

R操作生物晶片的基本分析流程

Outline

Biological experiments

Raw file_list | *condition_R and condition_G*

Preprocess microarray data

Remove statistical error

Statistical test to find DEGs


*(**D**ifferentially **E**xpressed **g**enes)*

Target

Biological meaning

Outline

*Biological experiments
(WSSV infection of *P. vannamei*)*



file_list	condition_list
5489_251841210001_1.txt	10hr_0hr
5491_251841210001_2.txt	12hr_1hr
5493_251841210001_3.txt	14hr_2hr
5495_251841210001_4.txt	18hr_3hr
5497_251841210002_1.txt	24hr_4hr
5499_251841210002_2.txt	30hr_5hr
5501_251841210002_3.txt	36hr_6hr
5503_251841210002_4.txt	48hr_8hr

Preprocess microarray data



1. Choice intensity index (Mean or Median)
2. Filtration (0hr_virus intensity)
3. Background adjustment (No substrate)
4. Normalization (**Quantile Normalization**)

*Statistical test to find DEGs (**D**ifferentially **E**xpressed **g**enes)*



ANOVA
Cluster

Target

1. Find the **expression pattern of virus gene**:
→ try to find the mechanism of virus attach
2. Find the **changed** genes of host:
→ which pathway that virus **use** it
→ which pathway that virus **shutdown** it

準備工作:

1. 當然是要先準備好R_version2.62，這一版我比較推薦
2. 安裝好limma跟marray這兩個好用的package也是一定要的
3. 將microarray的raw file放在"c:\WSSV\rawfile\"下吧...方便後續作業
4. 準備記事本，直接用這個寫code並紀錄注意事項...



前置說明:

1. 因為分析過程牽涉許多函數的使用，故在這裡先使用寫好的綜合函數簡化每一個階段所需要寫的code
2. 如果有較複雜的指令會直接寫在slide上…可以直接copy貼上R執行
3. 只用比較簡單的流程帶大家走，R裡面還有許多的function大家可以去了解
4. 先將黑框中的code複製到自己的記事本中

```
library("limma")  
library("marray")  
source("http://eln.iis.sinica.edu.tw/lms/files/function_20080812.txt")  
rawdir<- "C:/WSSV"
```

Raw file format

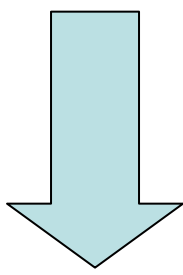
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	ATF	1															
2	29	56															
3	Type=GenePix Results 3																
4	DateTime=2008/03/24 14:26:48																
5	Settings=																
6	GalFile=C:\Documents and Settings\Administrator\桌面\seeing20070913.gal																
7	PixelSize=10																
8	Wavelengths=635532																
9	ImageFiles=Not SavedNot Saved																
10	NormalizationMethod=None																
11	NormalizationFactors=11																
12	JpegImage=D:\Room818\YAN\20080324\20080324_0hr_1.jpg																
13	StdDev=Type 1																
14	RatioFormulations=W2/W1 (532/635)																
15	FeatureType=Circular																
16	Barcode=																
17	BackgroundSubtraction=LocalFeature																
18	ImageOrigin=720, 8720																
19	JpegOrigin=1550, 9770																
20	Creator=GenePix Pro 6.0.1.25																
21	Scanner=GenePix 4100A 01 [114714]																
22	FocusPosition=0																
23	Temperature=24.4141																
24	LinesAveraged=1																
25	Comment=																
26	PMTGain=682537																
27	ScanPower=100100																
28	LaserPower=3.180853.2431																
29	Filters=670DF40575DF35																
30	ScanRegion=72,872,2044,4660																
31	Supplier=																
32	Block	Column	Row	Name	ID	X	Y	Dia.	F635 Medi	F635 Mean	F635 SD	F635 CV	B635	B635 Medi	B635 Mean	B635 SD	B635 CV
33	1	1	1	PmTwI08F A04		1780	10000	110	3355	3112	775	24	101	101	104	31	
34	1	2	1	PmTwI08F A04		1960	10000	110	3000	2899	668	23	102	102	106	31	
35	1	3	1	PmTwI08F A04		2150	10000	100	3121	2957	924	31	103	103	108	33	
36	1	4	1	PmTwI11F A08		2320	10010	110	480	541	204	37	110	110	111	29	
37	1	5	1	PmTwI11F A08		2510	10000	110	467	530	211	39	114	114	120	39	

Raw file format(2)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Block	Column	Row	Name	ID	X	Y	F635 Median	F635 Mean	B635 Median	B635 Mean	F532 Median	F532 Mean	B532 Median	B532 Mean
2	1	1	1	PmTwI08H	A04	1780	10000	3355	3112	101	104	2818	2738	101	104
3	1	2	1	PmTwI08H	A04	1960	10000	3000	2899	102	106	2553	2539	108	110
4	1	3	1	PmTwI08H	A04	2150	10000	3121	2957	103	108	2598	2494	111	113
5	1	4	1	PmTwI11F	A08	2320	10010	480	541	110	111	564	581	113	116
6	1	5	1	PmTwI11F	A08	2510	10000	467	530	114	120	563	601	114	123
7	1	6	1	PmTwI11F	A08	2690	10010	488	510	114	120	550	579	112	121
8	1	7	1	PmTwI13H	A12	2900	9990	195	199	114	118	246	250	110	115
9	1	8	1	PmTwI13H	A12	3080	9990	180	191	114	119	232	236	107	114

在array上的位置 annotation

紅光與綠光的gene intensity



經由read.anno.agilent
Output出gene的名字與位置到array_info.txt
及gene的重複次數與屬性到gene_probe_index.txt

read.anno.agilent(1)

從raw檔案中抽取probe的位置及屬性的指令

```
read.anno.agilent("rawfile/20080324_0hr_1.gpr", skip_line=31, col_num=1:5,  
                  probe_col=4:5, folder="array_info")
```

Raw檔所在位置

讀入table data所需空行數

讀成array_info.txt的位置

計算probe重複次數為array_info的第幾列

想要寫入的資料夾

成功後會回應這樣的訊息

```
> read.anno.agilent("rawfile/20080324_0hr_1.gpr", skip_line=31, col_num=1:5, $  
[1] columns of array_info  
[1] "Block" "Column" "Row"      "Name"   "ID"  
[1] columns of gene_probe_index  
[1] "ProbeName" "Type"      "repeat_time"
```


read.anno.agilent(2)

接下來我們可以用以下兩個指令去查看輸出的檔案：

```
array_info <- read.table(paste(rawdir, "array_info", "array_info.txt", sep="\\"),  
                        header=TRUE, sep = "\\t")  
gene_probe_index <- read.table(paste(rawdir, "array_info", "gene_probe_index.txt",  
                                    sep="\\"), header=TRUE, sep = "\\t")
```

試著用unique指令看一下資料的結構

```
unique(gene_probe_index[,1])  
unique(gene_probe_index[,2])  
unique(gene_probe_index[,3])
```

variation(1)

從以上的資料我們可以知道，通常microarray上的probe會有重複的設計(這是在gene量較少的物種中)，而每一個點的intensity又有mean與median兩種數值，我們可以利用重複probe intensity在mean及median的再現性程度來決定我們要選用mean或median(如果以object的方式讀入，通常是以mean來做probe的intensity)

H	I	J	K	L	M	N	O
F635 Median	F635 Mean	B635 Median	B635 Mean	F532 Median	F532 Mean	B532 Median	B532 Mean
3355	3112	101	104	2818	2738	101	104
3000	2899	102	106	2553	2539	108	110
3121	2957	103	108	2598	2494	111	113
480	541	110	111	564	581	113	116
467	530	114	120	563	601	114	123
488	510	114	120	550	579	112	121
195	199	114	118	246	250	110	115
180	191	114	119	232	236	107	114

Foreground

background

Foreground

background

variation(2)

Gene的名字為array_info\$Name內的資料

Array格式為gene_pix

Raw檔存在"rawfile"資料夾中

```
variation(array_type="gene_pix", raw_folder="rawfile", gene_name=array_info$Name,  
file_list="20080324_0hr_1.gpr", condition_list="a1_b1", skip_line=31,  
folder="choice_the_index")
```

共有"20080324_0hr_1.gpr"等...1個檔案

想要寫入的資料夾

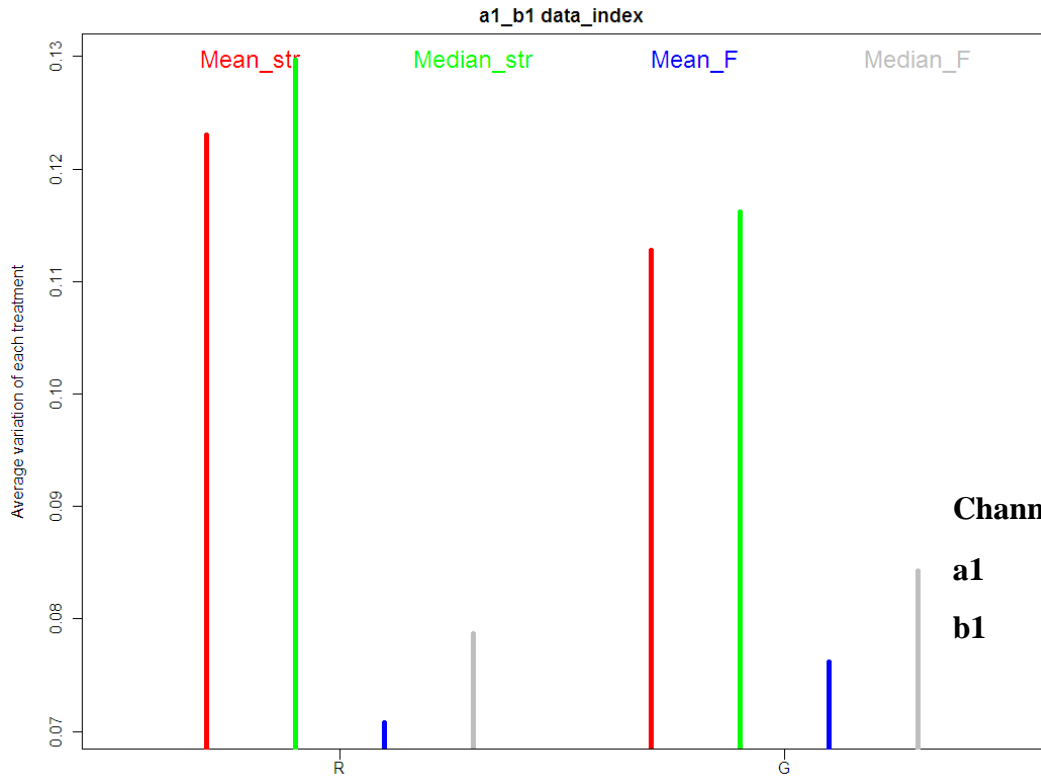
紅光及綠光的代表名稱為a1及b1

讀入table data所需空行數

variation(3)

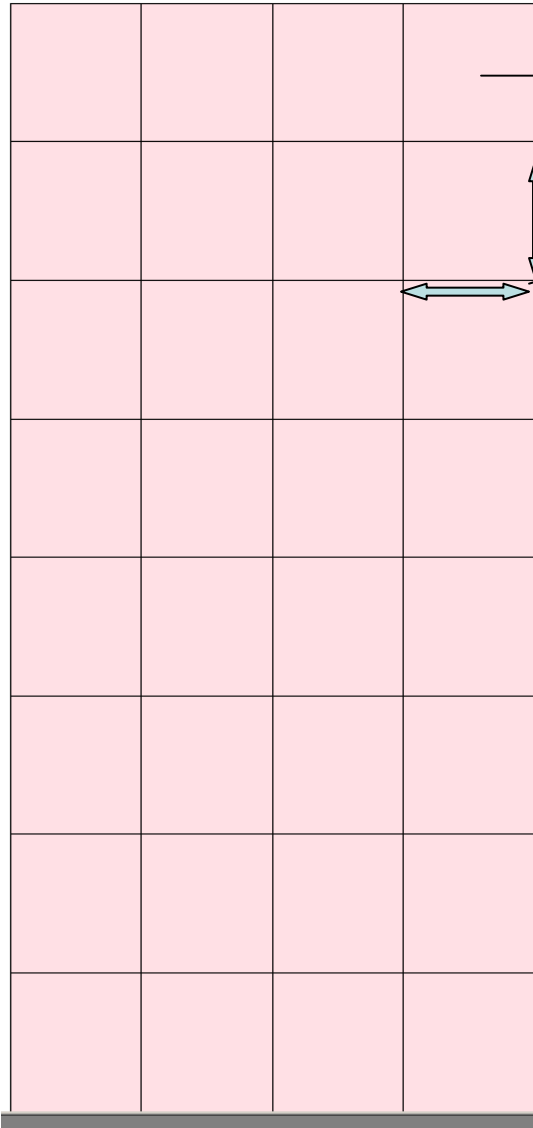
上一頁的指令會output出一張圖/每片array，還有一個數字表，其中的數值代表重複probe intensity在同一張array的變異程度，做四種數值的統計，分別為

- Mean_str:用mean值減掉background值當作intensity
- Median_str:用median值減掉background值當作intensity
- Mean_F:用mean值不減掉background值當作intensity
- Median_F:用median值不減掉background值當作intensity



Channel	Mean_str	Median_str	Mean_F	Median_F
a1	0.123006	0.129668	0.070779	0.07868
b1	0.112745	0.116168	0.076187	0.084245

image_plot.genepix(1)



→ **Block=32**

→ **Row=23**

→ **Column=23**

Spot=32*23*23=16928

右圖是genepix的microarray format，基本上是由32個blocks，每個block又有529個probe點所組成
因為我們已經決定了intensity的參考值，所以接下來便是檢視image會不會有什麼錯誤，同時可以將array intensity做輸出

image_plot.genepix(2)

Raw檔所在位置

Image繪製的各項參數

Array檔名及代稱

讀入table data所需空行數

```
image_plot.genepix(raw_folder="rawfile", image_factor=c(nrow(array_info),8,4,23,23),  
  file_list="20080324_0hr_1.gpr", condition_list="a1_b1",  
  skip_line=31, spot_index="median", folder="choice_the_index",  
  bg_thre=NULL)
```

Probe intensity的參考項

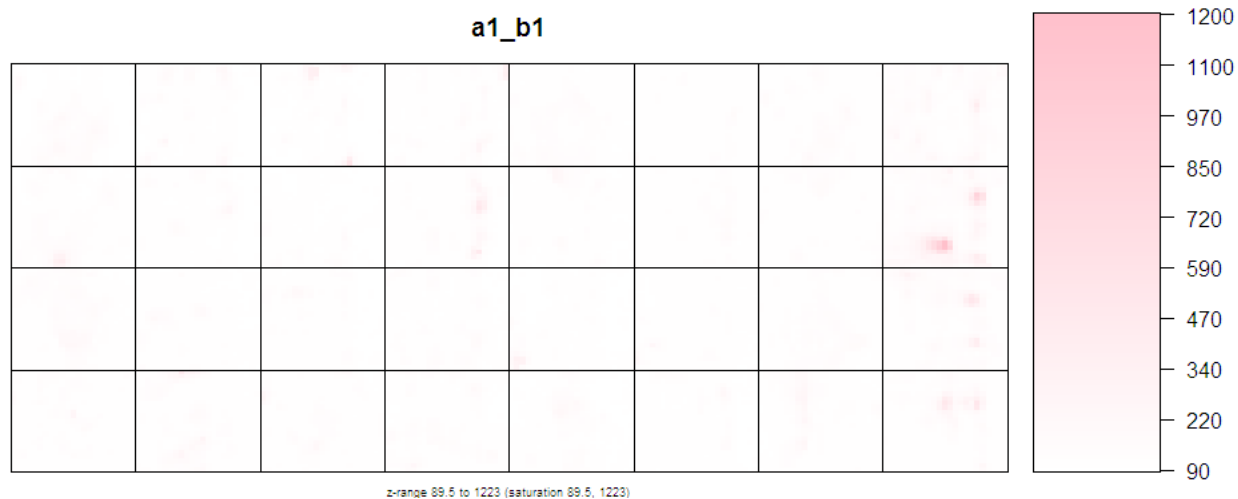
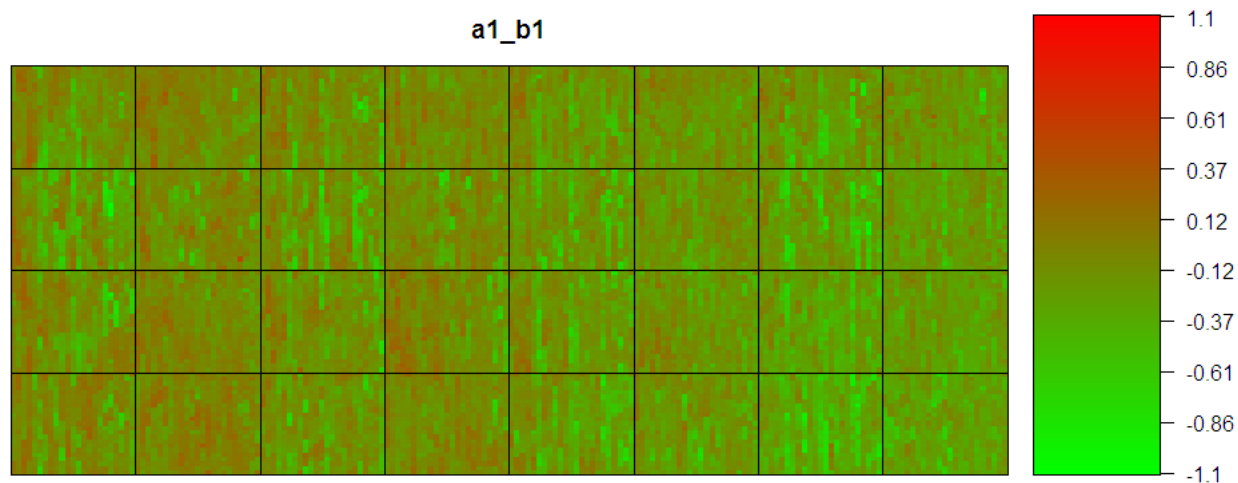
想要寫入的資料夾

限制background的最大值

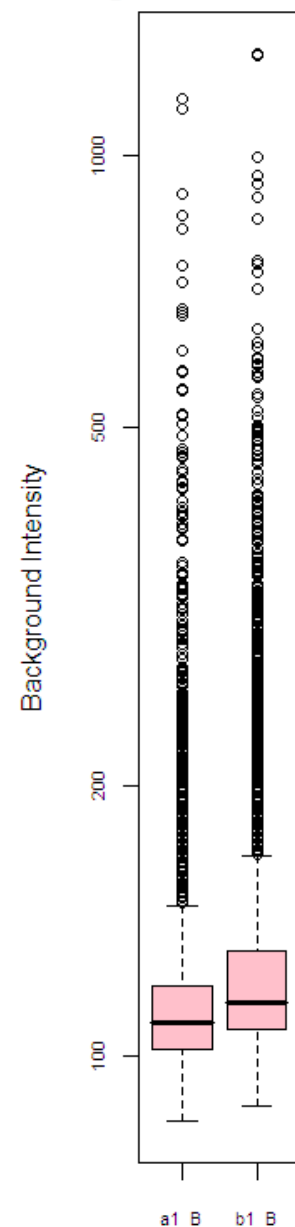
image_plot.genepix(3)

這個指令會output出三種圖

1. R/G foreground的log2數值圖
2. background的數值圖
3. 各個channel的background分布圖



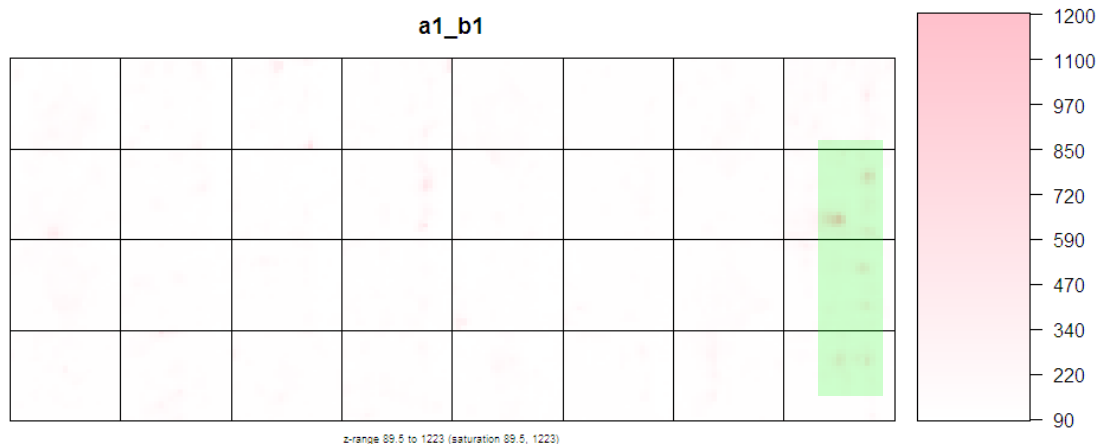
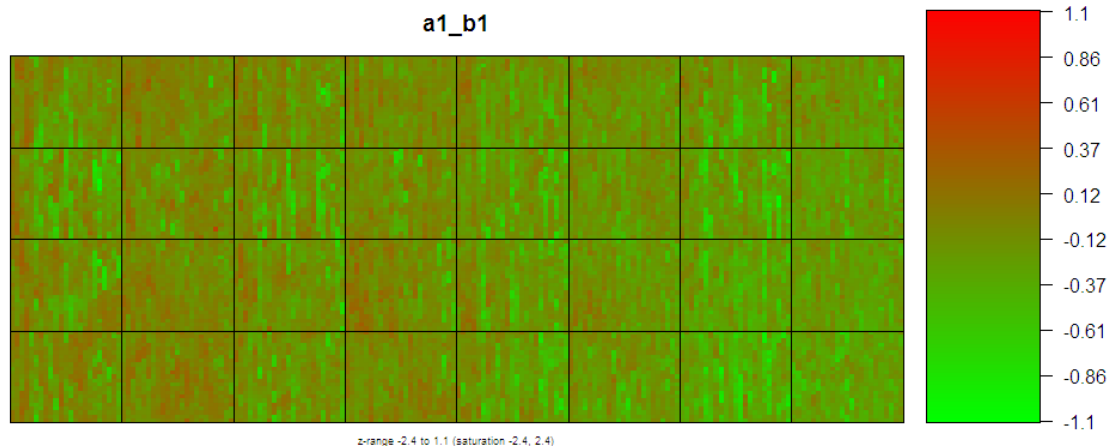
Background Distrib



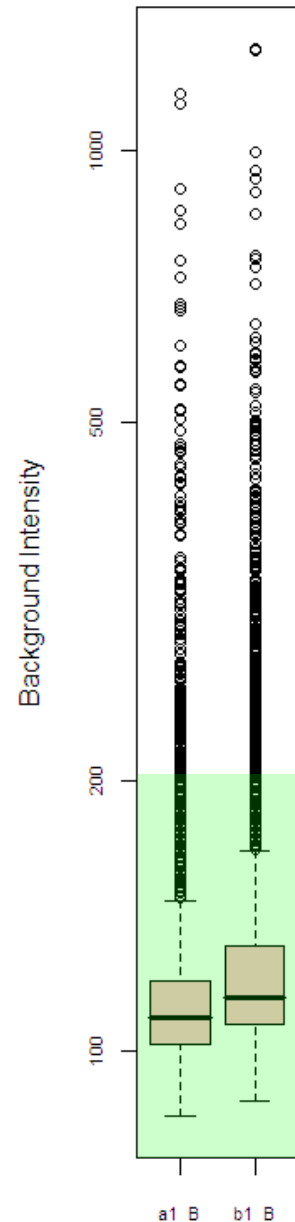
image_plot.genepix(4)

檢視重點：

1. 在image中可一看一下有沒有區域性相同顏色或缺值，如果有可能是遭受的四理性的破壞
2. 另外在background的intensity範圍也不過大，這樣表示probe spot的定位可能不甚準確



Background Distrib

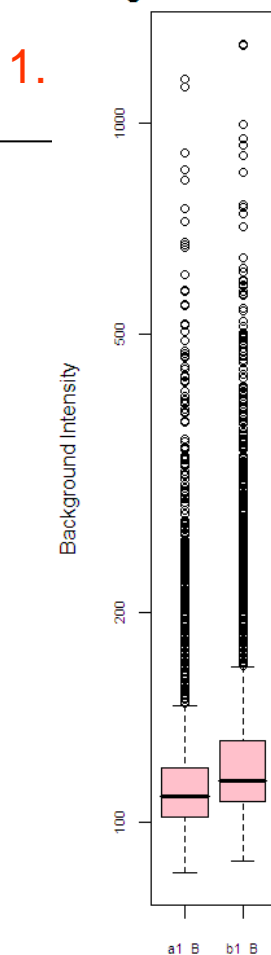


GlobalNorm(1)

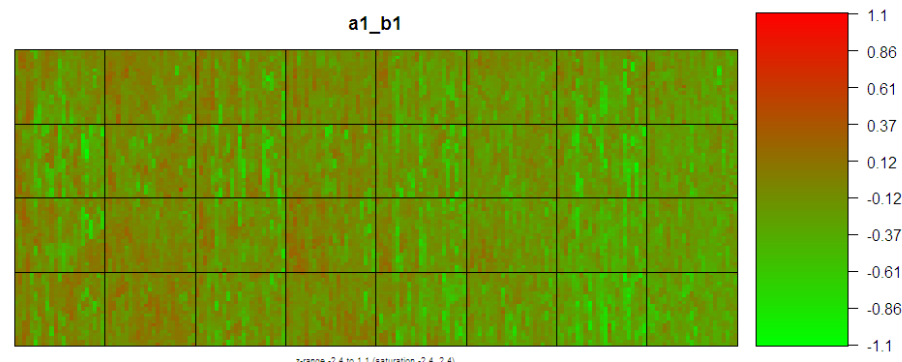
Preprocess的最後步驟也是最重要的步驟就是Normalization，Normalization的主要目的是在於將操作array所產生的error減到最低，而主要的error有下列幾項：

1. *Dyes effects*
2. *Scanning parameter...between arrays*
3. *Print-tip different*
4. *Spatial effects*

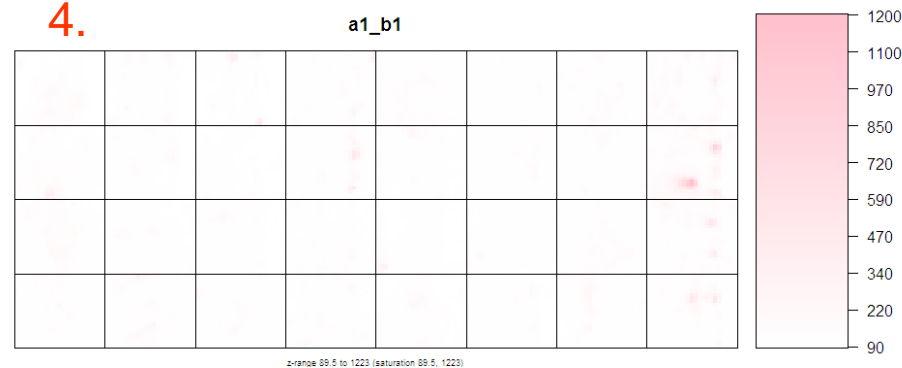
Background Distrib



3.



4.



GlobalNorm(2)...within array

Within array normalization:

在許多marray有許多函數就是針對normalization所開發的，在下面我會一一介紹，但因為這些函數是使用marray的物件格式，所以我們必須以這樣的格式讀入data...

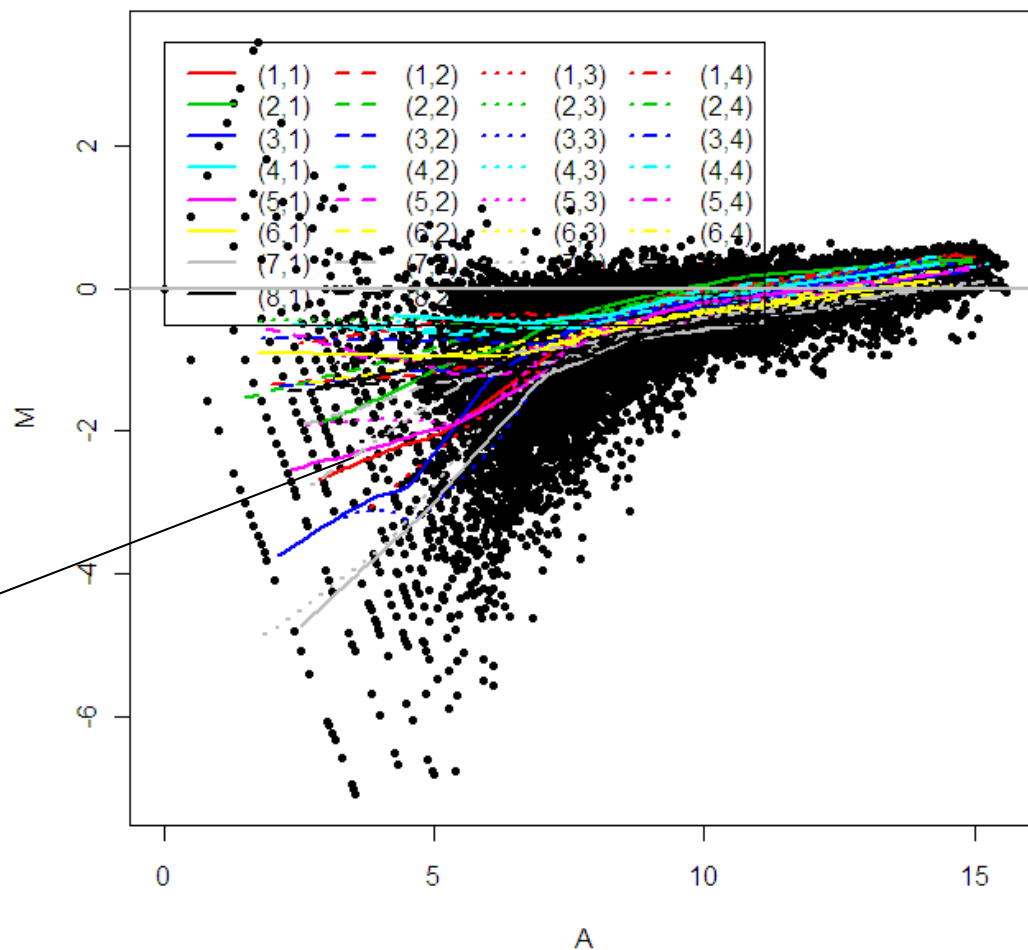
但在package normalization方法中有一個假設，就是大部分的gene在實驗及控制組間不無差別

```
data <- read.GenePix(fnames = "20080324_0hr_1.gpr", path =  
"C:/WSSV/rawfile", name.Gf = "F532 Mean",  
name.Gb = "B532 Mean", name.Rf = "F635 Mean", name.Rb = "B635 Mean")
```

GlobalNorm(3)...within array

```
maPlot(data)  
savePlot(paste(rawdir, "normalization", "maplot_1", sep="\\"), type="png")
```

最基本的maplot，這是在array分析中常看到的圖，主要是在看R/G(M)是否會隨著intensity的總強度(A)而改變，最好的狀態是大部分的gene分布是隨著 $y=0$ 的直線分布，而normalization的目的也是一樣



GlobalNorm(4)...within array

有3種normalization的函數可以處理:

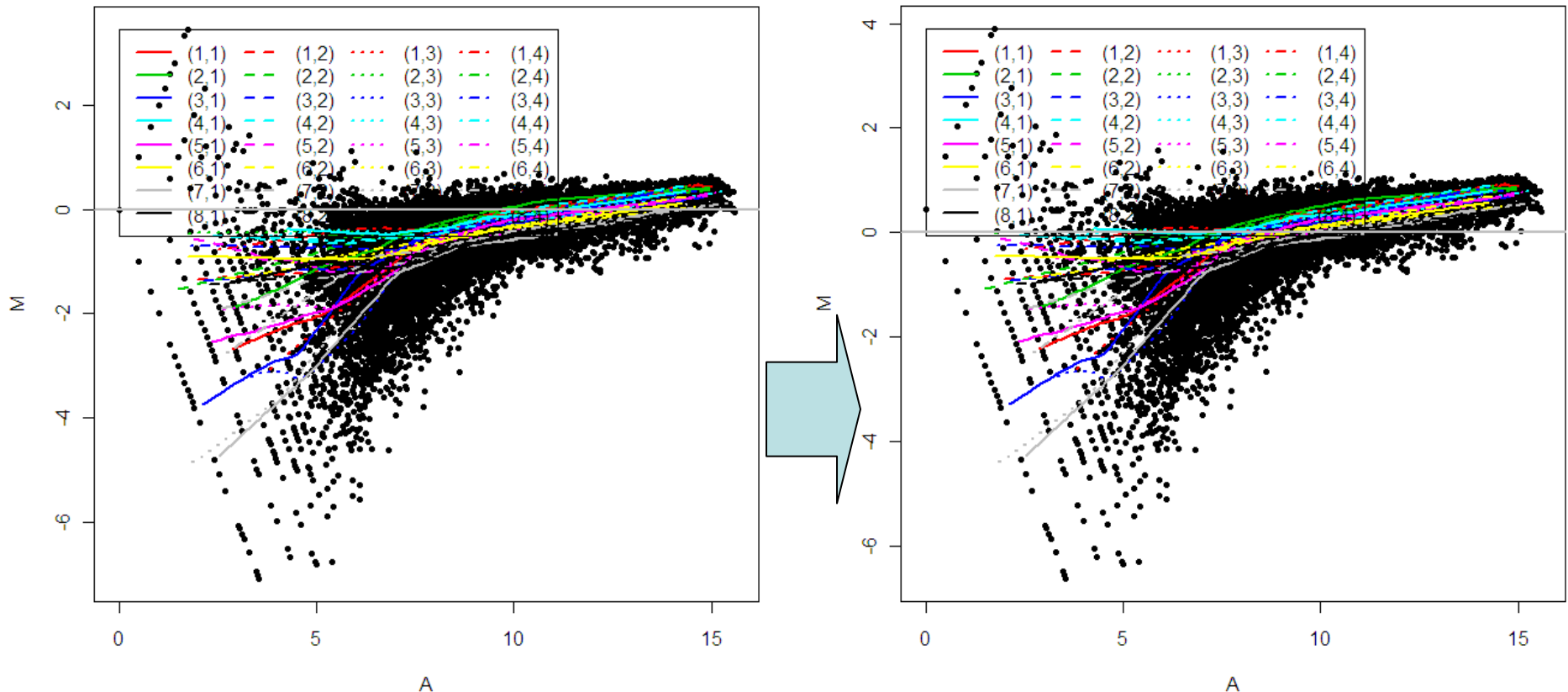
Medium Normalization

Loess Normalization

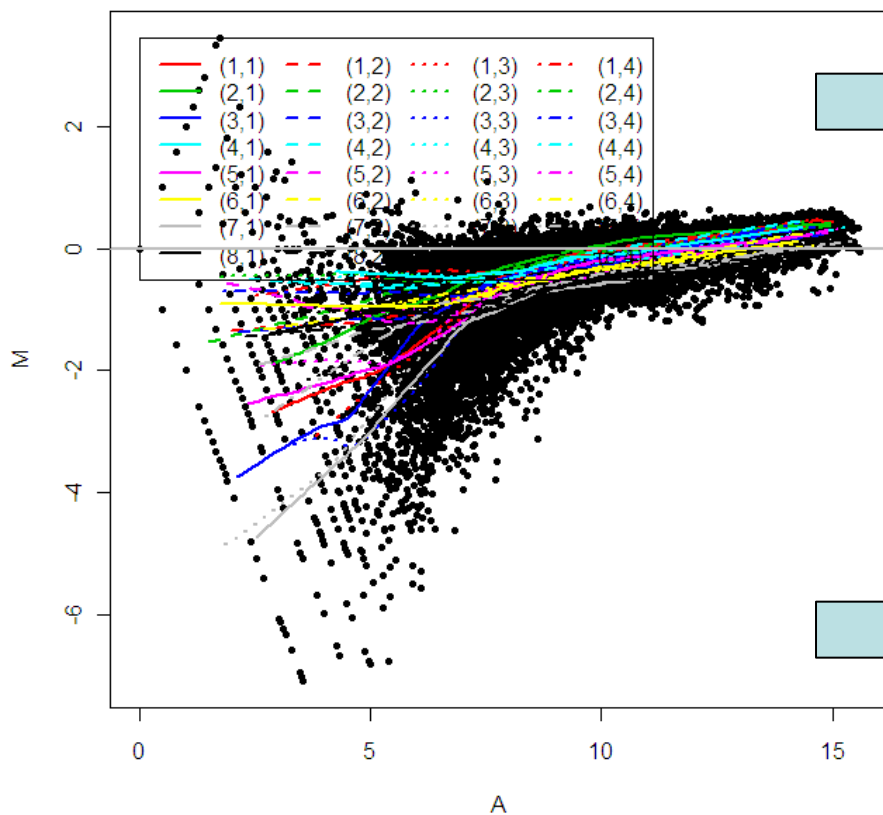
Print-tip Loess Normalization

```
data_M_Norm <- maNorm(data, norm="m")  
data_L_Norm <- maNorm(data, norm="l")  
data_P_Norm <- maNorm(data, norm="p")
```

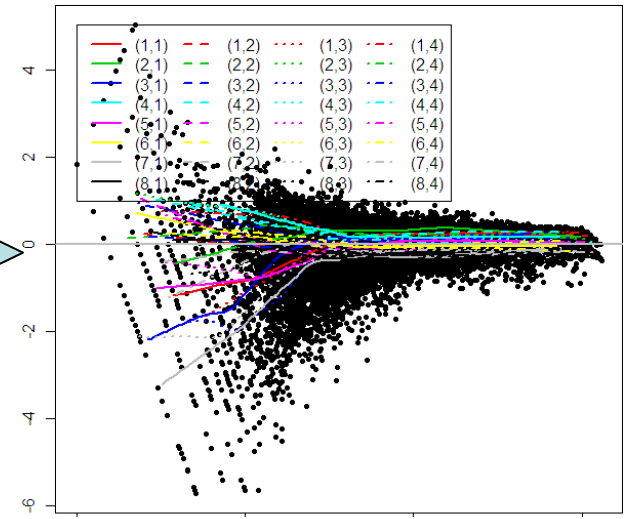
Medium Normalization



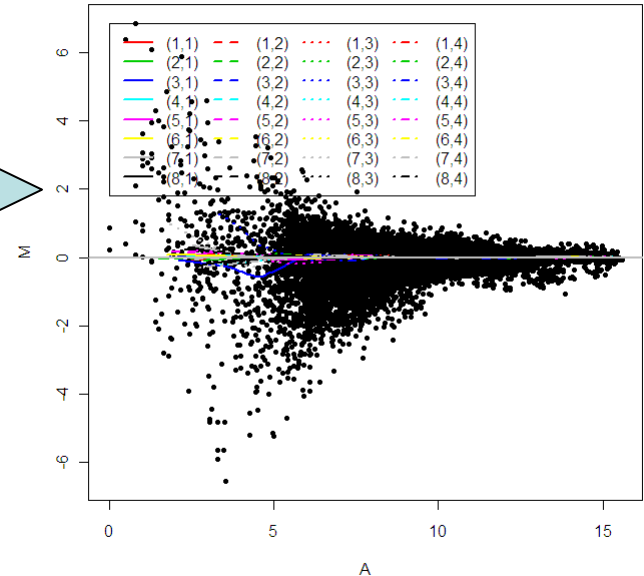
GlobalNorm(5)...within array



Loess Normalization



Print-tip Loess Normalization

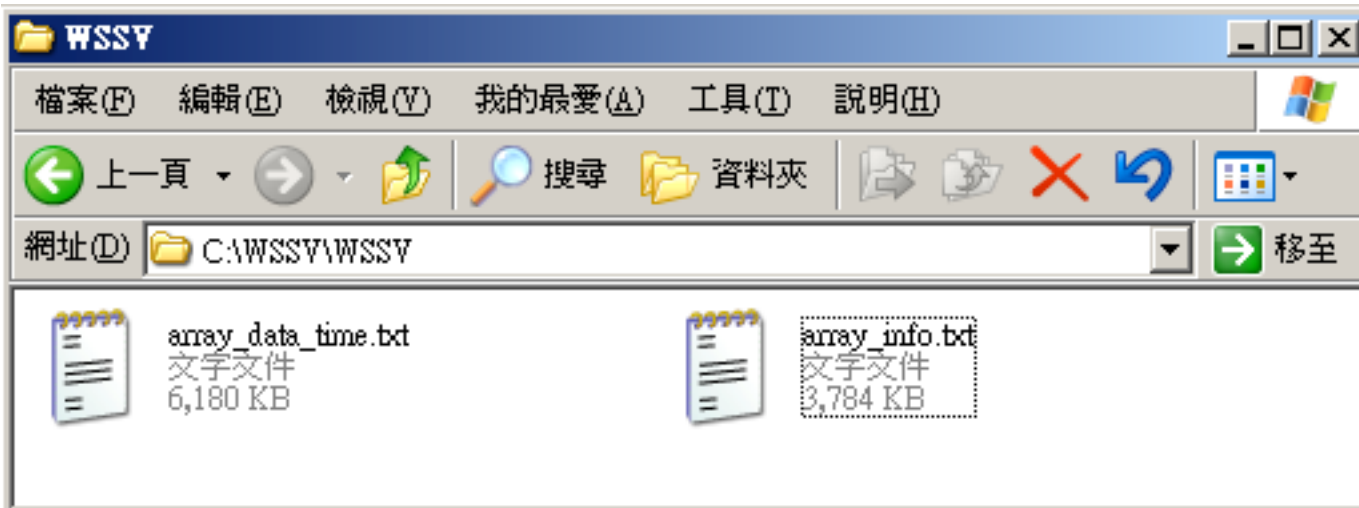


GlobalNorm(6)...between array

Between array normalization:

在另一方面，則是array與array之間的normalization，通常做array實驗時，還是會以多個array為主去做比較，所以多次取樣的error就會在此時顯現，在這方面通常會以簡單的Quantile Normalization去做，但比較常用的做法則是以表現較無變化的house keeping gene做normalization的依據

在這一部分我謹針對Quantile Normalization做說明，並以羅教授實驗室的data做範例



Outline

Biological experiments
(WSSV infection of *P. vannamei*)

file_list	condition_list
5489_251841210001_1.txt	10hr_0hr
5491_251841210001_2.txt	12hr_1hr
5493_251841210001_3.txt	14hr_2hr
5495_251841210001_4.txt	18hr_3hr
5497_251841210002_1.txt	24hr_4hr
5499_251841210002_2.txt	30hr_5hr
5501_251841210002_3.txt	36hr_6hr
5503_251841210002_4.txt	48hr_8hr

Preprocess microarray data

1. Choice intensity index (Mean)
2. Filtration (0hr_virus intensity)
3. Background adjustment (No substrate)
4. Normalization (Quantile Normalization)

Statistical test to find DEGs (Differentially Expressed genes)

ANOVA
Cluster

Target

1. Find the expression pattern of virus gene:
→ try to find the mechanism of virus attach
2. Find the changed genes of host:
→ which pathway that virus use it
→ which pathway that virus shutdown it

Repeat data type

Total gene: 7325

Gene name type	Repeat times	Amount
SA_	6	6508
SO_	6	45
WSSV001~718	6	698
Spike in (E1A_r60)	32	10
Total	45049	7325

Host gene

Host gene

Virus gene

Control gene

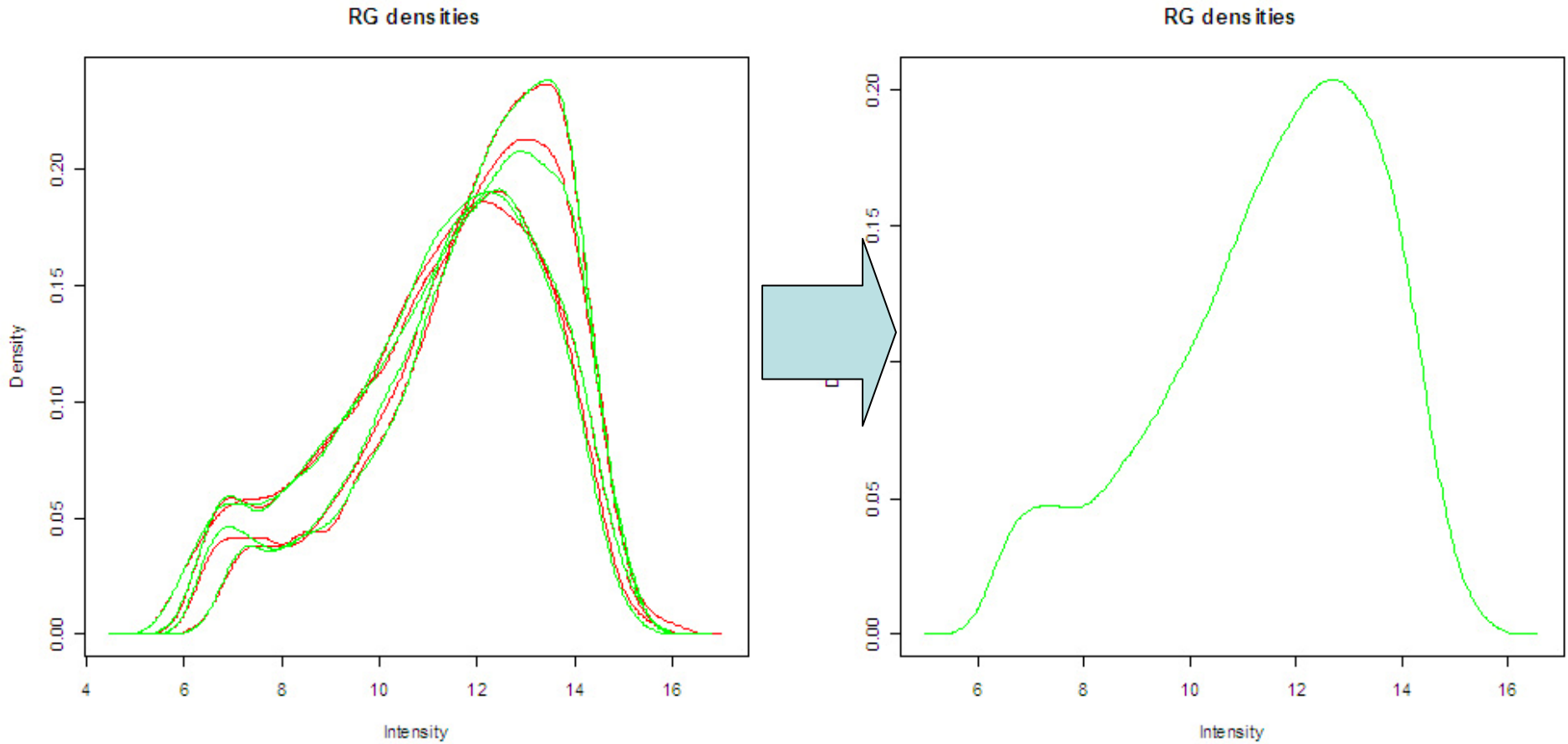
Control type=0, gene_number=43632

Spike-in gene spots=320

WSSV gene spots=4188

GlobalNorm(7)...between array

Quantile Normalization



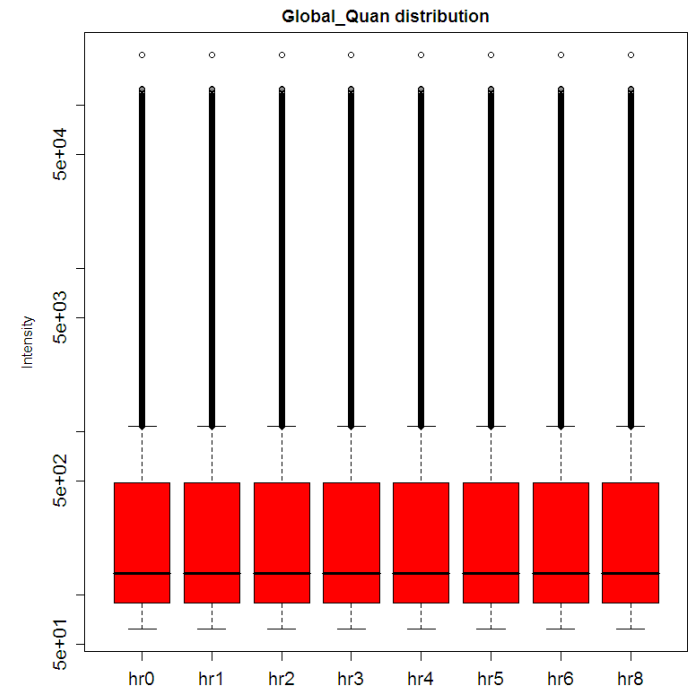
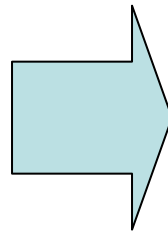
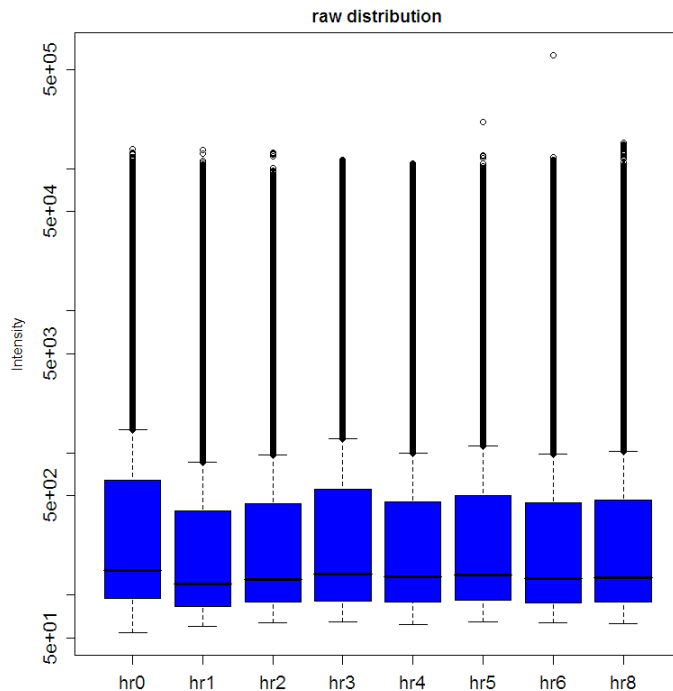
GlobalNorm(8)...between array

首先先將檔案讀入：

```
array_data_time <- read.table(paste(rawdir, "WSSV", "array_data_time.txt",  
sep="\\"), header=TRUE, sep = "\\t")
```

Quantile Normalization

```
GlobalNorm(array_data_time, "Norm_data_time","normalization")
```



ANOVA(1)

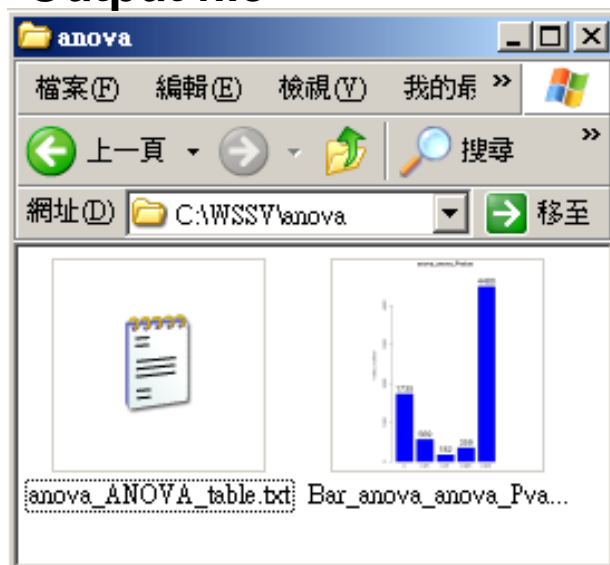
在找出明顯變化gene(*Differentially Expressed genes*)常用的方法就是t. test 跟ANOVA，t. test是針對兩兩比較的統計方式，而ANOVA則是用在time-series data上，test之後每個gene都給一個p-value，再以這個值做篩選即可

指令如下：

```
array_info <- read.table(paste(rawdir, "WSSV", "array_info.txt", sep="\\"),
header=TRUE, sep = "\\t")
Norm_data_time <- read.table(paste(rawdir, "normalization",
"Norm_data_time_data_GQuan.txt", sep="\\"), header=TRUE, sep = "\\t")
factor_gene <- factor(array_info$GeneName)
col_factor <- colnames(array_data_time)
ANOVA.F.P.num(Norm_data_time, factor_gene, col_factor, "anova", "anova")
```

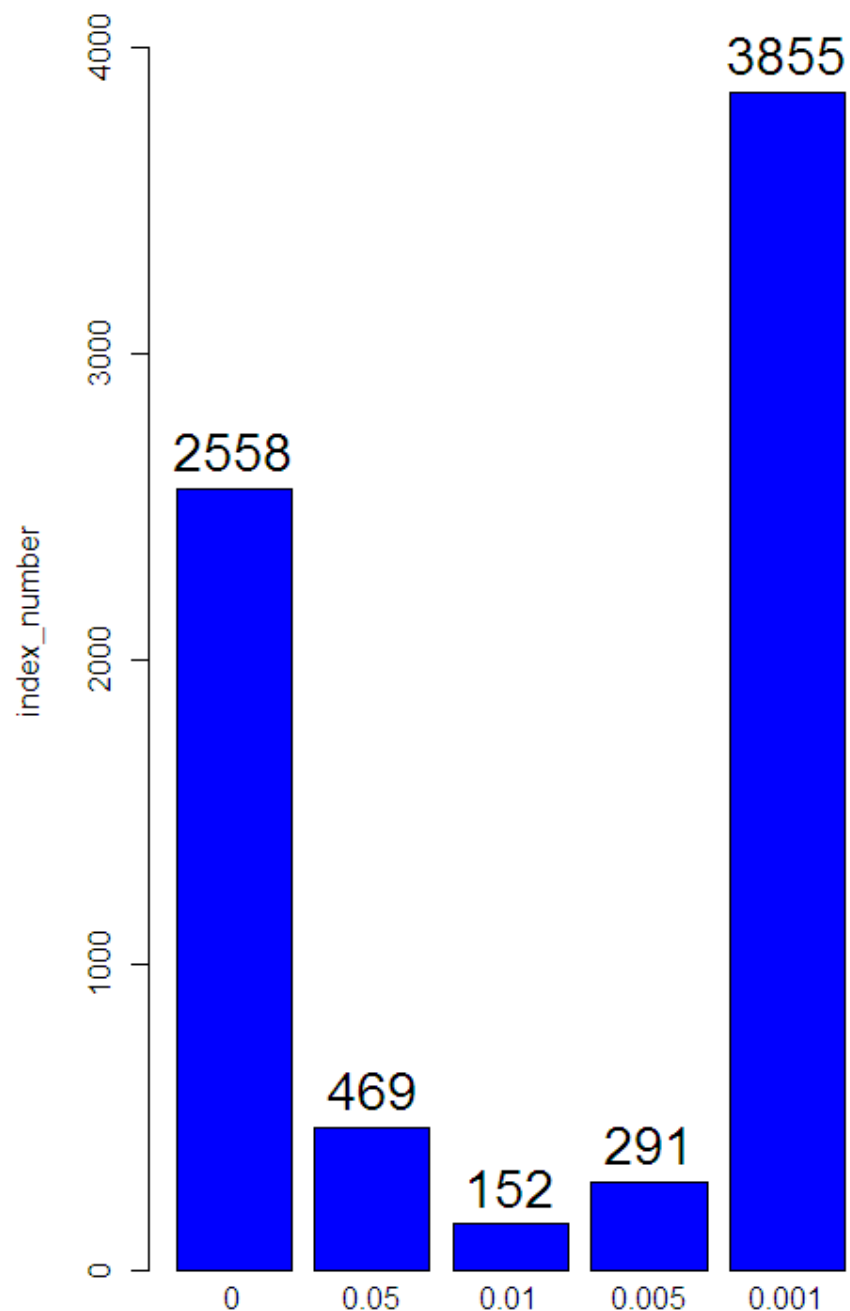
ANOVA(2)

Output file



	A	B	C	D
1	Name	F	P	Label
2	E1A_r60_n9	152.6034	4.15E-86	4
3	E1A_r60_1	151.7611	7.22E-86	4
4	E1A_r60_a20	150.1714	2.07E-85	4
5	E1A_r60_a107	141.6798	6.69E-83	4
6	E1A_r60_3	140.4799	1.55E-82	4
7	E1A_r60_a104	131.1349	1.30E-79	4

anova_anova_Pvalue

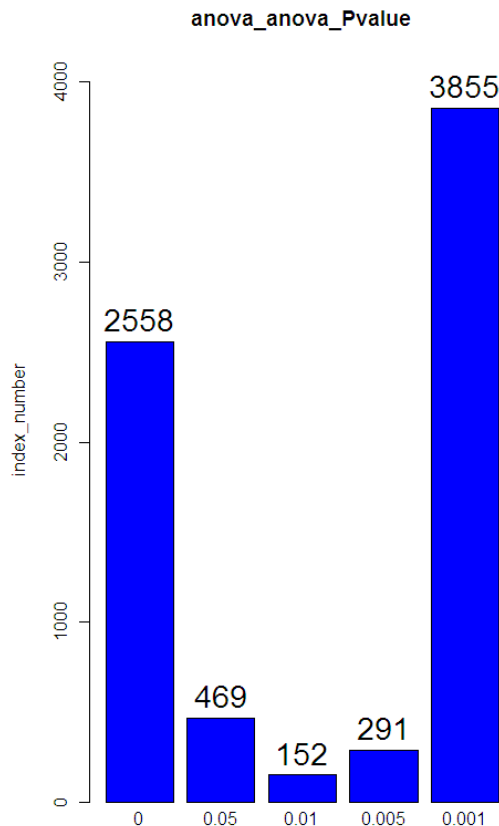


Cluster(1)

當我們挑出了DEGs之後，要做的就是針對這些gene表現做分類，將相似的pattern歸類在一起，這樣對後面biological function上的連結也會有幫助

先將重複的intensity取mean平均：

```
factor_mean <- function(x) {tapply(x,factor_gene,mean)}  
Norm_data_mean <- apply(Norm_data_time, 2,factor_mean)
```



Virus_001 #282

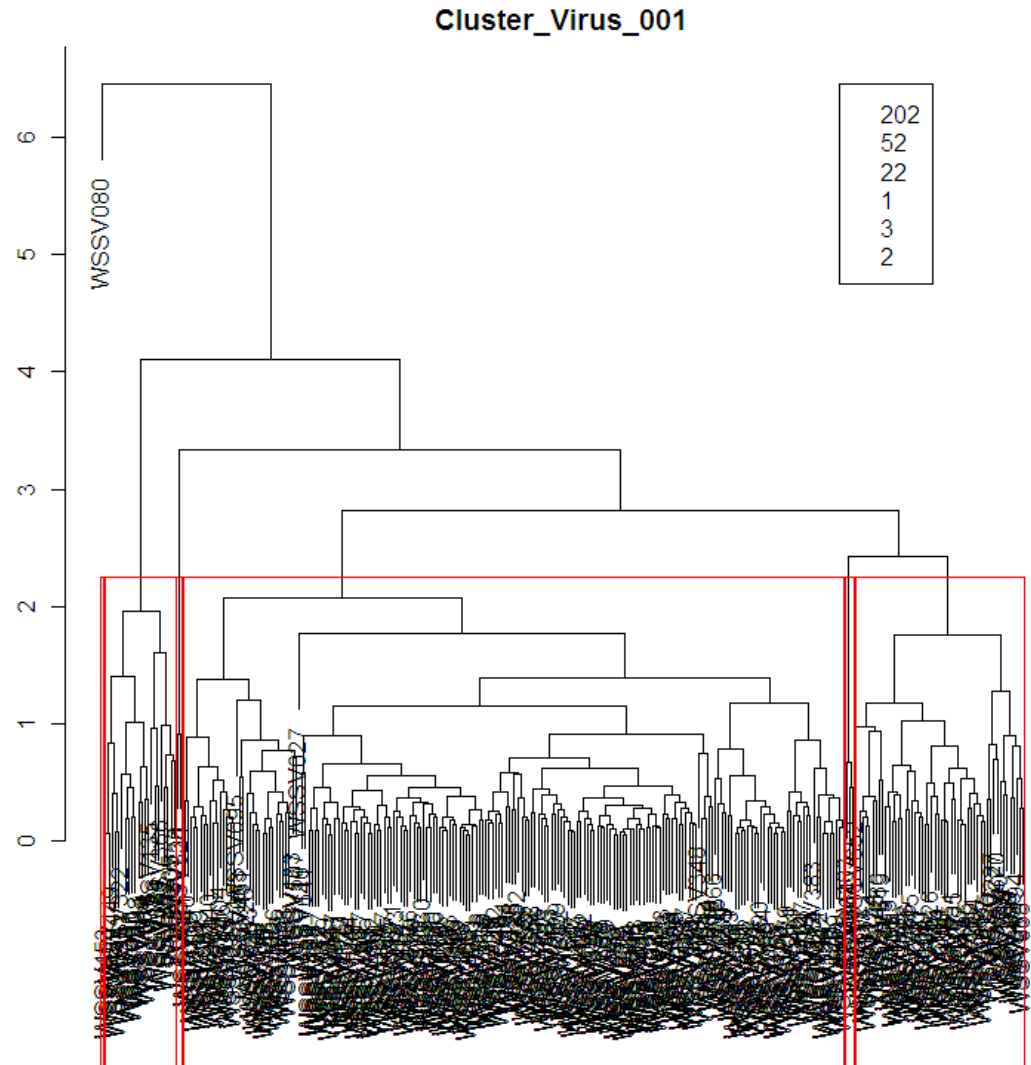
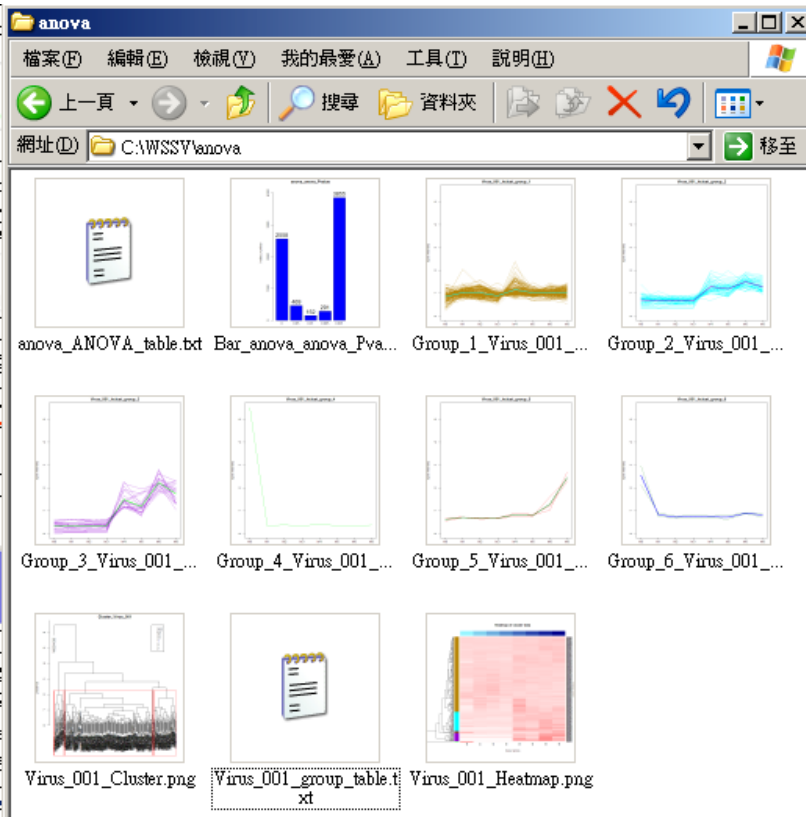
```
Virus_001_index <- substr(anova_table$Name, 1,  
4)=="WSSV" & anova_table$Label==4
```

Host_001 #3545

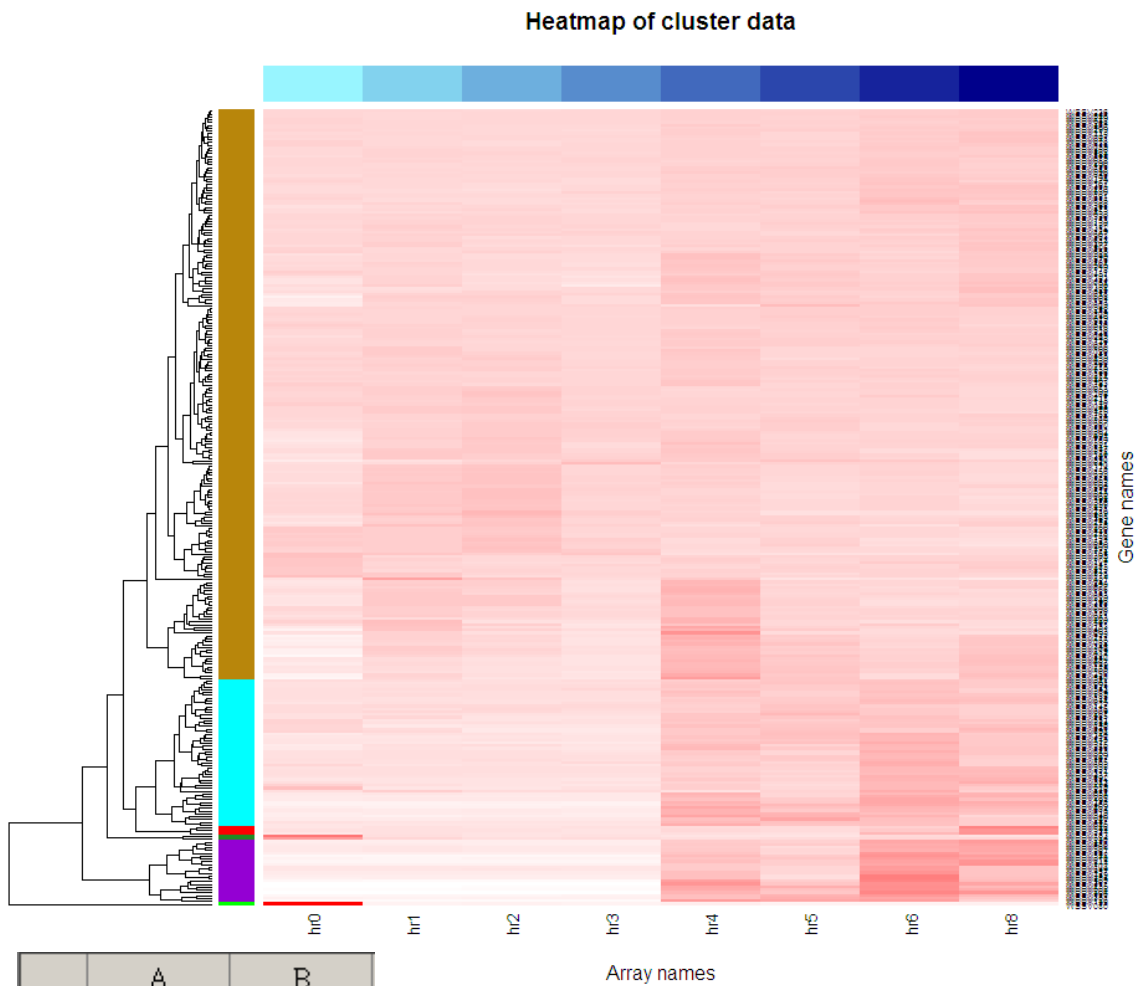
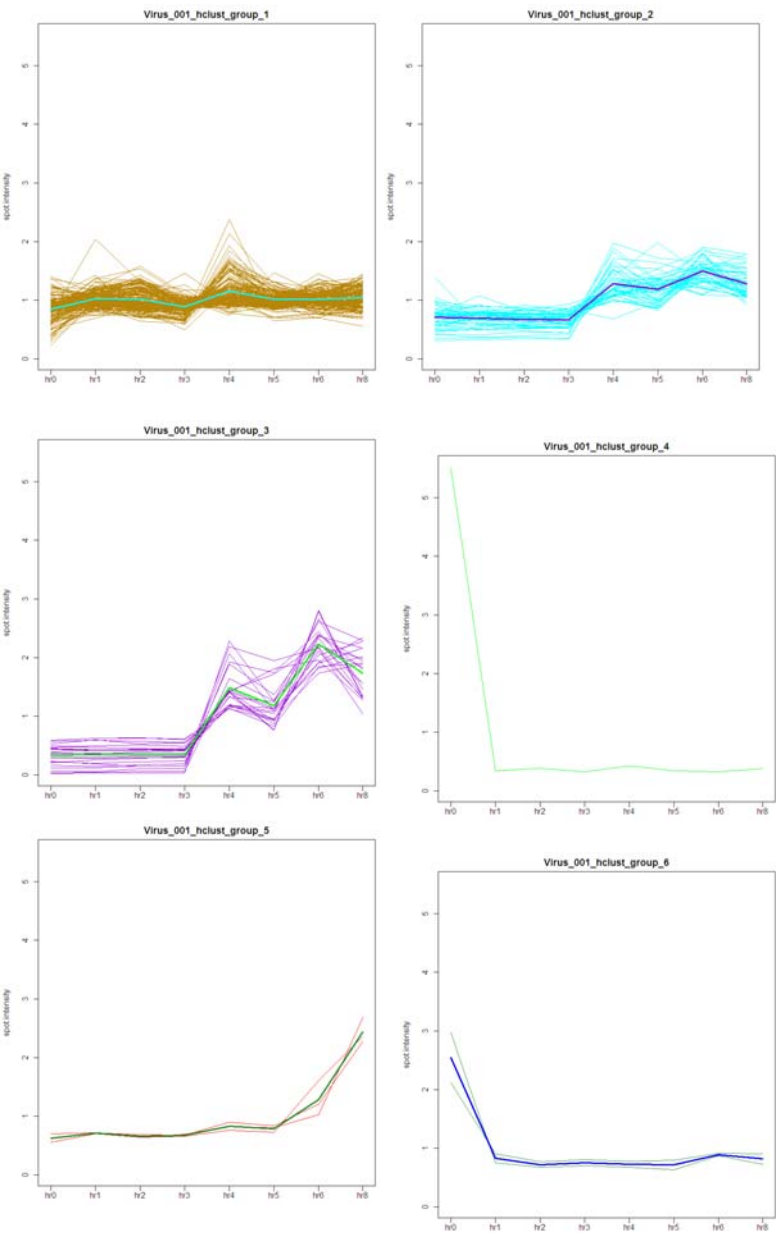
```
Host_001_index <- (substr(anova_table$Name, 1,  
2)=="SA" | substr(anova_table$Name, 1, 2)=="SO") &  
anova_table$Label==4
```

Cluster(2)

```
hclust.group.plot(Norm_data_mean[Virus_001_index],  
anova_table$Name[Virus_001_index], 6, "trend", 1, "Virus_001", "anova")
```



Cluster(3)



	A	B
1		group_data
2	WSSV003	1
3	WSSV004	2
4	WSSV006	1
5	WSSV009	1
6	WSSV014	3
7	WSSV015	2