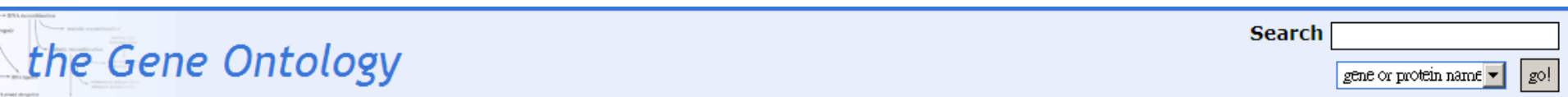


生物晶片與生物資料庫的連結 及基因網路的圖像化

GO(1)...



Gene Ontology Home

The Gene Ontology project provides a controlled vocabulary to describe gene and gene product attributes in any organism.

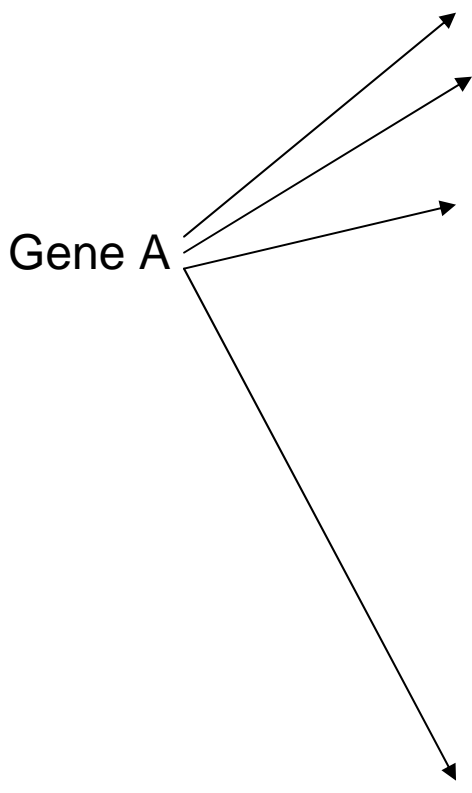
[Read more about the Gene Ontology...](#)

<http://www.geneontology.org>

GO database主要是紀錄gene product annotation的功能，是把annotation的記錄方式統一化，方便在查詢的時候，可以簡單將annotation做歸類，甚至是跨物種的查詢。收集這些資訊的方式則是從各物種的database上作統一歸類。

主要把annotation分成三大類，biological process，cellular component，molecular function，現在已經分出25XXX個GOID，大部分已知的gene(SWISS-PROT)在GO中都被貼上許多GOID來代表他們所具有的屬性，而這些GOID所代表的生物意義也是容易被理解的


GO(2)...



GO_ID	GO_Term	Type
GO:0000001	mitochondrion inheritance	P
GO:0000002	mitochondrial genome maintenance	P
GO:0000003	reproduction	P
GO:0000005	ribosomal chaperone activity	F
GO:0000006	high affinity zinc uptake transmembrane transporte...	F
GO:0000007	low-affinity zinc ion transmembrane transporter ac...	F
GO:0000008	thioredoxin	F
GO:0000009	alpha-1,6-mannosyltransferase activity	F
GO:0000010	trans-hexaprenyltranstransferase activity	F
GO:0000011	vacuole inheritance	P
GO:0000012	single strand break repair	P
GO:0000014	single-stranded DNA specific endodeoxyribonuclease...	F
GO:0000015	phosphopyruvate hydratase complex	C

GO(3)...GOMiner

<http://discover.nci.nih.gov/gominer/htgm.jsp>

 Application Build: 214 Database Build: 2008-04

[Home](#) [High-Throughput](#) [Getting Started](#) [Requirements](#) [Installation](#) [Downloads](#) [Command Line](#) [Database](#) [FAQ](#) [News](#) [Citing](#) [GoMiner in Pag](#)

Downloads

This page lists some of the most common file downloads for GoMiner.

[gominer.jar](#) The application file (without bundled database) for GUI GoMiner and accompanying command-line interface.


gominer.jar


gp2protein.u...


Homo_sapie... jo


netblast-2.2....


mLne


npp.2.0.1.01...

類型: Executable Jar File
修改日期: 2008/6/16 下午 05:56
大小: 17.7 MB

GoMiner Start Up Panel

Database

Organism(s)

Data Source(s)

Evidence Code(s)

all (default)

all (default)

all (default)

all (default)

Lookup Settings: ☒ Enhanced Names (UniProt Only) ☒ Synonym ☒ Cross Reference

View Settings: ☐ View All Genes and their Categories) ☐ Hide Genes ☐ View All GO Categories

Total:

Changed:

Browse

Browse

Auto-generate

Same as total

Process

Restore defaults

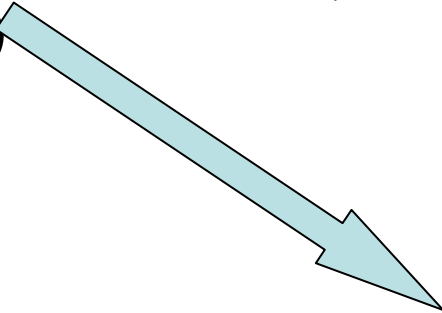
Exit

GO(4)…使用方法

DEGs
(gene list)



每個gene在GO database中被賦予的
annotation



DEGs的annotation在這實驗所代表的意義：

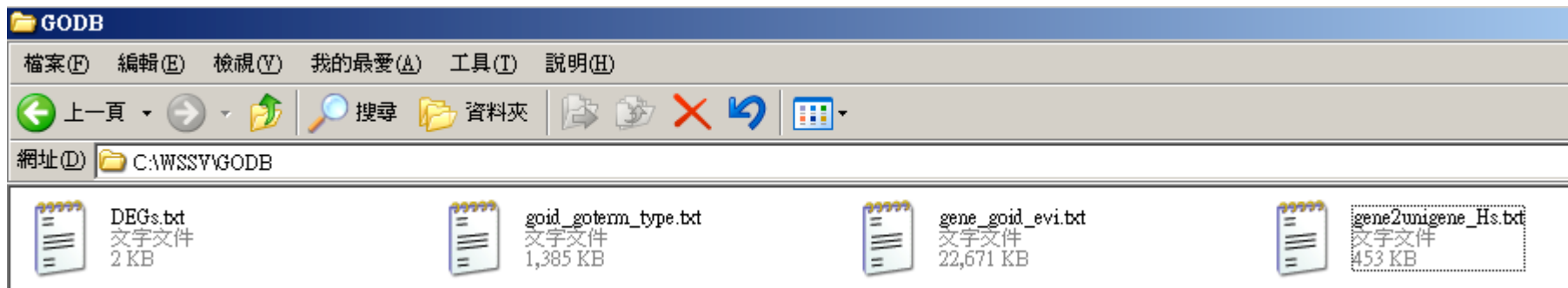
大多以Fisher 's exact test去計算每個annotation在DEGs所出現的機率是否是顯著的(每個annotation會給一個p_value)

所以是可以呈現DEGs所代表的function在細胞中被明顯改變的指標…

GO(5)···GO.anno.analysis

這邊以human的gene做一個範例，要如何用R去算DEGs的annotation的 p_value

首先需要以下檔案：



↓

	A
1	Hs.570688
2	Hs.493867
3	Hs.554050
4	Hs.368011
5	Hs.671209
6	Hs.512008
7	Hs.454921
8	Hs.606860

↓

	A	B	C
1	GO_ID	GO_Term	Type
2	GO:0000001	mitochondrion inheritance	P
3	GO:0000002	mitochondrial genome maintenance	P
4	GO:0000003	reproduction	P
5	GO:0000005	ribosomal chaperone activity	F
6	GO:0000006	high affinity zinc uptake transmembran	F

↓

	A	B	C
1	EntrezAC	GO_ID	Evidence
2	814629	GO:0003676	IEA
3	814629	GO:0005575	ND
4	814629	GO:0008150	ND
5	814629	GO:0008270	IEA
6	814630	GO:0003700	ISS

↓

	A	B
1	Entrez_Ser	UniGene
2	10	Hs.2
3	125	Hs.4
4	1084	Hs.11
5	1089	Hs.12
6	4049	Hs.36

Download from database

GO(6)···GO.anno.analysis

輸入以下程式：

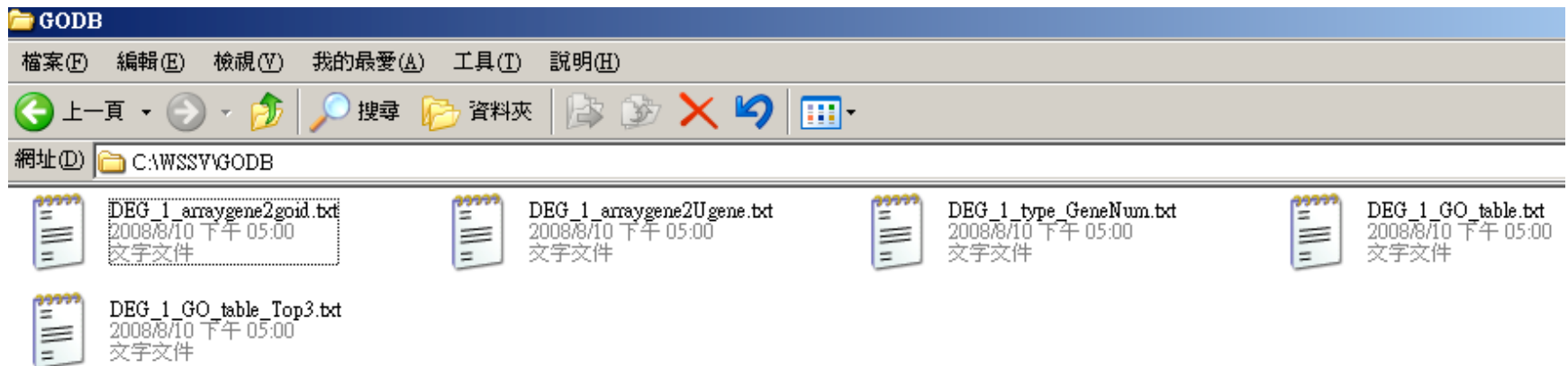
```
gene2unigene_Hs <- read.csv(paste(rawdir, "GODB", "gene2unigene_Hs.txt",  
sep="\\"), header=TRUE, sep = "\t")
```

```
gene_goid_evi <- read.csv(paste(rawdir, "GODB", "gene_goid_evi.txt",  
sep="\\"), header=TRUE, sep = "\t")
```

```
DEGs <- scan(paste(rawdir, "GODB", "DEGs.txt", sep="\\"), sep = "\t", what="")
```

```
GO.anno.analysis (DEGs, gene_goid_evi[,1:2], gene2unigene_Hs,  
"DEG_1","GODB")
```

Output files:



KEGG(1)...



Search KEGG

Get Entry

KEGG Home

Introduction

Overview

Release notes

Current statistics

KEGG Identifiers

KGML

KEGG API

KEGG FTP

KEGG: Kyoto Encyclopedia of Genes and Genomes

A grand challenge in the post-genomic era is a complete computer representation of the cell, the organism, and the biosphere, which will enable computational prediction of higher-level complexity of cellular processes and organism behaviors from genomic and molecular information. Towards this end we have been developing a bioinformatics resource named KEGG as part of the research projects of the Kanehisa Laboratories in the Bioinformatics Center of Kyoto University and the Human Genome Center of the University of Tokyo.

<http://www.genome.jp/kegg/>

KEGG則主要是紀錄gene product pathway，將pathways區分為許多大類，記錄每個pathway包含的genes，pathway figure，並給一個id，而不同的物種可能會有相似的pathway(有同一個pathway id)及ortholog gene。

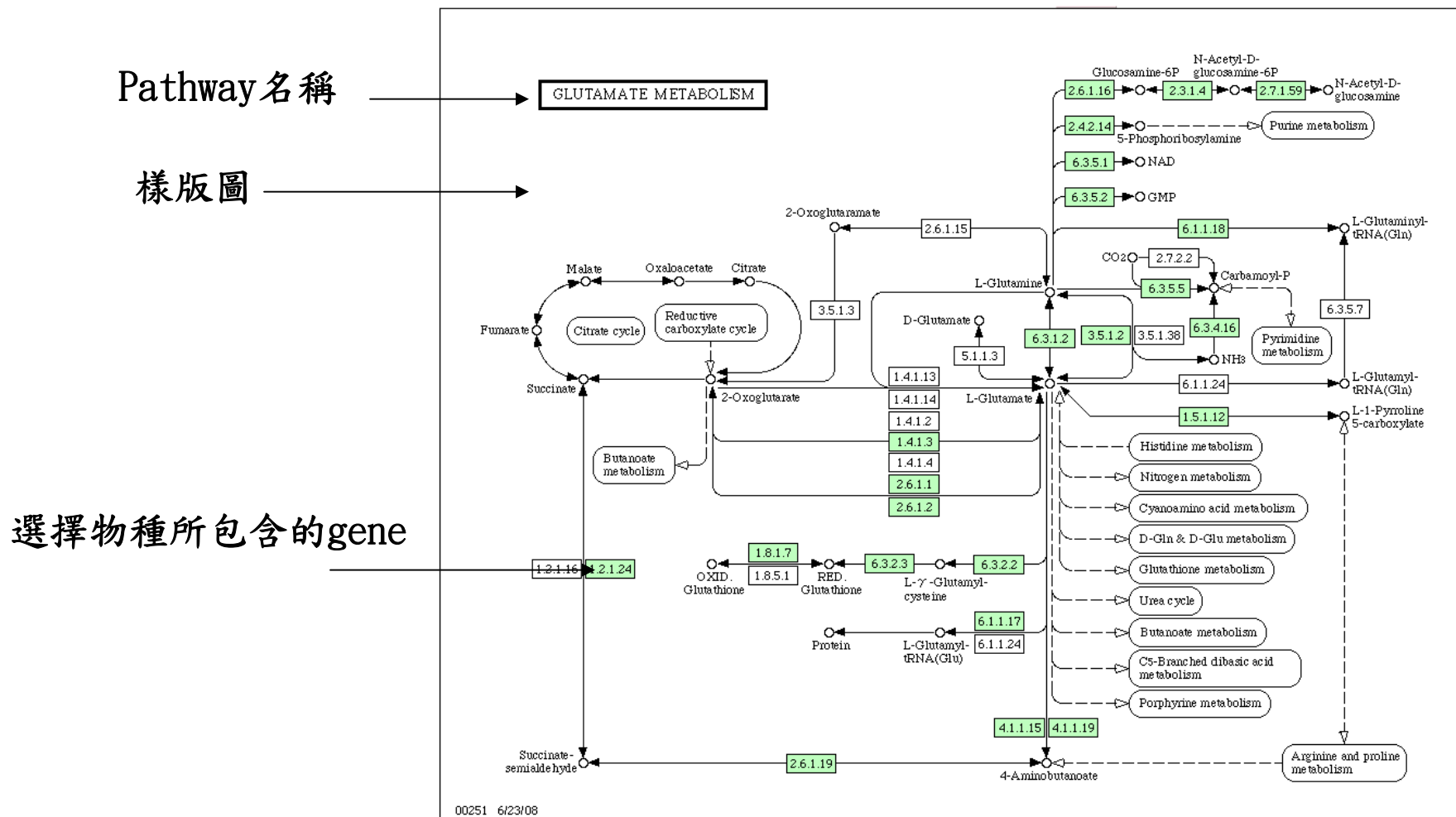
KEGG(2)...



KEGG Organisms in the Taxonomy

Eukaryotes: 72+27+49 Bacteria: 675 Archaea: 52
(including draft genomes and EST contigs)

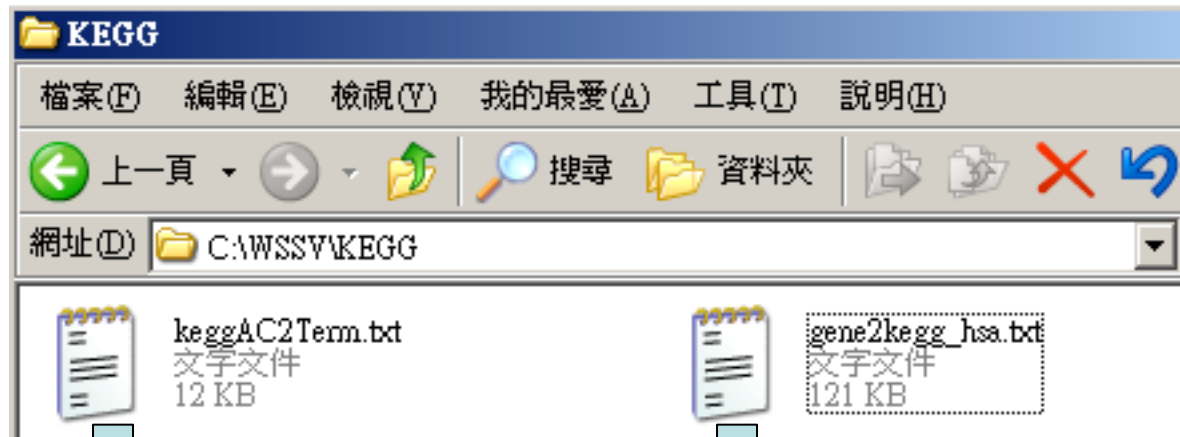
所記錄的物種數共有872種，*pathway*種類則有三百多個



KEGG(3)···使用方法

除了找出DEGs所在的pathways外，我們也可以用Fisher's exact test去統計DEGs所具有的pathways相對於所有gene來說是不是顯著的…

首先需要以下檔案：



	A	B
1	KEGGID	KEGGTerm
2	10	Glycolysis / Gluconeogenesis
3	20	Citrate cycle (TCA cycle)
4	30	Pentose phosphate pathway
5	31	Inositol metabolism
6	40	Pentose and glucuronate interconversions
7	51	Fructose and mannose metabolism
8	52	Galactose metabolism
9	53	Ascorbate and aldarate metabolism
10	61	Fatty acid biosynthesis

	A	B
1	EntrezAC	KEGGID
2	10	232
3	10	983
4	100	230
5	1000	4514
6	10000	4010
7	10000	4012
8	10000	4150
9	10000	4210

KEGG(4)...KEGG.anno.analysis

輸入以下程式：

```
gene2kegg_hsa <- read.csv(paste("C:/WSSV/KEGG", "gene2kegg_hsa.txt",  
sep="\\"), header=TRUE, sep = "\\t", colClasses="character")
```

```
KEGG.anno.analysis(DEGs, gene2kegg_hsa, gene2unigene_Hs,  
"DEG_1","KEGG")
```

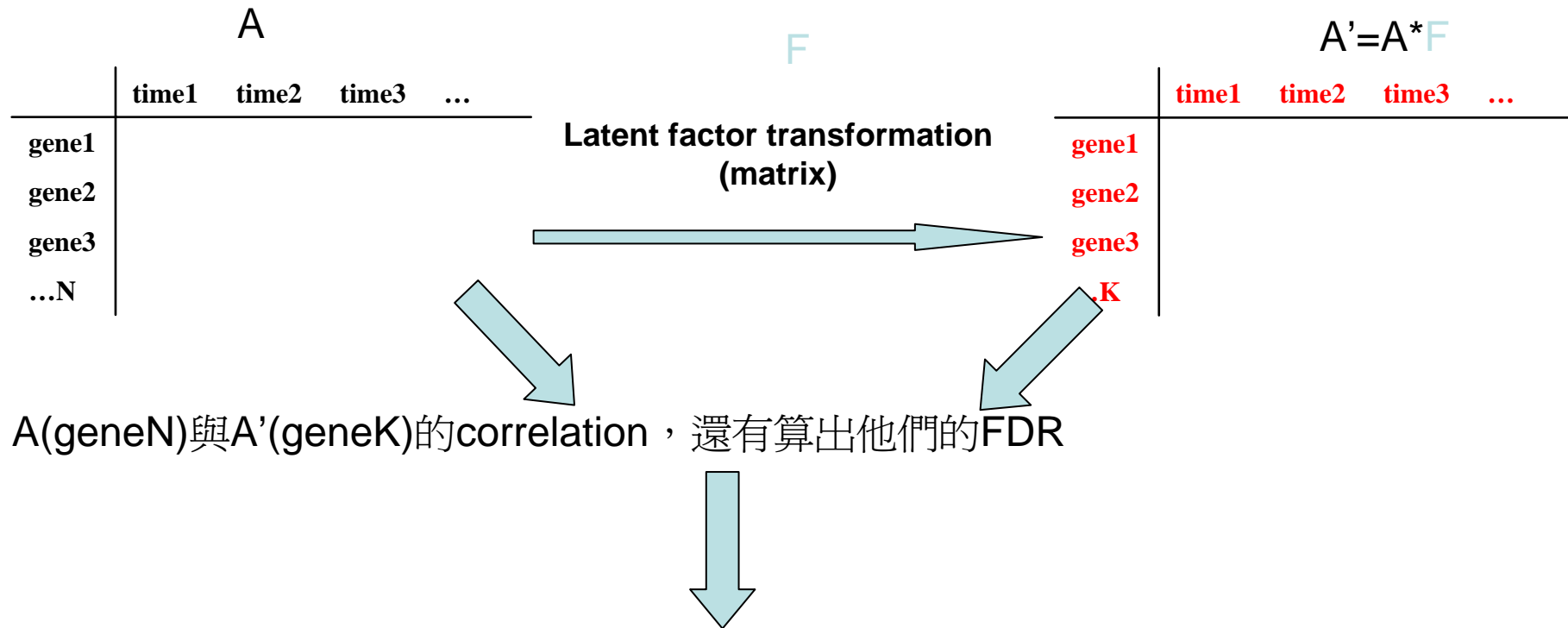
Output files:



基因網路

顧名思義，就是由生物體中的gene-gene interaction所交織而成的network，gene-gene interaction的確認，通常是經由單一基因的over express或repress去觀察其他gene的表現改變的觀察，而如果要從microarray data中，直接得到gene-gene interaction，通常是用一些演算法所得到

PLS Introduction

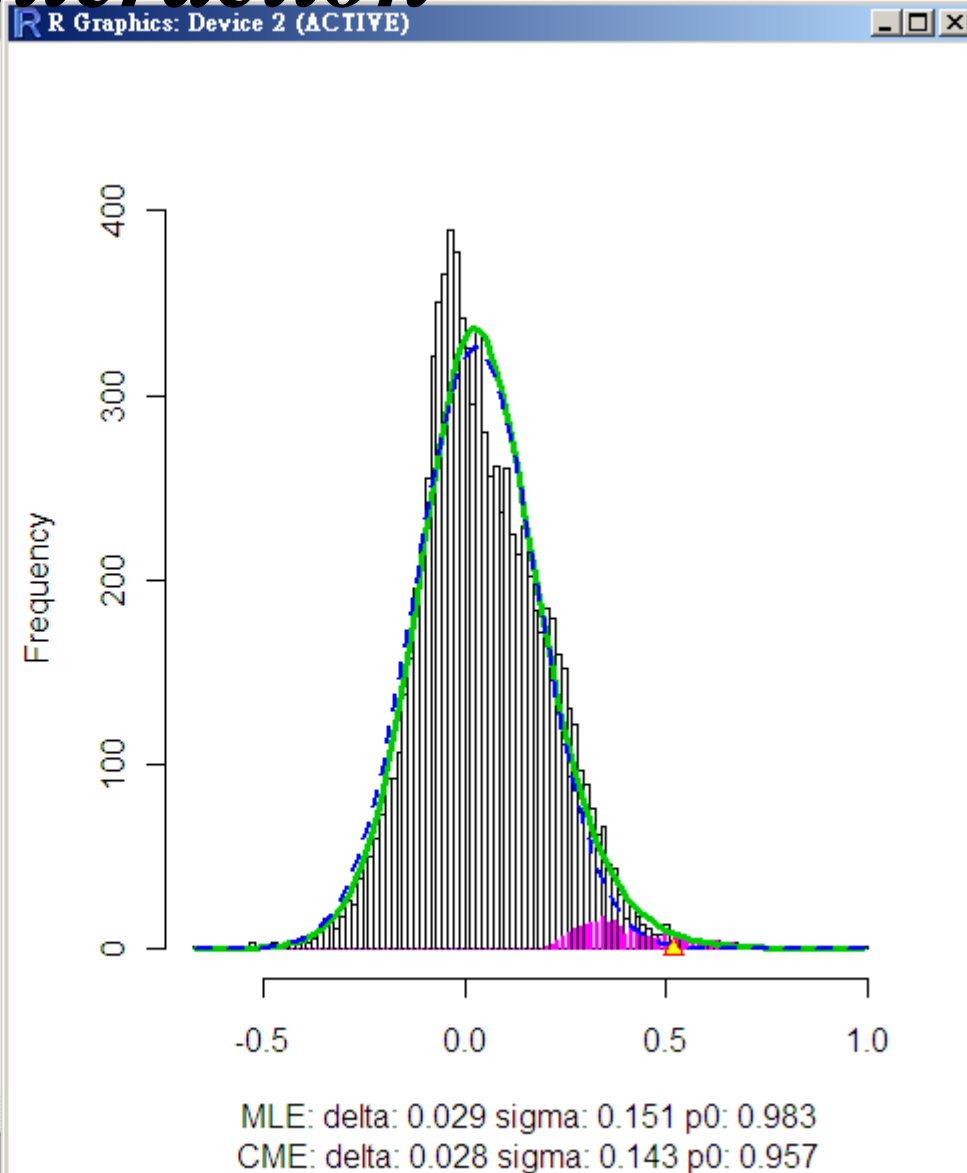


藉由correlation的rank跟FDR的cutline，就可以列出我們要的gene interaction list

PLS and gene-gene interaction

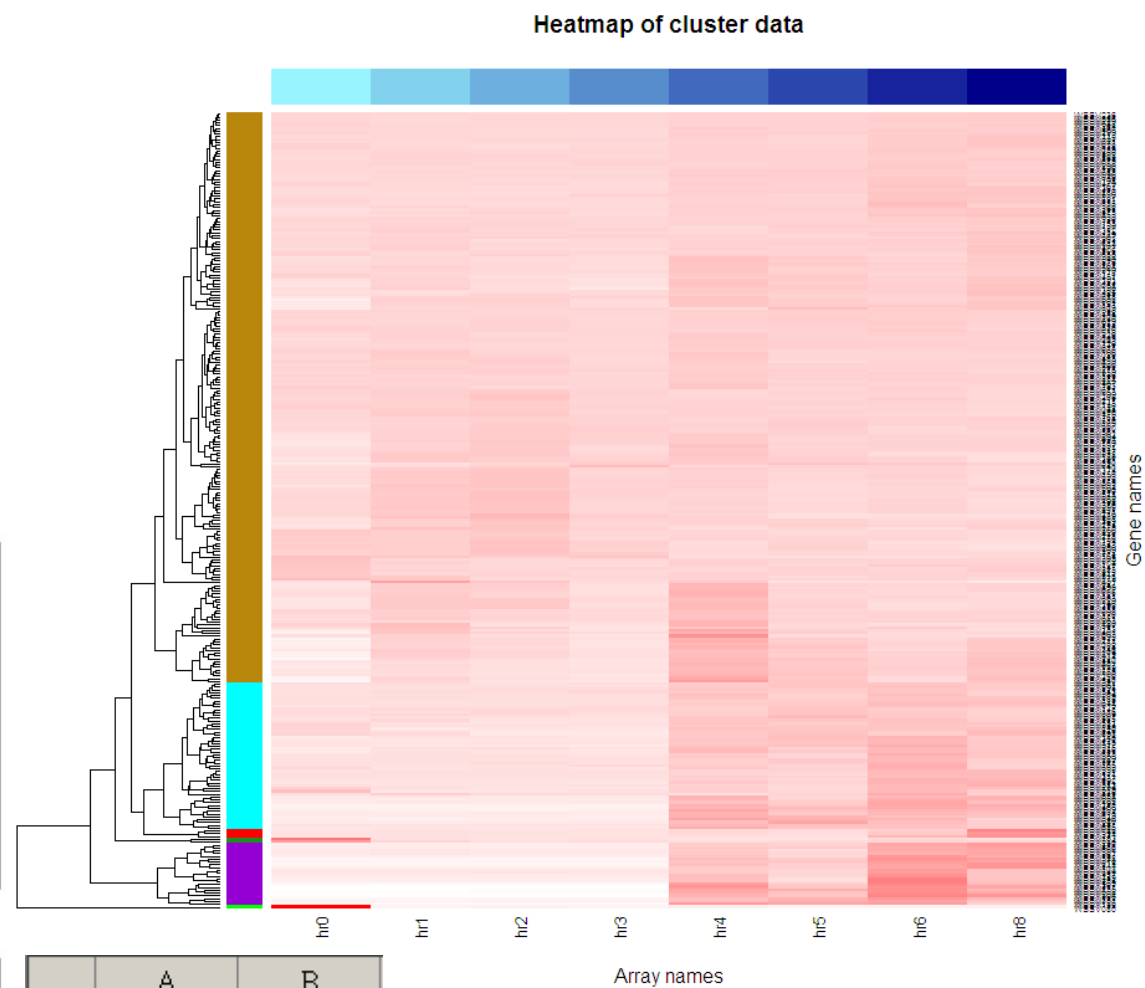
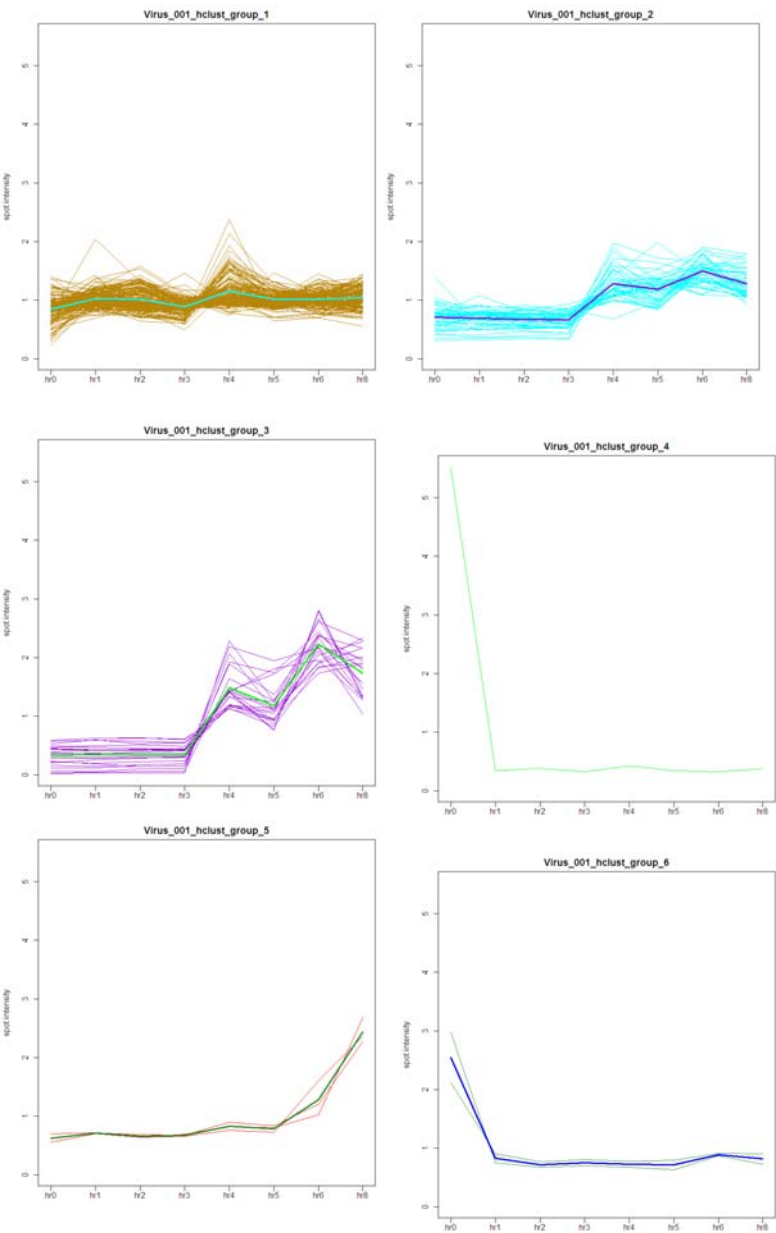
```
library(locfdr)  
library(splines)
```

```
Console  
Loading required package: splines  
[,1] [,2]  
[1,] "WSSV136" "WSSV140"  
[2,] "WSSV140" "WSSV143"  
[3,] "WSSV154" "WSSV156"  
[4,] "WSSV032" "WSSV179"  
[5,] "WSSV080" "WSSV179"  
[6,] "WSSV205" "WSSV217"  
[7,] "WSSV080" "WSSV218"  
[8,] "WSSV218" "WSSV220"  
[9,] "WSSV143" "WSSV230"  
[10,] "WSSV027" "WSSV234"  
[11,] "WSSV175" "WSSV274"  
[12,] "WSSV094" "WSSV294"  
[13,] "WSSV175" "WSSV359"  
[14,] "WSSV274" "WSSV359"  
[15,] "WSSV181" "WSSV461"  
[16,] "WSSV050" "WSSV469"  
[17,] "WSSV274" "WSSV469"  
[18,] "WSSV359" "WSSV469"  
[19,] "WSSV347" "WSSV521"  
[20,] "WSSV154" "WSSV524"  
[21,] "WSSV156" "WSSV524"  
[22,] "WSSV179" "WSSV524"  
[23,] "WSSV521" "WSSV524"  
[24,] "WSSV050" "WSSV543"  
[25,] "WSSV274" "WSSV543"  
[26,] "WSSV359" "WSSV543"  
[27,] "WSSV469" "WSSV543"  
[28,] "WSSV161" "WSSV613"  
[29,] "WSSV518" "WSSV652"
```



```
PLSnet <- function(data, genes, ncom=3, qvalue=.1, df.fit=5, ...)
```


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

PLSnet(1)

輸入以下程式：

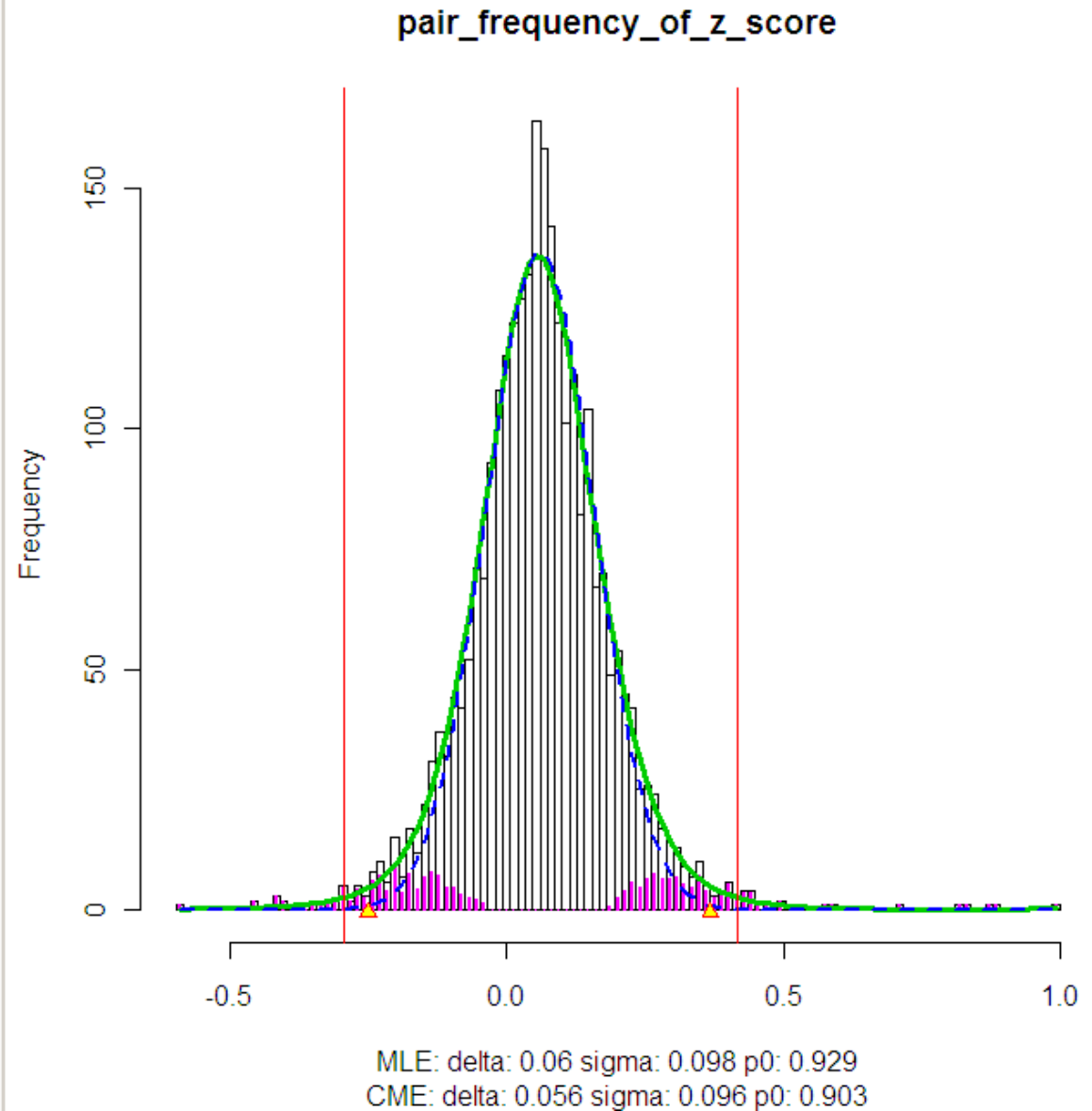
```
Virus_001_group_table <- read.csv(paste(rawdir, "anova",  
"Virus_001_group_table.txt", sep="\\"), header=TRUE, sep = "\\t")  
  
Norm_data_mean <- read.csv(paste(rawdir, "anova", "Norm_data_mean.txt",  
sep="\\"), header=TRUE, sep = "\\t")  
  
group2 <- Virus_001_group_table[Virus_001_group_table[,2]==2,1]  
  
group3 <- Virus_001_group_table[Virus_001_group_table[,2]==3,1]  
  
select_names <- c(as.character(group2),as.character(group3))  
  
select_column <- Norm_data_mean[,1] %in% select_names  
  
data <- Norm_data_mean[select_column,-1]  
  
names <- Norm_data_mean[select_column,1]  
  
net_pair <- PLSnet(data,names,qvalue=.1)
```


PLSnet(2)

net_pair

```
R Console  
[25,] "WSSV145" "WSSV626"  
[26,] "WSSV286" "WSSV626"  
[27,] "WSSV344" "WSSV626"  
[28,] "WSSV019" "WSSV652"  
[29,] "WSSV026" "WSSV652"  
[30,] "WSSV165" "WSSV652"  
[31,] "WSSV207" "WSSV652"  
[32,] "WSSV340" "WSSV652"  
[33,] "WSSV344" "WSSV652"  
[34,] "WSSV407" "WSSV652"  
[35,] "WSSV551" "WSSV652"  
[36,] "WSSV626" "WSSV652"  
[37,] "WSSV644" "WSSV652"  
[38,] "WSSV652" "WSSV658"  
[39,] "WSSV124" "WSSV663"  
[40,] "WSSV652" "WSSV663"  
[41,] "WSSV652" "WSSV684"  
[42,] "WSSV124" "WSSV698"  
[43,] "WSSV304" "WSSV698"  
[44,] "WSSV453" "WSSV698"  
[45,] "WSSV454" "WSSV698"  
[46,] "WSSV652" "WSSV698"  
[47,] "WSSV663" "WSSV698"  
> |
```

R Graphics: Device 2 (ACTIVE)



plot.network(1)

```
library("graph")  
library("RBGL")  
library("Rgraphviz")
```

輸入以下程式：

```
Node_file <- cbind(unique(c(net_pair[,1],net_pair[,2])),0)  
  
Node_file[Node_file[,1] %in% group2,2] <- 2  
Node_file[Node_file[,1] %in% group3,2] <- 3  
  
colnames(Node_file) <- c("gene_name","cluster")  
  
plot.network(as.data.frame(Node_file),net_pair,"WSSV_2_3_PLS","network")
```


plot.network(2)

