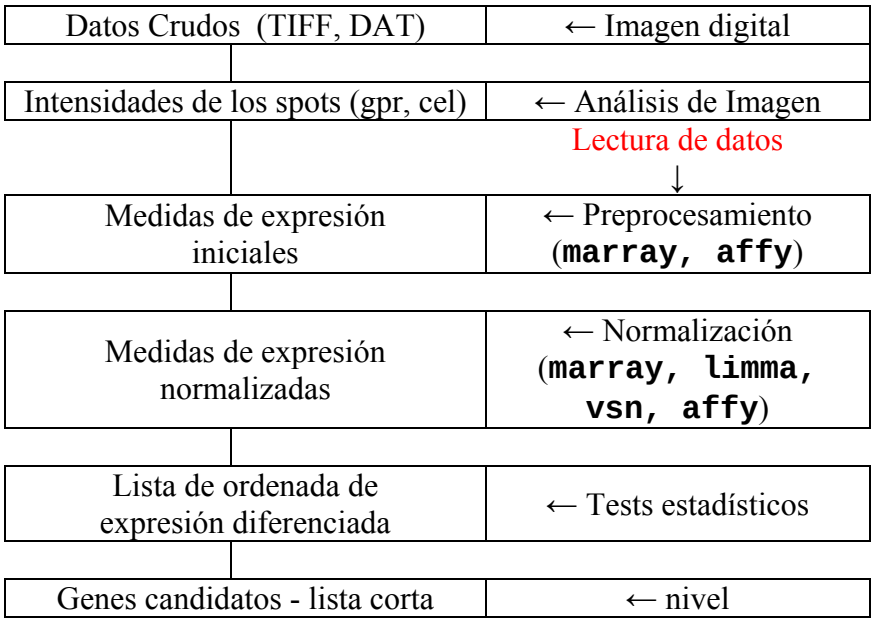
4. Recordemos algunos aspectos del análisis de datos de microarreglos

4.1 Esquema general para hallar genes diferencialmente expresados



Bioconductor tiene unos 180 paquetes para procesamiento de datos de microarreglos

4.2 Preguntas que interesa responder en genómica funcional

¿Qué secuencias genómicas están expresadas diferencialmente en cada tejido?

¿ Cómo es la expresión de un gen afectado por influencias extracelulares?

¿ Qué genes se expresan durante el desarrollo de un organismo?

¿ Cómo cambia el nivel de expresión de un gen durante el desarrollo y la diferenciación?

¿ Cuál es el efecto de una mala regulación en la expresión de un gen?

¿ Qué patrones de la expresión del gen causan una enfermedad o conducen a la progresión de la enfermedad?

¿ Qué patrones de expresión de influencian la respuesta a un tratamiento?

Para responder cada una de las preguntas anteriores debemos detectar grupos de genes diferencialmente expresados entre dos o más grupos.

5. Lectura de datos de un experimento de microarreglos

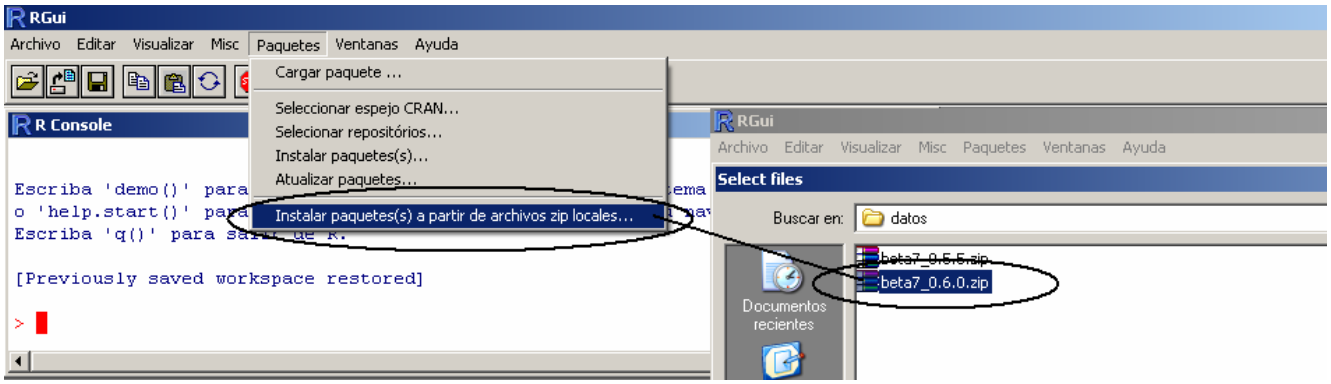
Utilizaremos como ejemplo los datos que bajamos del paquete beta7.

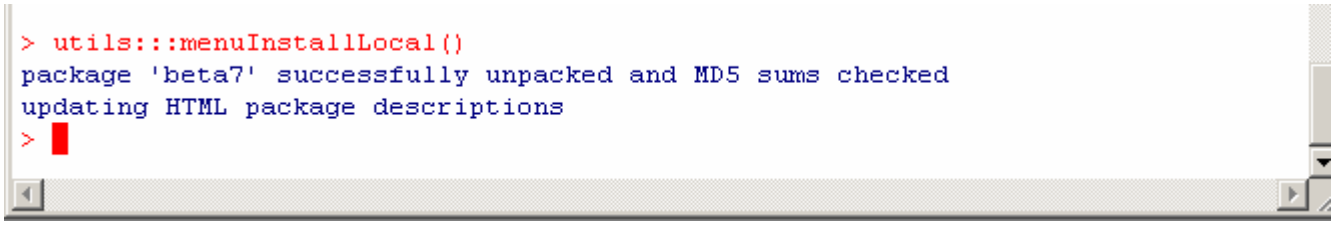
en 2006
http://www.bioconductor.org/packages/monograph/1.2/src/contrib/html/beta7.html
-> Win32 package download -> beta7_05.05.zip

en 2007
http://bioconductor.org/packages/2.0/data/experiment/html/beta7.html
->Windows binary beta7_0.6.0.zip

y los guardamos en una carpeta, por ejemplo la carpeta
"E:\diana\cursos\Microarrays 2007\ datos"

Desde R: **`Paquetes -> Instalar paquetes(s) a partir de archivos zip locales...`**





Datos de Rodriguez et al. (2004) Differential Gene Expression by Memory/Effector T Helper Cells Bearing the Gut-Homing Receptor Integrin alpha4 beta7.

Microarreglos: de oligonucleótidos de 70 bases con 23 184 sondas que incluyen 1760 spots de control (negativos positivos y de normalización)

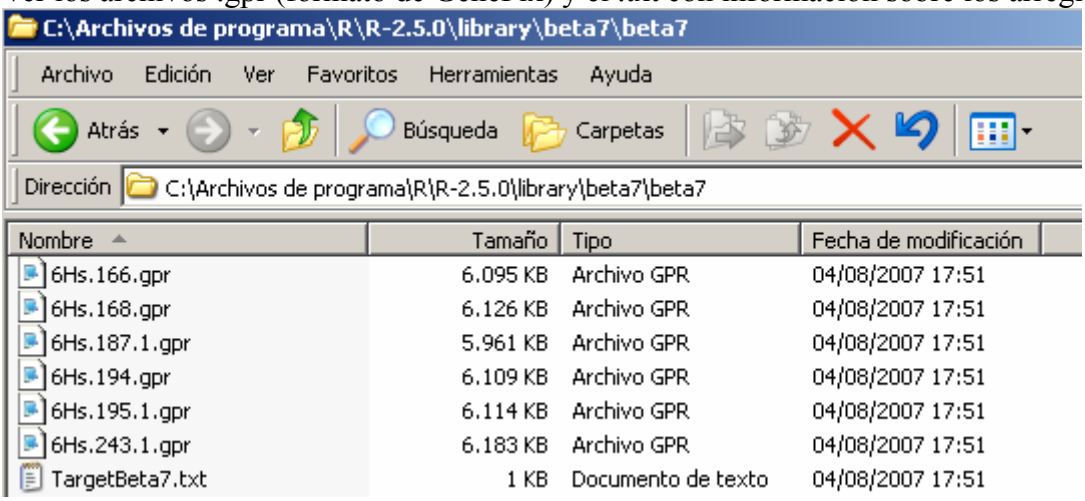
Muestras (target): Cada microarreglo fue hibridizado con RNA de dos tipos de células humanas sanguíneas $CD4^+T$: $\beta7^-$ y $\beta7^+$ (las células se distinguen por la expresión superficial del "integrin β7") provenientes de un mismo sujeto sano. Diferentes tintes se utilizaron para etiquetar el RNA de cada tipo de célula (dos canales)

Cuantificación: Todos los arreglos fueron cuantificados usando el software GenePix de Axon, así que tenemos archivos de la cuantificación gpr. Los archivos del tiff (Tagged Image File Format) están también disponibles para bajar.

Descripción: Los datos contienen sólo un subconjunto de 6 archivos gpr del experimento que apunta a identificar genes expresados diferencialmente entre las células 7+ beta y las 7- beta. Los 6 correspondían a un solo tipo de la plataforma, y tenían un formato común de en la disposición de las sondas (array layout).

¿Dónde están los datos?
En la subcarpeta library dentro de las carpetas de instalación del R:
"C:\Archivos de programa\R\R-2.5.0\library\beta7\beta7"

Podemos ver los archivos .gpr (formato de GenePix) y el .txt con información sobre los arreglos:



Nombre	Tamaño	Tipo	Fecha de modificación
6Hs.166.gpr	6.095 KB	Archivo GPR	04/08/2007 17:51
6Hs.168.gpr	6.126 KB	Archivo GPR	04/08/2007 17:51
6Hs.187.1.gpr	5.961 KB	Archivo GPR	04/08/2007 17:51
6Hs.194.gpr	6.109 KB	Archivo GPR	04/08/2007 17:51
6Hs.195.1.gpr	6.114 KB	Archivo GPR	04/08/2007 17:51
6Hs.243.1.gpr	6.183 KB	Archivo GPR	04/08/2007 17:51
TargetBeta7.txt	1 KB	Documento de texto	04/08/2007 17:51

Veamos que aspecto tienen los archivos anteriores con la información del procesamiento de las imágenes. Son archivos texto que abrimos con el notepad

.....

5.1 Datos en R

Un Objeto en R es una colección de variables individuales variables y/u otros objetos que están unidos

Un objeto con datos de un experimento de microarreglos contendrá

- intensidades de los spots, u otros estadísticos
- datos de las muestras (paciente, tipo de tejido, localización, diagnostico, seguimiento, tipo de línea celular, etc.)
- datos sobre los genes (secuencia, identificación, anotaciones)

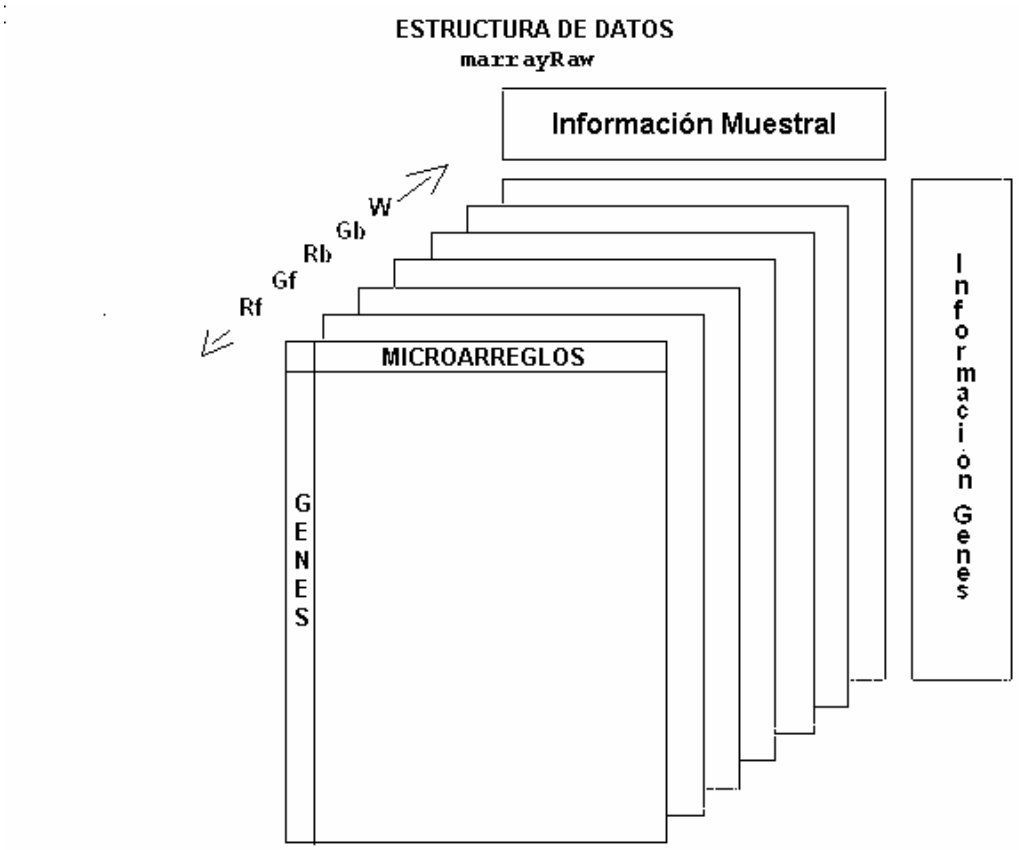
Veamos algunos términos que utilizaremos en R en el contexto de los paquetes de manejo de datos de microarreglos:

- **class - clase**: la definición "abstracta" del objeto
- **object - objeto**: una instancia concreta
- **method - método**: otra palabra para 'function'

- **slot - lugar** : una componente de un objeto

```
> class(beta7)
[1] "marrayRaw"
attr(,"package")
[1] "marray"
```

5.2 Estructura de datos en el paquete `marray` para el preprocesamiento de datos de microarreglos de 2 canales



Estructuras básicas

`marrayInfo` : contiene la información sobre la muestra o sobre los genes
`marrayLayout` : describe la geometría del arreglo
`marrayRaw` : contiene los datos iniciales y las dos estructuras anteriores
`marrayNorm` : contiene los datos después de la normalización y las dos estructras anteriores

```
> slotNames(beta7)
[1] "maRf"      "maGf"      "maRb"      "maGb"      "maW"
[6] "maLayout"  "maGnames"  "maTargets" "maNotes"
```

Estructura del objeto **`marrayInfo`**

El mismo tipo de objeto (**`marrayInfo`**)es utilizado para guardar información **sobre la muestra** (**`maTargets`**) o **sobre los genes (`maGnames`)**. Este objeto es un *data frame* con información adicional. La información adicional incluye etiquetas y una cadena de caracteres con las notas que se quiera agregar al objeto.

Cuando se utiliza para describir genes, las filas corresponden a los spots en el arreglo y las columnas a las anotaciones respecto a los genes.

```
> slot(beta7,"maGnames")
An object of class "marrayInfo"
@maLabels
[1] "H200000297" "H200000303" "H200000321" "H200000327"
[5] "H200000345"
23179 more elements ...

@maInfo
                  ID
H200000297 H200000297
H200000303 H200000303
H200000321 H200000321
H200000327 H200000327
H200000345 H200000345
                                                                                Name
H200000297                              OVGP1 - Oviductal glycoprotein 1, 120kD (mucin 9, oviductin)
H200000303 TAF1 - TAF1 RNA polymerase II, TATA box binding protein (TBP)-associated factor, 250 kD
H200000321                                                                 APOA4 - Apolipoprotein A-IV
H200000327                                                       BMP1 - Bone morphogenetic protein 1
H200000345                                            F12 - Coagulation factor XII (Hageman factor)
23179 more rows ...

@maNotes
[1] ""
```

Cuando se utilizan para describir muestras, las filas corresponden a los microarreglos y las columnas contienen la información respecto a las muestras. En particular las columnas deberían identificar las muestras utilizadas en los canales Cy3 y Cy5

```
> slot(beta7,"maTargets")
An object of class "marrayInfo"
@maLabels
[1] "6Hs.195.1.gpr" "6Hs.168.gpr"   "6Hs.166.gpr"   "6Hs.187.1.gpr"
[5] "6Hs.194.gpr"   "6Hs.243.1.gpr"

@maInfo
      FileNames SubjectID  Cy3  Cy5 Date of Blood Draw Date of Scan
1 6Hs.195.1.gpr         1 b7 - b7 +         2002.10.11   2003.07.25
2   6Hs.168.gpr         3 b7 + b7 -         2003.01.16   2003.08.07
3   6Hs.166.gpr         4 b7 + b7 -         2003.01.16   2003.08.07
4 6Hs.187.1.gpr         6 b7 - b7 +         2002.09.16   2003.07.18
5   6Hs.194.gpr         8 b7 - b7 +         2002.09.18   2003.07.25
6 6Hs.243.1.gpr        11 b7 + b7 -         2003.01.13   2003.08.06

@maNotes
[1] "TargetBeta7.txt"
```

Estructura del objeto **`marrayLayout - lugares - slots`**

El objeto **marrayLayout** (que también está contenido en **marrayRaw**) utiliza cinco números (**slots**) para describir la geometría del vidrio :

maNgr : cantidad filas de sectores de la grilla
maNgc : cantidad columnas de sectores de la grilla
maNsr : cantidad filas de spots dentro de cada sector
maNsc : cantidad columnas de spots dentro de cada sector
maNspots : cantidad total de spots (aunque ésta puede deducirse de las anteriores)

Puede tener tres vectores adicionales, de longitud igual a la cantidad de spots del microarreglo.

maSub : un vector lógico (TRUE, FALSE) indicando si se tiene interés en ese spot.
maPlate : un vector numérico, de qué plato tomó el robot este spot?
maControls : que tipo de material se fijó en ese spot (sonda , buffer, vacío, etc.)?

Recordemos que los microarreglos contienen una grilla rectangular compuesta por sectores rectangulares. A su vez los spots dentro de cada sector han sido generados por la misma aguja y entre sectores, en general, por agujas distintas.

```
> slot(beta7,"maLayout")
An object of class "marrayLayout"
@maNgr
[1] 12

@maNgc
[1] 4

@maNsr
[1] 23

@maNsc
[1] 21

@maNspots
[1] 23184

@maSub
[1] TRUE TRUE TRUE TRUE TRUE
23179 more elements ...

@maPlate
[1] 1 1 1 1 1
61 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 ... 61
23179 more elements ...

@maControls
[1] probes probes probes probes probes
Levels: Buffer Empty Negative Positive probes
23179 more elements ...

@maNotes
character(0)
```

Métodos para los **marrayLayout**

Los siguientes métodos pueden aplicarse a un objeto de clase **marrayLayout**

maPrintTip : vector indicando la aguja para cada spot
maGridCol : vector indicando la columna del sector de la grilla
maGridRow : vector indicando la columna del sector de la grilla
maSpotCol : vector indicando la columna del spot dentro del sector
maSpotRow : vector indicando la fila del spot dentro del sector

***Estructura datos del* `marrayRaw` - lugares- slots**

A los datos iniciales, en este contexto se los llama *crudos* **`Raw`**, pero sabemos que ya han sido bastante *cocinados* durante el paso de procesamiento de la imagen.

Recordemos

```
> slotNames(beta7)
[1] "maRf"      "maGf"      "maRb"      "maGb"      "maW"
[6] "maLayout"  "maGnames"  "maTargets" "maNotes"
```

Los datos crudos de los niveles de expresión de un experimento de microarreglo de vidrio son guardados en un objeto **`marrayRaw`**. Actualmente este objeto contiene, además de los datos crudos, la información sobre el microarreglo, la muestra y los genes. Su estructura está dada por los siguientes slots:

- Cuatro matrices de datos crudos (**`maRf, maGf, maRb, maRb`**) con las estimaciones del foreground (f) y background (b) para los canales rojo (R) y verde (G, green) de un lote de microarreglos
- Una matriz opcional **`(maW`**) con pesos de calidad del spot
- **`maLayout`**, contiene la geometría del arreglo (ya vimos su estructura)
- **`maGnames`**, es un objeto de la clase **`marrayInfo`** conteniendo la información de los genes
- **`maTargets`**, es un objeto de la clase **`marrayInfo`** conteniendo la información de la muestra

Los objetos de clase **`"matrix"`** (**`maRf, maGf, maRb, maRb`**) contienen los estadísticos para cada spot, son matrices en las cuales las *filas* corresponden a los diferentes probes - secuencias o genes spoteados - y las *columnas* a cada uno de los arreglos del lote.

```
> slot(beta7,"maRf")[1:5,]# primeras 5 filas
     6Hs.195.1.gpr 6Hs.168.gpr 6Hs.166.gpr 6Hs.187.1.gpr 6Hs.194.gpr 6Hs.243.1.gpr
[1,]           227         370         139            69         187           241
[2,]           208         273         181            80         213           241
[3,]           239         288         238            77         377           307
[4,]           247         301         228            91         329           280
[5,]           221         290         218            91         351           236
```

Es decir que cada columna representa un arreglo. Se supone que los spots en las columnas están ordenados por bloque (de izquierda a derecha y de arriba hacia abajo) y dentro de cada bloque (también de izquierda a derecha y de arriba hacia abajo). Veremos esto con más detalle en el laboratorio.

Métodos de la clase **`marrayRaw`**

Los métodos que hemos visto pueden aplicarse a un objeto de clase **`marrayLayout`**, también pueden aplicarse a un objeto de clase **`marrayRaw`**

Además existen métodos específicos para la clase **`marrayRaw`**

maA : matriz de logaritmos en base 2 de intensidades (con corrección por background)
maM : matriz de logaritmos en base 2 de cocientes (con corrección por background)
maLR : matriz de logaritmos de (intensidades - background) para el canal rojo
maLG: matriz de logaritmos de (intensidades - background) para el canal verde

Utilizamos la siguiente notación:

intensidad = $((Rf-Rb)*(Gf-Gb))^{1/2}$
cociente = (Rf-Rb)/ (Gf-Gb)
A = log2(intensidad)
M= log2(cociente)

Pueden calcularse una intensidad y un cociente (razón, ratio) para cada gen de cada microarreglo

Observación

Los objetos **`marrayRaw`** y **`marrayNorm`** contienen su propia copia del objeto **`marrayInfo`** con la información de los genes. Esta estructura ha sido criticada porque desperdicia espacio en disco y es más difícil de mantener la información (Kevin R. Coombes and Keith A. Baggerly 2005 http://bioinformatics.mdanderson.org/MicroarrayCourse). Para actualizar las anotaciones, deben realizarse en cada una de las copias.

5.3 Lectura de datos en el `marray`. Armando un objeto de clase marrayRaw

Ejemplificaremos la construcción de un objeto de clase marrayRaw a partir de los archivos ASCII que contienen la información del experimento beta7

5.3.1 Cuantificación de datos microarreglos

La mayoría de los escaners de microarreglos incluyen un software para su cuantificación. Cada uno de ellos utiliza su propio

- método de direccionamiento, segmentación y cuantificación de los spots.
- esquema para el etiquetado de los spots
- orden para registrar los spots
- nombres para las medidas que reportan.

Todos pueden exportar los datos en archivos ASCII delimitado por tabulaciones (tab-separated-values format) con filas representando los spots y columnas representando las mediciones (tales como posición, intensidad de la señal, intensidad del fondo, etc.)

Funciones de lectura

Para resolver el problema de esta variabilidad de formatos el paquete **marray** diferentes funciones de lectura de datos:

- **read.GenePix**
- **read.Spot**
- **read.SMD**
- **read.marrayRaw**

5.3.2 Lectura de la información sobre las muestras

En R cargamos el paquete "marray"

```
> library("marray")
```

Primero leemos el archivo: **TargetBeta7.txt** para obtener información sobre las muestras.

Generamos un objeto que llamaremos "**TargetInfo**" de clase **"marrayInfo"** con la información sobre las muestras

```
> Infomuestras <- read.marrayInfo("C://Archivos de programa//R//R-
2.5.0//library//beta7//beta7//TargetBeta7.txt")

> Infomuestras
An object of class "marrayInfo"
@maLabels
[1] "6Hs.195.1.gpr" "6Hs.168.gpr" "6Hs.166.gpr" "6Hs.187.1.gpr" "6Hs.194.gpr"
[6] "6Hs.243.1.gpr"

@maInfo
      FileNames SubjectID  Cy3  Cy5 Date of Blood Draw Date of Scan
1 6Hs.195.1.gpr         1 b7 - b7 +         2002.10.11   2003.07.25
2   6Hs.168.gpr         3 b7 + b7 -         2003.01.16   2003.08.07
3   6Hs.166.gpr         4 b7 + b7 -         2003.01.16   2003.08.07
4 6Hs.187.1.gpr         6 b7 - b7 +         2002.09.16   2003.07.18
5   6Hs.194.gpr         8 b7 - b7 +         2002.09.18   2003.07.25
6 6Hs.243.1.gpr        11 b7 + b7 -         2003.01.13   2003.08.06

@maNotes
[1] "C://Archivos de programa//R//R-
2.5.0//library//beta7//beta7//TargetBeta7.txt"
```

Cada archivo **.gpr** contiene la información de un microarreglo. El conjunto de los 6 microarreglos corresponden a un lote (**batch**) de microarreglos.

Otra forma de leer el archivo **TargetBeta7.txt** es:

```
> read.marrayInfo(file.path(dirección.beta7,"TargetBeta7.txt"))
```

5.3.3 Lectura de la información numérica sobre las intensidades. Generación de datos de la clase marrayRaw

Ahora obtenemos información numérica de los archivos **.gpr** mediante la función **read.GenePix**() que lee todos los archivos **.gpr** del directorio en que se está parado y genera un objeto de la clase **marrayRaw** que contendrá la información de todos los microarreglos del lote más la información de las muestras en **TargetInfo**.

```
> crudos <- read.GenePix(targets = Infomuestras, path =  "C://Archivos de programa//R//R-
2.5.0//library//beta7//beta7")
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.195.1.gpr
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.168.gpr
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.166.gpr
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.187.1.gpr
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.194.gpr
Reading ...  C://Archivos de programa//R//R-2.5.0//library//beta7//beta7/6Hs.243.1.gpr
```

otra forma

```
>  dirección.beta7 <-  "C://Archivos de programa//R//R-
2.5.0//library//beta7//beta7"
>   setwd(dirección.beta7) # para leer desde esa dirección
>   crudos <- read.GenePix(targets = Infomuestras)
Reading ...  6Hs.195.1.gpr
Reading ...  6Hs.168.gpr
Reading ...  6Hs.166.gpr
Reading ...  6Hs.187.1.gpr
Reading ...  6Hs.194.gpr
Reading ...  6Hs.243.1.gpr
```

Con la opción **targets = Infomuestras** estamos incluyendo en el objeto **crudos** la información contenida en **TargetInfo** el objeto de clase "'marrayInfo" que generamos previamente.

También podemos generar un objeto **marrayRaw** seleccionando un subconjunto de los arreglos, en este caso elegimos dos **"6Hs.166.gpr"** y **"6Hs.187.1.gpr"**.

```
> archivos <- c ( "6Hs.166.gpr" ,  "6Hs.187.1.gpr")
> crudos2 <- read.GenePix(archivos, name.Gb= NULL, name.Rb= NULL)
Reading ...  6Hs.166.gpr
Reading ...  6Hs.187.1.gpr
```

Ignoramos las estimaciones de las intensidades del background para los canales verde (**G**) y rojo (**R**) fijando el valor **NULL** para los argumentos **name.Gb** y **name.Rb**.

```
>  slotNames(crudos)
[1] "maRf"      "maGf"      "maRb"      "maGb"      "maW"
[6] "maLayout"  "maGnames"  "maTargets" "maNotes"

>  slotNames(crudos2)
[1] "maRf"      "maGf"      "maRb"      "maGb"      "maW"
```

```
[6] "maLayout"  "maGnames"  "maTargets" "maNotes"
```

Ambos objetos (**crudos, crudos2**) tienen la misma estructura de slots. Los primeros 5
"maRf" "maGf" "maRb" "maGb" "maW"
son las *matrices* que contienen la información cuantitativa básica extraida de los archivos**.gpr**. Los demás están asociados con los archivos de la estructura de los arreglos (*layout*) y las anotaciones.

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

```
> crudos2@maGf[1:5,] # seleccionamos las primeras 5 filas
     6Hs.166.gpr 6Hs.187.1.gpr
[1,]         189           106
[2,]         255           133
[3,]         256           123
[4,]         302           141
[5,]         328           156

> crudos@maGf[1:5,] # seleccionamos las primeras 5 filas
     6Hs.195.1.gpr 6Hs.168.gpr 6Hs.166.gpr 6Hs.187.1.gpr
[1,]           357         304         189           106
[2,]           292         235         255           133
[3,]           284         223         256           123
[4,]           320         254         302           141
[5,]           361         290         328           156
     6Hs.194.gpr 6Hs.243.1.gpr
[1,]         218           165
[2,]         216           183
[3,]         290           194
[4,]         301           252
[5,]         327           247
```

Ya hemos visto que la función **slot** también nos permite acceder a los slots.

```
> slot(crudos2 ,"maGf")[1:5,]
     6Hs.166.gpr 6Hs.187.1.gpr
[1,]         189           106
[2,]         255           133
[3,]         256           123
[4,]         302           141
[5,]         328           156
```

La matriz **crudos2@maGf** tiene *2 columnas* mientras que la **crudos@maGf** tiene 6 como era de esperar.

5.3.4 Verificación de algunos métodos de la clase `marrayRaw`

En la sección 5.2 hemos mencionado los siguientes métodos específicos para la clase **marrayRaw**

maA : matriz de logaritmos en base 2 de intensidades (con corrección por background)
maM : matriz de logaritmos en base 2 de cocientes (con corrección por background)
maLR : matriz de logaritmos de (intensidades - background) para el canal rojo
maLG : matriz de logaritmos de (intensidades - background) para el canal verde

Vamos a realizar cálculos "a mano" y compararlos con los métodos anteriores. Utilizaremos el objeto de clase **`marrayRaw`** beta7 de la librería beta7 .

```
> library(beta7)
```

Recordemos la estructura de los arreglos

```
> summary(beta7@maLayout)
Array layout:    Object of class marrayLayout.

Total number of spots:                  23184
Dimensions of grid matrix:              12 rows by 4 cols
Dimensions of spot matrices:            23 rows by 21 cols

Currently working with a subset of 23184spots.

Control spots:
There are   5 types of controls :

  Buffer    Empty Negative Positive   probes
       3     1328      225      204    21424


Notes on layout:
```

Los microarreglos del lote contenido en beta7 tienen una grilla 12 filas y 4 columnas. Esto es 12 x 4 = 48 grupos de agujas (pin-tip groups), cada uno de los cuales ha sido impreso con la misma aguja. A su vez cada aguja imprimió sobre una matriz con 23 filas y 21 columnas. Hay 5 tipos de spots (**`Buffer,Empty, Negative,Positive, probes`**).

La figura 33 muestra, a modo de ejemplo, un microarreglo de 4 x 4 grupos de agujas construido a partir de un plato con 96 cavidades.

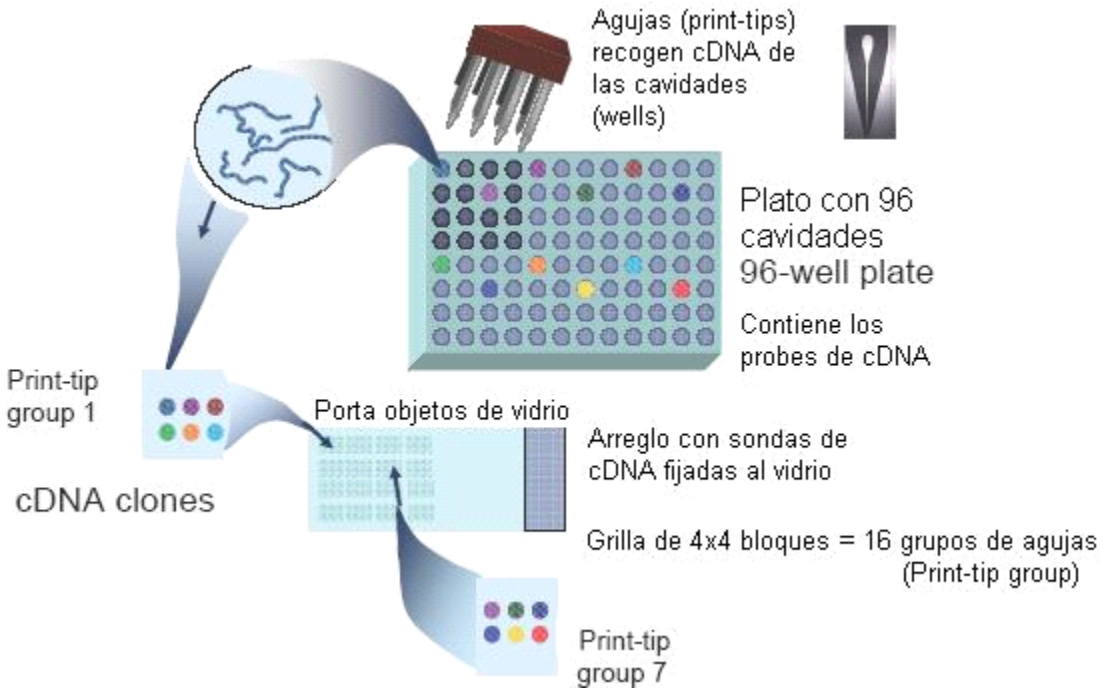


Figura 33: Descripción de los grupos de spots que tiene cada aguja. El gráfico en su contexto original se encuentra en **www.stat.berkeley.edu/~sandrine/Docs/Talks/MBI04/mbi.html**

Obtenemos información sobre las muestras

```
> summary(beta7@maTargets)
Object of class marrayInfo.

       maLabels     FileNames SubjectID  Cy3  Cy5 Date of Blood Draw
1 6Hs.195.1.gpr 6Hs.195.1.gpr         1 b7 - b7 +         2002.10.11
2   6Hs.168.gpr   6Hs.168.gpr         3 b7 + b7 -         2003.01.16
3   6Hs.166.gpr   6Hs.166.gpr         4 b7 + b7 -         2003.01.16
4 6Hs.187.1.gpr 6Hs.187.1.gpr         6 b7 - b7 +         2002.09.16
5   6Hs.194.gpr   6Hs.194.gpr         8 b7 - b7 +         2002.09.18
6 6Hs.243.1.gpr 6Hs.243.1.gpr        11 b7 + b7 -         2003.01.13

 Date of Scan
1   2003.07.25
2   2003.08.07
3   2003.08.07
4   2003.07.18
5   2003.07.25
6   2003.08.06



Number of labels:  6
Dimensions of maInfo matrix:  6  rows by  6  columns

Notes:
 TargetBeta7.txt

> summary(beta7)
```

```
Pre-normalization intensity data:        Object of class marrayRaw.

Number of arrays:       6 arrays.

A) Layout of spots on the array:

B) Samples hybridized to the array:

C) Summary statistics for log-ratio distribution:

                Min. 1st Qu. Median  Mean 3rd Qu. Max. NA's
6Hs.195.1.gpr -6.13   -1.00  -0.52 -0.50   -0.08 5.95 3415
6Hs.168.gpr   -7.08   -0.80  -0.21 -0.23    0.34 5.19 2839
6Hs.166.gpr   -7.07   -1.25  -0.64 -0.62   -0.02 6.15 3440
6Hs.187.1.gpr -9.81   -0.92  -0.60 -0.55   -0.25 5.00 2942
6Hs.194.gpr   -5.93    0.00   0.44  0.53    0.90 7.74 6090
6Hs.243.1.gpr -6.38   -1.13  -0.69 -0.64   -0.21 7.05 2227
```

Log ratios , cuál es el cociente que realiza por defecto, sin o con corrección por background? Cy3/Cy5? Cy5/Cy3? El último de acuerdo con la documentación, pero más adelante tendremos que verificarlo.

```
D) Notes on intensity data:
GenePix Data
```

Retomemos los métodos de la clase **marrayRaw**: **maA, maM, maLR, maLG** y apliquemos estos métodos a **beta7**

```
> intensid <-maA(beta7)
> cocientes <-maM(beta7)
> difLR <-maLR(beta7)
> difLG<- maLG (beta7)
```

Vemos las intensidades de las dos primeras filas

```
> instensid[1:2,]
     6Hs.195.1.gpr 6Hs.168.gpr 6Hs.166.gpr 6Hs.187.1.gpr 6Hs.194.gpr
[1,]      6.895683    7.298095          NA            NA          NA
[2,]      5.965369    6.322153     5.60807      3.953445          NA
     6Hs.243.1.gpr
[1,]      3.336213
[2,]      5.213656
>
```

¿Porqué da **NA** en algunas intensidades? Resta el background. Veamos qué ocurre con la primera fila.

El canal rojo no presenta inconvenientes, todos los valores del background son menores que los del foreground:

```
>   slot(beta7,"maRf")[1, ] #primera fila de la matriz maRf - foreground
6Hs.195.1.gpr   6Hs.168.gpr   6Hs.166.gpr 6Hs.187.1.gpr   6Hs.194.gpr
          227           370           139            69           187
```

```
6Hs.243.1.gpr
          241


>    slot(beta7,"maRb")[1, ] #primera fila de la matriz maRb - background
6Hs.195.1.gpr   6Hs.168.gpr   6Hs.166.gpr 6Hs.187.1.gpr   6Hs.194.gpr
          125           178           137            67           182
6Hs.243.1.gpr
          190
```

El problema se encuentra en el canal verde: los valores del background son mayores que los del foreground en el 3ro. 4to. y 5to. arreglo:

```
>   slot(beta7,"maGf")[1, ] #primera fila de la matriz maGf
6Hs.195.1.gpr   6Hs.168.gpr   6Hs.166.gpr 6Hs.187.1.gpr   6Hs.194.gpr
          357           304           189           106           218
6Hs.243.1.gpr
          165
>    slot(beta7,"maGb")[1, ] #primera fila de la matriz maGb
6Hs.195.1.gpr   6Hs.168.gpr   6Hs.166.gpr 6Hs.187.1.gpr   6Hs.194.gpr
          218           175           191           109           224
6Hs.243.1.gpr
          163
```

Realizaremos los cálculos con y sin corrección por background para el primero y el segundo arreglo

Recordemos que con corrección de fondo:

```
A = log2(((Rf-Rb)*(Gf-Gb))1/2)
M = log2((Rf-Rb)/ (Gf-Gb))
```

Seleccionamos los valores foreground de los canales Rojo y Verde del primer arreglo -**6Hs.195.1-**

```
>  #los datos de 6Hs.195.1.gpr están en la primera columna
>  R195f <- slot(beta7,"maRf")[,1] #primera columna de la matriz maRf
>  G195f <- slot(beta7,"maGf")[,1]#primera columna de la matriz maGf
```

Seleccionamos los valores background de los canales Rojo y Verde del primer arreglo

```
>  R195b <- slot(beta7,"maRb")[,1] #primera columna de la matriz maRb
>  G195b <- slot(beta7,"maGb")[,1]#primera columna de la matriz maGb
```

Seleccionamos los valores foreground de los canales Rojo y Verde del segundo arreglo

```
> #los datos de 6Hs.168.gpr  están en la segunda columna
>  R168f <- slot(beta7,"maRf")[,2] #segunda columna de la matriz maRf
>  G168f <- slot(beta7,"maGf")[,2]#segunda columna de la matriz maGf
```

Seleccionamos los valores background de los canales Rojo y Verde del segundo arreglo

```
> R168b <- slot(beta7,"maRb")[,2] #segunda columna de la matriz maRb
```

```
> G168b <- slot(beta7,"maGb")[,2]#segunda columna de la matriz maGb
```

Transformamos los datos por logaritmo-base 2, sin corrección por background

```
> LR195f <- log2( R195f )
> LG195f <- log2( G195f )
>
> LR168f <- log2( R168f )
> LG168f <- log2( G168f )
```

Transformamos los datos por logaritmo-base 2, con corrección por background

```
> LR195 <- log2( R195f- R195b )
Warning message:
NaNs produced in: log(x, base)
> LG195 <- log2( G195f - G195b)
Warning message:
NaNs produced in: log(x, base)
>
> LR168 <- log2( R168f- R168b )
Warning message:
NaNs produced in: log(x, base)
> LG168 <- log2( G168f-G168b )
Warning message:
NaNs produced in: log(x, base)
```

Comparo los cálculos de M y A sin y con corrección por background

```
> a <- cbind(0.5*(LR195f +LG195f ),0.5*(LR168f +LG168f ))
> a1 <- cbind(0.5*(LR195 +LG195 ),0.5*(LR168 +LG168 ))
>
> m <- cbind( (LR195f -LG195f ), (LR168f -LG168f ))
> m1 <- cbind( (LR195 -LG195 ), (LR168 -LG168 ))
```

Comparo A

```
> a1[1:2,]# Cálculos “a mano” con corrección
         [,1]     [,2]
[1,] 6.895683 7.298095
[2,] 5.965369 6.322153
> a [1:2,] # Cálculos “a mano” sin corrección
         [,1]     [,2]
[1,] 8.153164 8.389654
[2,] 7.945132 7.984637
> intensid[1:2,1:2]            # intensid <-  maA(beta7) usando maA
     6Hs.195.1.gpr 6Hs.168.gpr
[1,]      6.895683    7.298095
[2,]      5.965369    6.322153
```

Comparo M

```
> m1[1:2,]# Cálculos “a mano” con corrección
```

```
            [,1]      [,2]
[1,] -0.44651573 0.5737352
[2,]  0.06926266 0.5555187
> m [1:2,] # Cálculos "a mano" sin corrección
           [,1]      [,2]
[1,] -0.6532318 0.2834539
[2,] -0.4893848 0.2162402


> cocientes[1:2,1:2]                  #cocientes <-maM(beta7)
       6Hs.195.1.gpr 6Hs.168.gpr
[1,]   -0.44651573   0.5737352
[2,]    0.06926266   0.5555187
```

Veamos que efectivamente la función **maLR** devuelve una matriz de logaritmos de (intensidades - background) para el canal rojo.

```
> cbind(LR195,LR168)[1:2,]
        LR195    LR168
[1,] 6.672425 7.584963
[2,] 6.000000 6.599913

> difLR[1:2,1:2]                       #  difLR <-maLR(beta7)
     6Hs.195.1.gpr 6Hs.168.gpr
[1,]      6.672425    7.584963
[2,]      6.000000    6.599913


> cbind(LG195,LG168)[1:2,]
        LG195    LG168
[1,] 7.118941 7.011227
[2,] 5.930737 6.044394

> difLG[1:2,1:2]                        # difLG<- maLG (beta7)
     6Hs.195.1.gpr 6Hs.168.gpr
[1,]      7.118941    7.011227
[2,]      5.930737    6.044394
>
```

5.4 Más información

Puede obtenerse más información sobre la estructura de los objetos del paquete marray utilizando las viñetas.

```
> library(marray)
Loading required package: limma
Loading required package: tkWidgets
Loading required package: widgetTools
Loading required package: tcltk
```

```
Loading Tcl/Tk interface ... done
Loading required package: Biobase
Loading required package: tools

Welcome to Bioconductor

  Vignettes contain introductory material. To view, type
  'openVignette()'. To cite Bioconductor, see
  'citation("Biobase")' and for packages 'citation(pkgname)'.

Loading required package: DynDoc
> openVignette()
Please select a vignette:

 1: Biobase - An introduction to Biobase and ExpressionSets
 2: Biobase - Bioconductor Overview
 3: Biobase - esApply Introduction
 4: Biobase - Notes for eSet developers
 5: Biobase - Notes for writing introductory 'how to' documents
 6: limma - Limma Vignette
 7: marray - marray Normalization
 8: marray - marray Overview
 9: marray - marrayClasses Overview
10: marray - marrayClasses Tutorial (short)
11: marray - marrayInput Introduction
12: marray - marrayPlots Overview
13: tkWidgets - tkWidgets contents
14: tkWidgets - tkWidgets importWizard
15: widgetTools - widgetTools Introduction

Selección: 9
Opening C:/ARCHIV~1/R/R-25~1.0/library/marray/doc/marrayClasses.pdf
>
```

Seleccionamos la opción 9, se abre una ventana con el documento "marrayClasses.pdf" que describe detalladamente la estructura de los objetos del paquete **marray**

Estructuras de datos de marray

Objeto	slotNames(Objeto)	Métodos	Funciones
marrayRaw	maRf maGf maRb maGb maW maLayout maGnames maTargets maNotes	maA maM maLR maLG	maGeneTable
maLayout	maNgr maNgc	maPrintTip maGridCol	maCompPlate maCompCoord

	maNsr **maNsc** **maNspots** **maSub** **maPlate** **maControls** **maNotes**	**maGridRow** **maSpotCol** **maSpotRow**	**maCompInd** **maCoord2Ind** **maInd2Coord**
maGnames	**maLabels** **maInfo** **maNotes**		
maTargets	**maLabels** **maInfo** **maNotes**		