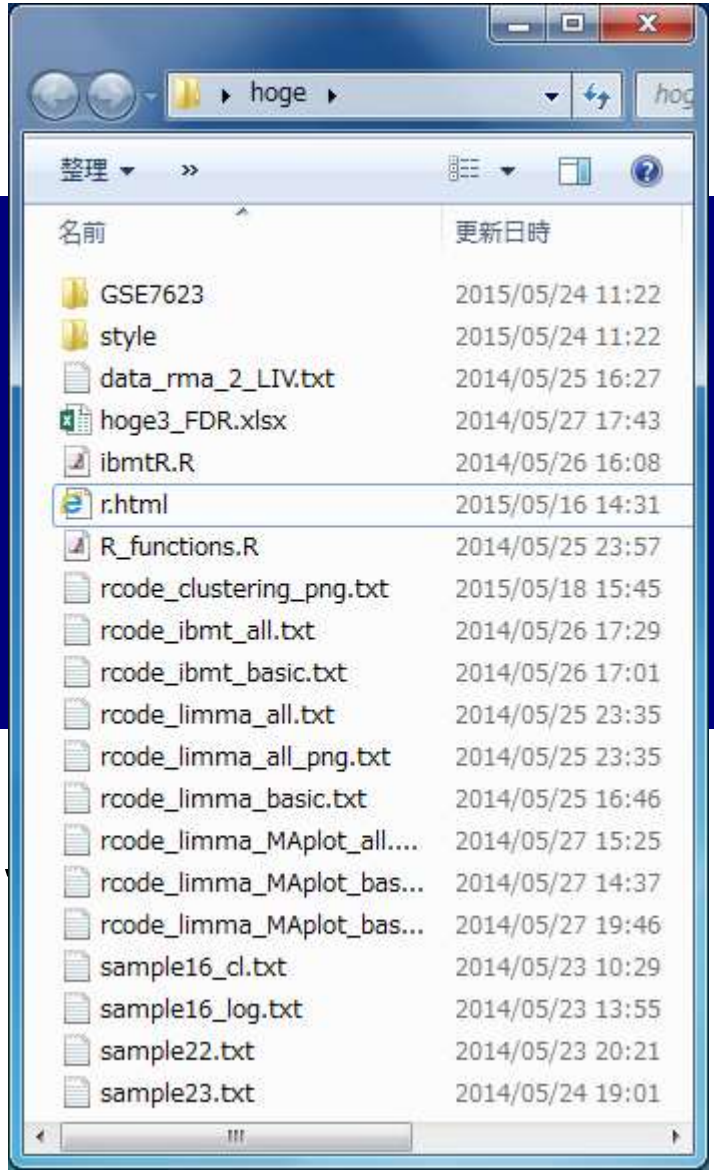
USBメモリ中のhogeフォルダをデスクトップにコピーしておいてください。



前回（5/19）のhogeフォルダがデスクトップに残っているかもしれないのでご注意ください。

機能ゲノム学 第3回

大学院農学生命科学研究科
アグリバイオインフォマティクス教育研究プログラ
門田幸二（かどた　こうじ）
kadota@iu.a.u-tokyo.ac.jp
http://www.iu.a.u-tokyo.ac.jp/~kadota/

講義予定

細胞中で発現している全転写物（トランスクリプトーム）の解析技術は、マイクロアレイから次世代シーケンサ（RNA-seq）に移行しつつあります。しかしRNA-seqデータ解析の多くは、マイクロアレイの知識を前提としています。本科目では、マイクロアレイデータを主な例として、各種トランスクリプトーム解析手法について解説します。

- 第1回（2015年5月12日）
  - 原理、各種データベース、生データ取得
  - 教科書の1.2節、2.2節周辺
- 第2回（2015年5月19日）
  - 遺伝子発現行列作成（データ正規化）
  - クラスタリング（データ変換や距離の定義など）
  - 教科書の3.2節周辺
- 第3回（2015年5月26日）
  - 実験デザイン、発現変動解析（多重比較問題）、M-A plot
  - 教科書の3.2節と4.2節周辺
- 第4回（2015年6月9日）
  - 機能解析（Gene Ontology解析やパスウェイ解析）、分類など

Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

教科書p107-108

「個体間のばらつき」を考慮した実験デザインを組むべし、という話

実験デザイン（§3.2.2）

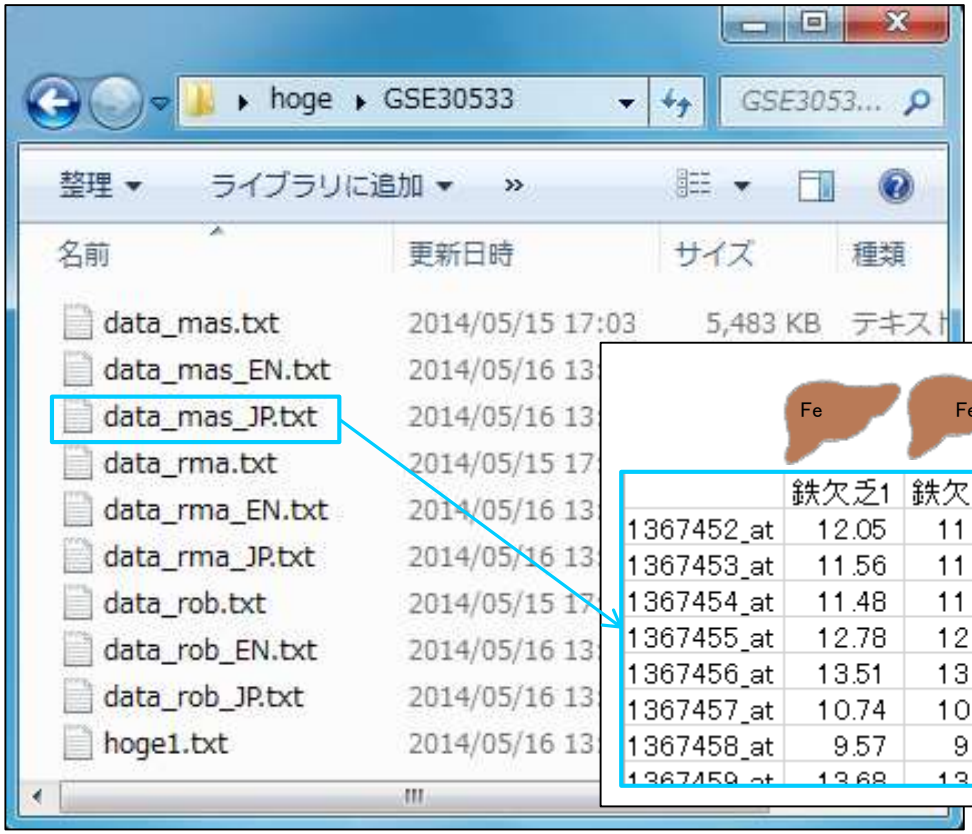
- Affymetrix GeneChip
  - Ge et al., *Genomics*, **86**: 127–141, 2005
    - **GSE2361**、GPL96 (Affymetrix Human Genome U133A Array)、22,283 probesets
    - **ヒト36サンプル**：Heart (心臓)、Thymus (胸腺)、Spleen (脾臓)、Ovary (卵巣)、Kidney (腎臓)、Skeletal Muscle (骨格筋)、Pancreas (膵臓)、Prostate (前立腺)、…
  - Nakai et al., *Biosci Biotechnol Biochem.*, **72**: 139–148, 2008　8匹のラットを使用
    - **GSE7623**、GPL1355 (Affymetrix Rat Genome 230 2.0 Array)、31,099 probesets
    - **ラット24サンプル**：Brown adipose tissue (褐色脂肪組織; BAT)8サンプル、White adipose tissue (白色脂肪組織; WAT)8サンプル、Liver (肝臓; LIV)8サンプル
      - BAT 8サンプル：通常（BAT_fed） 4サンプル　対　24時間絶食（BAT_fas） 4サンプル
      - WAT 8サンプル：通常（WAT_fed） 4サンプル　対　24時間絶食（WAT_fas） 4サンプル
      - LIV 8サンプル：通常（LIV_fed） 4サンプル　対　24時間絶食（LIV_fas） 4サンプル
  - Kamei et al., *PLoS One*, **8**: e65732, 2013　10匹のラットを使用
    - **GSE30533**、GPL1355 (Affymetrix Rat Genome 230 2.0 Array)、31,099 probesets
    - **ラット10サンプル**：全てLiver (肝臓)サンプル
    - iron-deficient diet (Iron_def) 5サンプル　対　control diet (Control) 5サンプル

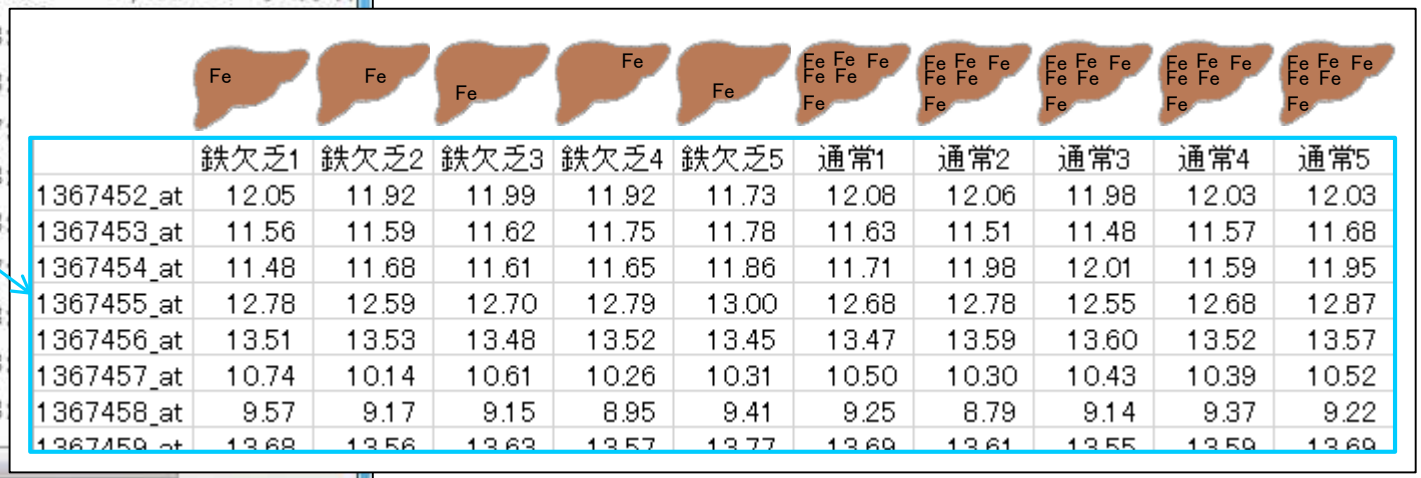
教科書p107-108

実験デザイン（§3.2.2）

2群間比較が主な目的であり、各群につき5反復(five replicates)とっている。生物学的なばらつき(biological variation)を考慮すべく、反復データは別々の個体からとっている(biological replicates)

- Kamei et al., PLoS One, 8: e65732, 2013
  - GSE30533、GPL1355 (Affymetrix Rat Genome 230 2.0 Array)、31,099 probesets
  - ラット10サンプル：全てLiver（肝臓）サンプル
  - iron-deficient diet（Iron_def）5サンプル　対　control diet（Control）5サンプル





	鉄欠乏1	鉄欠乏2	鉄欠乏3	鉄欠乏4	鉄欠乏5	通常1	通常2	通常3	通常4	通常5
1367452_at	12.05	11.92	11.99	11.92	11.73	12.08	12.06	11.98	12.03	12.03
1367453_at	11.56	11.59	11.62	11.75	11.78	11.63	11.51	11.48	11.57	11.68
1367454_at	11.48	11.68	11.61	11.65	11.86	11.71	11.98	12.01	11.59	11.95
1367455_at	12.78	12.59	12.70	12.79	13.00	12.68	12.78	12.55	12.68	12.87
1367456_at	13.51	13.53	13.48	13.52	13.45	13.47	13.59	13.60	13.52	13.57
1367457_at	10.74	10.14	10.61	10.26	10.31	10.50	10.30	10.43	10.39	10.52
1367458_at	9.57	9.17	9.15	8.95	9.41	9.25	8.79	9.14	9.37	9.22
1367459_at	13.68	13.56	13.63	13.57	13.77	13.69	13.61	13.55	13.59	13.69

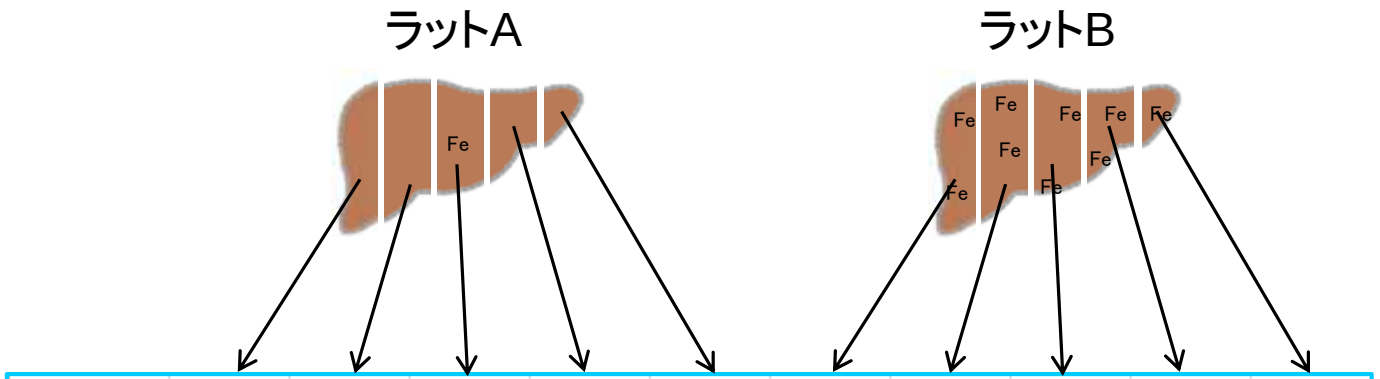
教科書p107-108

実験デザイン（§3.2.2）

このやり方で得られる結論は限定的!できるだけ多様な別個体サンプルを沢山用いるべし!

- Kamei et al., PLoS One. 8: e65732. 2013
  - GSE30533、 31,099 probesets
  - ラット10サン
  - iron-deficient diet (iron_def) 5サンプル 対 control diet (Control) 5サンプル

対比的な用語は技術的なばらつき（technical variation）であり、同一個体由来サンプルを分割して得られた反復データ（technical replicates）



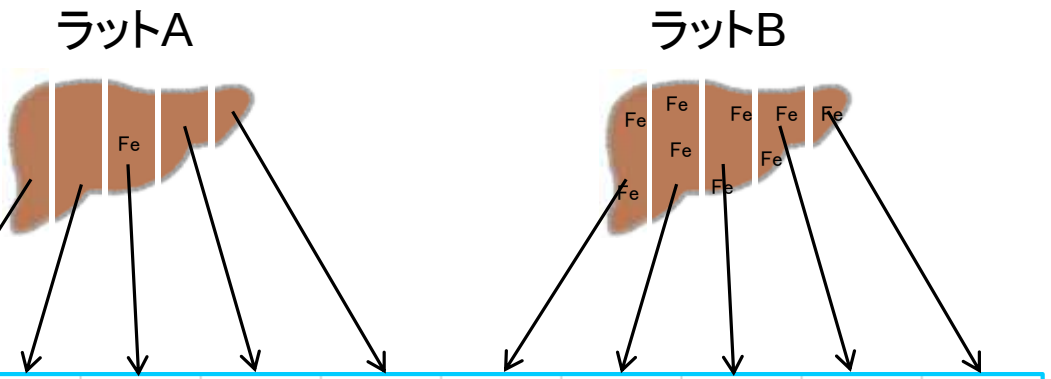
	鉄欠乏1	鉄欠乏2	鉄欠乏3	鉄欠乏4	鉄欠乏5	通常1	通常2	通常3	通常4	通常5
1367452_at	12.05	11.92	11.99	11.92	11.73	12.08	12.06	11.98	12.03	12.03
1367453_at	11.56	11.59	11.62	11.75	11.78	11.63	11.51	11.48	11.57	11.68
1367454_at	11.48	11.68	11.61	11.65	11.86	11.71	11.98	12.01	11.59	11.95
1367455_at	12.78	12.59	12.70	12.79	13.00	12.68	12.78	12.55	12.68	12.87
1367456_at	13.51	13.53	13.48	13.52	13.45	13.47	13.59	13.60	13.52	13.57
1367457_at	10.74	10.14	10.61	10.26	10.31	10.50	10.30	10.43	10.39	10.52
1367458_at	9.57	9.17	9.15	8.95	9.41	9.25	8.79	9.14	9.37	9.22
1367459_at	13.68	13.56	13.63	13.57	13.77	13.69	13.61	13.55	13.59	13.69

実験デザイン（§3.2.2）

2群間での発現変動遺伝子(DEG)検出結果は多くなる傾向。**多ければいいというものではない!**

- Technical replicatesだと…
  1. 自分は「鉄欠乏　対　通常」の違いを見ているつもりでも、個体間の他の違い（身長、体重など）由来要因との区別がつかない

     高身長　対　低身長、低体重　対　高体重、他の病気の有無、家系の違いなど
  2. 得られる結果から導き出される結論は、そのラット間のみで成立する事象であり、ラットという生物種全体に適用可能なわけではない



	鉄欠乏1	鉄欠乏2	鉄欠乏3	鉄欠乏4	鉄欠乏5	通常1	通常2	通常3	通常4	通常5
1367452_at	12.05	11.92	11.99	11.92	11.73	12.08	12.06	11.98	12.03	12.03
1367453_at	11.56	11.59	11.62	11.75	11.78	11.63	11.51	11.48	11.57	11.68
1367454_at	11.48	11.68	11.61	11.65	11.86	11.71	11.98	12.01	11.59	11.95
1367455_at	12.78	12.59	12.70	12.79	13.00	12.68	12.78	12.55	12.68	12.87
1367456_at	13.51	13.53	13.48	13.52	13.45	13.47	13.59	13.60	13.52	13.57
1367457_at	10.74	10.14	10.61	10.26	10.31	10.50	10.30	10.43	10.39	10.52
1367458_at	9.57	9.17	9.15	8.95	9.41	9.25	8.79	9.14	9.37	9.22
1367459_at	13.68	13.56	13.63	13.57	13.77	13.69	13.61	13.55	13.59	13.69

教科書p107-108

実験デザイン（§3.2.2）

普遍的な結果を得たいのなら、できるだけ多様な別個体サンプルを沢山用いるべし!
Expression Atlasも**3 biological replicates**以上を基本としているようだ。

- Biological replicatesでも多様性が不十分な場合はイマイチ…

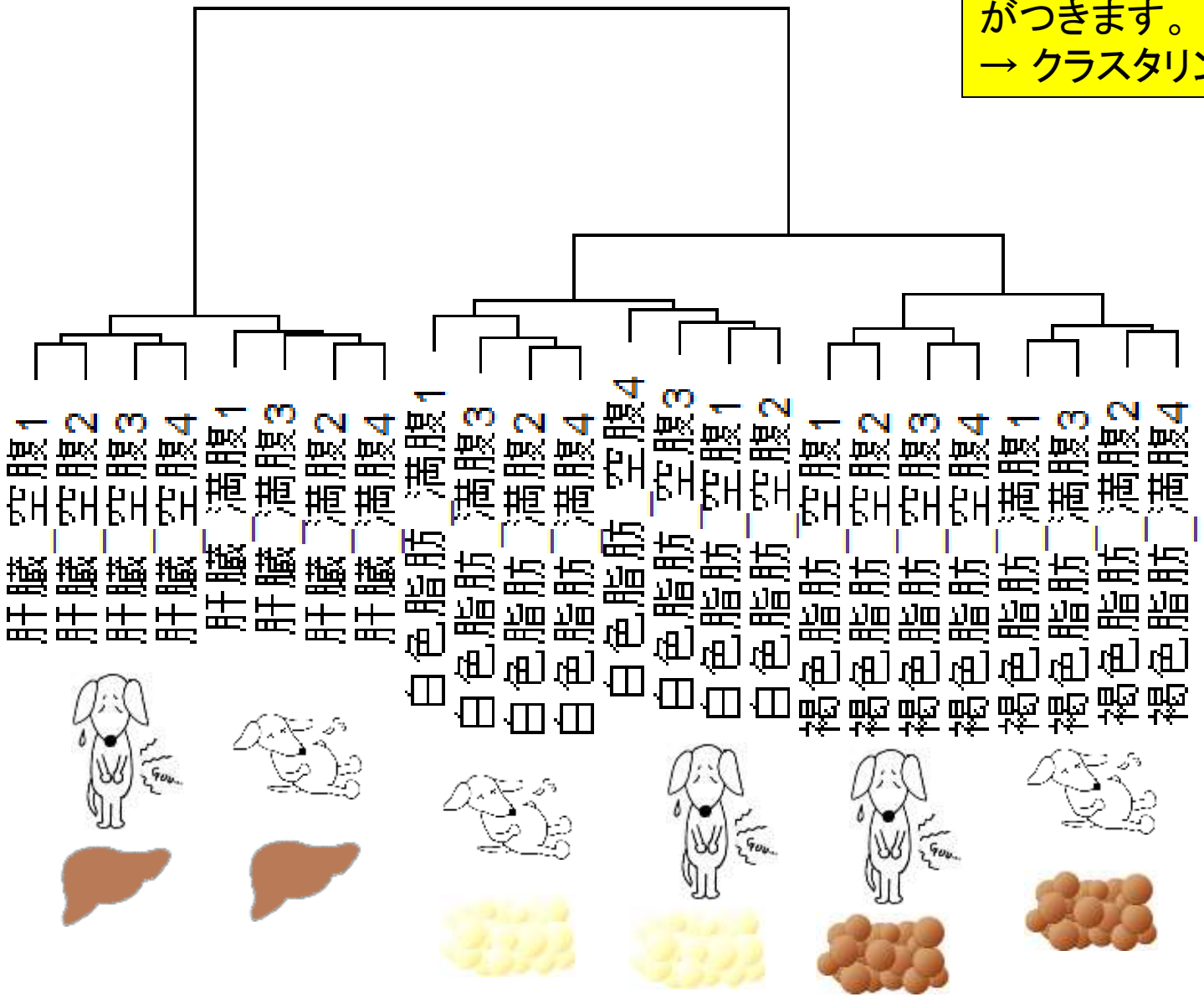
家系A　　家系B

	鉄欠乏1	鉄欠乏2	鉄欠乏3	鉄欠乏4	鉄欠乏5	通常1	通常2	通常3	通常4	通常5
1367452_at	12.05	11.92	11.99	11.92	11.73	12.08	12.06	11.98	12.03	12.03
1367453_at	11								11.57	11.68
1367454_at	11								11.59	11.95
1367455_at	12								12.68	12.87
1367456_at	13								13.52	13.57
1367457_at	10								10.39	10.52
1367458_at	9.57	9.17	9.15	8.95	9.41	9.25	8.79	9.14	9.37	9.22
1367459_at	13.68	13.56	13.63	13.57	13.77	13.69	13.61	13.55	13.59	13.69

Technical repliatesよりはましだろうが、鉄欠乏状態による結果なのか家系の違いによるものなのかの区別はつけられない。

クラスタリングと発現変動解析

クラスタリング結果を眺めれば、発現変動遺伝子（DEG）数に関するおおよその見当がつきます。
→ クラスタリングって重要



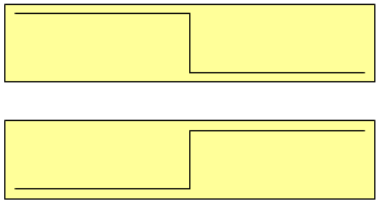
Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

対数変換後のデータを用いて2群間比較

データ解析もいろいろ

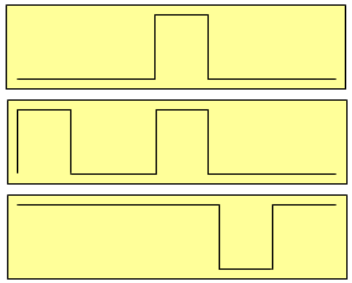
発現変動遺伝子同定



遺伝子発現行列

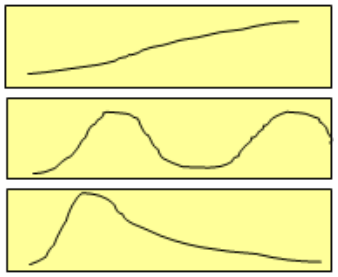
二群間比較用

	A群 A1	A2	…	B群 B1	B2	…
gene 1	$x^A_{1,1}$	$x^A_{1,2}$		$x^B_{1,2}$	$x^B_{1,2}$	
gene 2	$x^A_{2,1}$	$x^A_{2,2}$		$x^B_{2,2}$	$x^B_{2,2}$	
…	…	…	…	…	…	…
gene i	$x^A_{i,1}$	$x^A_{i,2}$		$x^B_{i,2}$	$x^B_{i,2}$	
…	…	…	…	…	…	…
gene n	$x^A_{n,1}$	$x^A_{n,2}$		$x^B_{n,2}$	$x^B_{n,2}$	



様々な組織（条件）

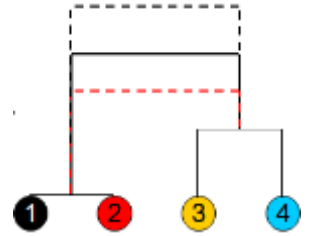
	S1	S2	S3	S4	…
gene 1	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	…
gene 2	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	…
…	…	…	…	…	…
gene i	$x_{i,1}$	$x_{i,2}$	$x_{i,3}$	$x_{i,4}$	
…	…	…	…	…	…
gene n	$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$x_{n,4}$	…



時系列データ

	T1	T2	T3	T4	…
gene 1	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	…
gene 2	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	…
…	…	…	…	…	…
gene i	$x_{i,1}$	$x_{i,2}$	$x_{i,3}$	$x_{i,4}$	
…	…	…	…	…	…
gene n	$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$x_{n,4}$	…

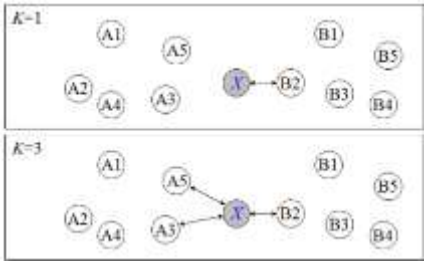
クラスタリング



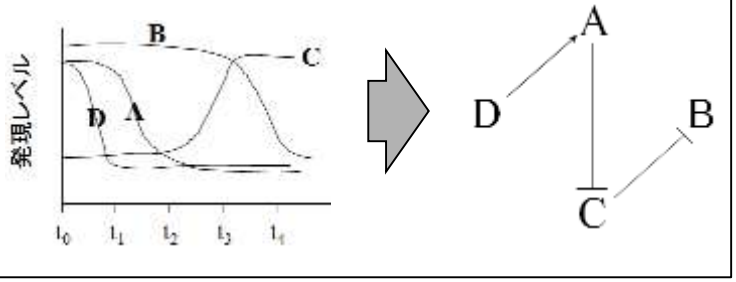
機能解析

- Gene Ontology（GO）
- パスウェイ解析

分類（診断）



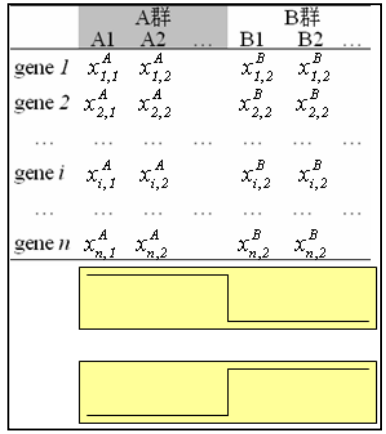
遺伝子ネットワーク推定

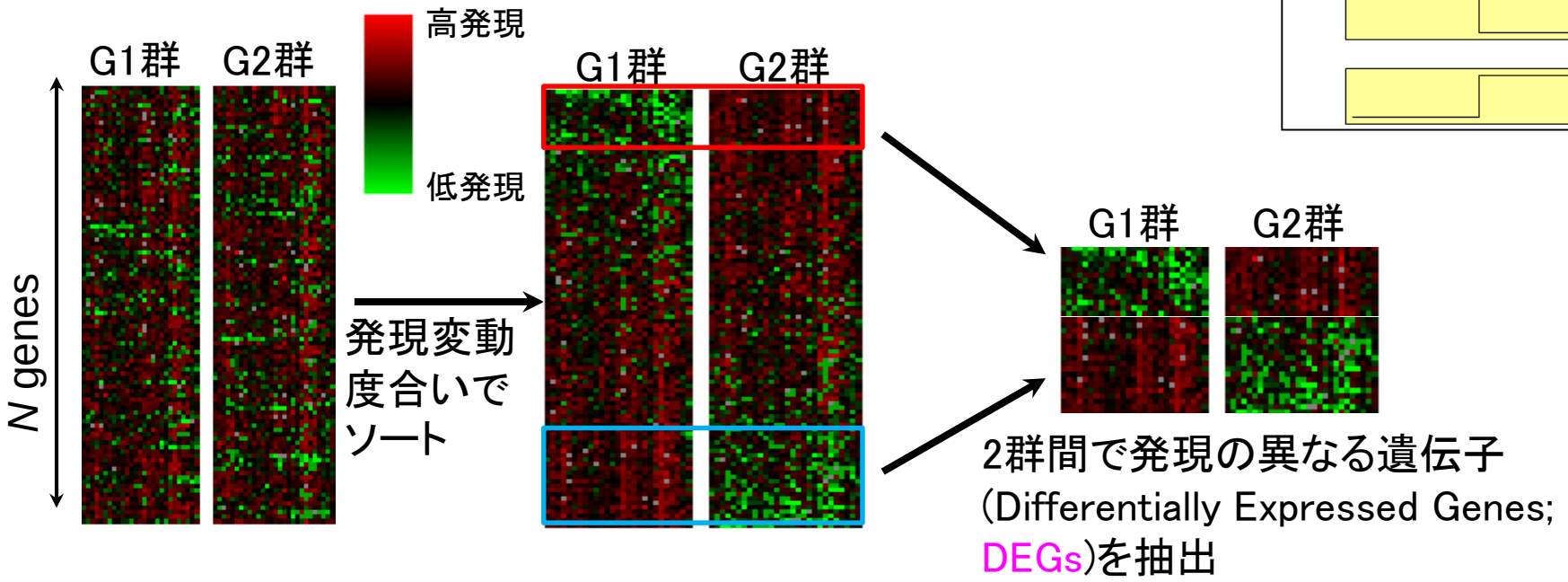


2群間比較

比較したいグループ間で発現変動している遺伝子または転写物を同定することはデータ解析の基本

- 農産物の栽培条件の違い（通常 対 低温、通常 対 乾燥）
- 味の違い（おいしい 対 まずい）
- サンプルの状態の違い（癌 対 正常）



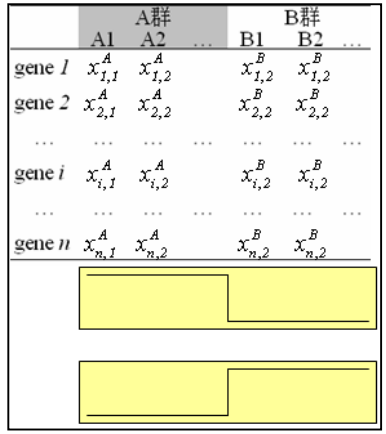


2群間比較

相関係数rの絶対値が大きいほど
発現変動の度合いが大きいと解釈

- パターンマッチング法
  - 理想的なパターンyとの類似度が高い順にランキング

$$\text{相関係数 } r = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i-\bar{y})^2}} \qquad (-1 \le r \le 1)$$



y	1	1	1	1	1	1	0	0	0	0	0

rownan	G1_1	G1_2	G1_3	G1_4	G1_5	G1_6	G2_1	G2_2	G2_3	G2_4	G2_5	r
gene1	6.44	6.30	6.51	6.36	6.49	6.39	3.58	4.39	4.25	3.70	4.09	0.98
gene2	5.81	6.93	6.73	5.55	6.39	6.61	2.81	5.46	1.00	3.46	4.17	0.81
gene3	3.91	4.81	5.04	3.17	4.75	5.36	5.58	5.52	5.70	5.64	5.61	−0.71

パターンマッチング法

テンプレートパターン情報を含むファイルを読み込んでパターンマッチングを行ってみよう

(Rで)マイクロアレイデータ解析

(last modified 2015/05/16, since 2005)

What's new?

- 門田幸二 イ解析に関 どについて このページ
- お知らせは イ関連の講 ら辿れます
- はじめに(1

- 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test(last modified 2014/05/25)
- 解析 | 発現変動 | 2群間 | 対応なし | Welch t-test(last modified 2014/05/23)
- 解析 | 発現変動 | 2群間 | 対応なし | Mann-Whitney U-test(last modified 2013/10/15)
- 解析 | 発現変動 | 2群間 | 対応なし | パターンマッチング法(last modified 2014/05/23)
- 解析 | 発現変動 | 2群間 | 対応あり | について(last modified 2009/11/11)
- 解析 | 発現変動 | 2群間 | 対応あり | SAM (Tusher, 2001) (last modified 2014/06/02)

	A群			B群		
	A1	A2	…	B1	B2	…
gene 1	$x^A_{1,1}$	$x^A_{1,2}$		$x^B_{1,2}$	$x^B_{1,2}$	
gene 2	$x^A_{2,1}$	$x^A_{2,2}$		$x^B_{2,2}$	$x^B_{2,2}$	
…	…	…	…	…	…	…
gene i	$x^A_{i,1}$	$x^A_{i,2}$		$x^B_{i,2}$	$x^B_{i,2}$	
…	…	…	…	…	…	…
gene n	$x^A_{n,1}$	$x^A_{n,2}$		$x^B_{n,2}$	$x^B_{n,2}$	

解析 | 発現変動 | 2群間 | 対応なし | パターンマッチング法 NEW

パターンマッチング法を用いて、2群間での発現変動遺伝子の同定を行うやり方を紹介します。
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し、以下をコピペ

1. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用するやり方です。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param <- "pearson"                       #相関係数の種類を指定("pearson"または"spearman")

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="")#in_f1で指定したファイルの読
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]                      #テンプレートパターンベクトルdata.clを作成

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子についてテンプレートパターンdata.clと

#ファイルに保存
tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係数rのベクトルを結合した結果をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保
```

パターンマッチング法

入出力の全体像。出力ファイルは、入力ファイルin_f1の右側に相関係数計算結果が追加されたもの。

解析 | 発現変動 | 2群間 | 対応なし | パターンマッチング法 NEW

パターンマッチング法を用いて、2群間での発現変動遺伝子の同定を行うやり方を紹介します。
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し、以下をコピペ

1. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用するやり方です。

```
in_f1 <- "sample16_log.txt"            #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"             #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                   #出力ファイル名を指定してout_fに格納
param <- "pearson"                     #相関係数の種類を指定("pearson"または"spearman")

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="")#in_f1で指定したファイルの読
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]                    #テンプレートパターンベクトルdata.clを作成

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子についてテンプレートパターンdata.clと

#ファイルに保存
tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係数rのベクトルを結合した結果をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保
```

G1_1	1
G1_2	1
G1_3	1
G1_4	1
G1_5	1
G1_6	1
G2_1	0
G2_2	0
G2_3	0
G2_4	0
G2_5	0

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	rownan	G1_1	G1_2	G1_3	G1_4	G1_5	G1_6	G2_1	G2_2	G2_3	G2_4	G2_5	r
2	gene1	6.44	6.30	6.51	6.36	6.49	6.39	3.58	4.39	4.25	3.70	4.09	0.984
3	gene2	5.81	6.93	6.73	5.55	6.39	6.61	2.81	5.46	1.00	3.46	4.17	0.811
4	gene3	3.91	4.81	5.04	3.17	4.75	5.36	5.58	5.52	5.70	5.64	5.61	-0.708

入力ファイルの読み込みがうまくできていることがわかる

コードの解説

1. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用するやり方です。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param <- "pearson"                       #相関係数の種類を指定("pearson"または"spearman")

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=
hoge <- read.table(in_f2, sep="\t
data.cl <- hoge[,2]

#本番
r <- apply(data, 1, cor, y=data.c

#ファイルに保存
tmp <- cbind(rownames(data), data
write.table(tmp, out_f, sep="\t",
```

R Console

```
> getwd()
[1] "C:/Users/kadota/Desktop/hoge"
> list.files(pattern="16")
[1] "sample16_cl.txt"  "sample16_log.txt"
> in_f1 <- "sample16_log.txt"              #入力ファイル名を指$
> in_f2 <- "sample16_cl.txt"               #入力ファイル名を指$
> out_f <- "hoge1.txt"                     #出力ファイル名を指$
> param <- "pearson"                       #相関係数の種類を指$
>
> #入力ファイルの読み込みとラベル情報の作成
> data <- read.table(in_f1, header=TRUE, row.names=1, sep="\$
> data
      G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5
gene1 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09
gene2 5.81 6.93 6.73 5.55 6.39 6.61 2.81 5.46 1.00 3.46 4.17
gene3 3.91 4.81 5.04 3.17 4.75 5.36 5.58 5.52 5.70 5.64 5.61
>
```

コードの解説

読み込み時にheader=TRUEや
row.names=1の記述がない点に注目!

1. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用するやり方です。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param <- "pearson"                       #相関係数の種類を指定("pearson"または"spearman")

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="")#in_f1で指定したファイルの読み込み
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]                      #テンプレートパターンベクトルdata.clを作成

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子についてテンプレートパターンdata.clとの相関

#ファイルに保存
tmp <- cbind(rownames(data), data
write.table(tmp, out_f, sep="\t",
```

G1_1	1
G1_2	1
G1_3	1
G1_4	1
G1_5	1
G1_6	1
G2_1	0
G2_2	0
G2_3	0
G2_4	0
G2_5	0

R Console

```
> data
      G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5
gene1 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09
gene2 5.81 6.93 6.73 5.55 6.39 6.61 2.81 5.46 1.00 3.46 4.17
gene3 3.91 4.81 5.04 3.17 4.75 5.36 5.58 5.52 5.70 5.64 5.61
> hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定$
> data.cl <- hoge[,2]                          #テンプレートパタ$
> data.cl
 [1] 1 1 1 1 1 1 0 0 0 0 0
>
```

コードの解説

hogeの中身は入力ファイルと同じだが、
欲しいのはhogeオブジェクトの2列目部分

1. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用するやり方です。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_f1に格納
param <- "pearson"                       #相関係数の種類を指定("p

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t",
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したフ
data.cl <- hoge[,2]                      #テンプレートパターンベク

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子

#ファイルに保存
tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names
```

R Console

```
> hoge
      V1 V2
1   G1_1  1
2   G1_2  1
3   G1_3  1
4   G1_4  1
5   G1_5  1
6   G1_6  1
7   G2_1  0
8   G2_2  0
9   G2_3  0
10  G2_4  0
11  G2_5  0
> dim(hoge)
[1] 11  2
> hoge[3, ]
    V1 V2
3 G1_3  1
> hoge[, 2]
 [1] 1 1 1 1 1 1 0 0 0 0 0
>
```

G1_1	1
G1_2	1
G1_3	1
G1_4	1
G1_5	1
G1_6	1
G2_1	0
G2_2	0
G2_3	0
G2_4	0
G2_5	0

読み込み時にrow.names=1をつけてもよい

Tips

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に格納(発現データ)
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に格納(テンプレート情報)
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param <- "pearson"                       #相関係数の種類を指定("pearson"または"spearman")

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="")#in_f1で指定したファイルの読み込み
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]

#本番
r <- apply(data, 1, cor,

#ファイルに保存
tmp <- cbind(rownames(dat
write.table(tmp, out_f, s
```

R Console

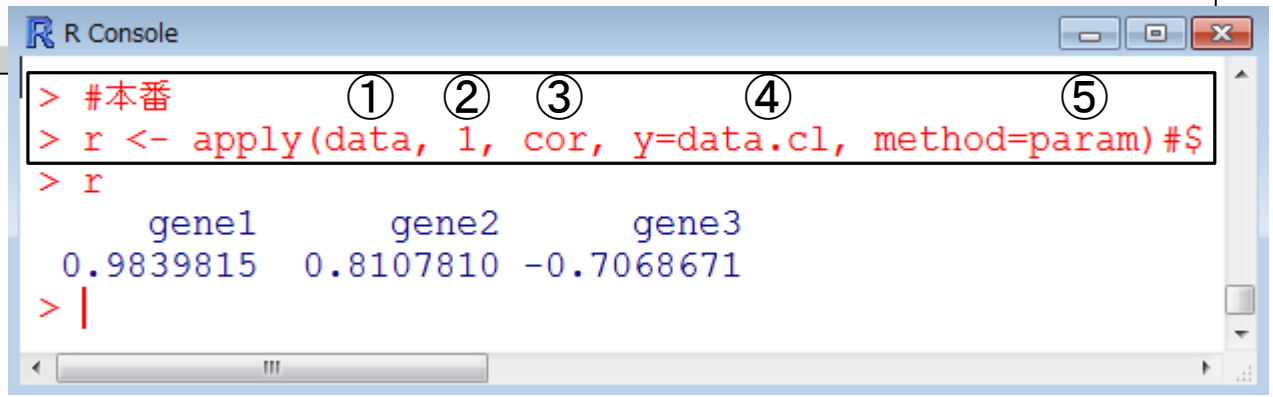
```
> hoge <- read.table(in_f2, row.names=1, sep="\t", quote="")
> hoge
      V2
G1_1   1
G1_2   1
G1_3   1
G1_4   1
G1_5   1
G1_6   1
G2_1   0
G2_2   0
G2_3   0
G2_4   0
G2_5   0
> data.cl <- hoge[, 1]
> data.cl
 [1] 1 1 1 1 1 1 0 0 0 0 0
>
```

G1_1	1
G1_2	1
G1_3	1
G1_4	1
G1_5	1
G1_6	1
G2_1	0
G2_2	0
G2_3	0
G2_4	0
G2_5	0

コードの解説：apply

apply関数は行ごとや列ごとに同じ関数を繰り返して実行させたい場合に便利。①dataオブジェクトの、②各行に対して、③cor関数を適用せよ。その際、④テンプレートyはdata.clとし、⑤相関係数の種類はparamで指定したものとする。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに
param <- "pearson"                       #相関係数の種類を指定("pearson"

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="")#in_f1で指定したファイルの読み込み
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]                      #テンプレートパターンベクトルdata.clを作成

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子についてテンプレートパターンdata.clとの相関

#ファイルに保存
tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係数rのベクトルを結合した結果をtmpに格納。
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保存
```



Tips：cor，データの型

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに
param <- "pearson"                       #相関係数の種類を指定("pearson"

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote=
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイル
data.cl <- hoge[,2]                      #テンプレートパターンベクトルdat

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子について
```

apply関数を使わずに、遺伝子（つまり行）ごとにcor関数を用いて相関係数を計算する手順。as.numeric関数は、data.clオブジェクトとデータの型を揃える目的で利用している。「互換性のない次元です」と言われているが、ここでは、相関係数を計算するときの入力データの見栄えが異なると文句を言われていると解釈すればよい。この「見栄え」が「データの型」と呼ばれるものに相当する。

```
#ファイルに保存
tmp <- cbind(r
write.table(tm
```

R Console

```
> r
     gene1       gene2       gene3
 0.9839815   0.8107810 -0.7068671
> data[1, ]
      G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5
gene1 6.44  6.3 6.51 6.36 6.49 6.39 3.58 4.39 4.25  3.7 4.09
> data.cl
 [1] 1 1 1 1 1 1 0 0 0 0 0
> cor(data[1, ], data.cl, method=param)
 以下にエラー cor(data[1, ], data.cl, method = param) :  互換性のない次元です
> as.numeric(data[1, ])
 [1] 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09
> cor(as.numeric(data[1, ]), data.cl, method=param)
[1] 0.9839815
>
```

Tips：as.vector, mode

慣れてくるとas.numericを使えばいいとすぐにわかるが、はじめのうちはas.character, as.integer, as.vectorなど既知のas…関数候補の中からそれっぽいものを試して型（見栄え）が揃うものを探すのが一般的かも…。modeというデータの型を表示する関数もある。

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定してin_f1に
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定してin_f2に
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに
param <- "pearson"                       #相関係数の種類を指定("pearson"

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f1, header=TRUE, row.names=1, sep="\t", quote="
hoge <- read.table(in_f2, sep="\t", quote="")#in_f2で指定したファイルの読み込み
data.cl <- hoge[,2]                      #テンプレートパターンベクトルdata.clを作成

#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子についてテンプレートパターンdata.clとの相関

#ファイルに保存
tmp <- cbind(r
write.table(tm
```

R Console

```
> data.cl
 [1] 1 1 1 1 1 1 0 0 0 0 0
> cor(data[1, ], data.cl, method=param)
 以下にエラー cor(data[1, ], data.cl, method = param) :  互換性のない次元です
> as.numeric(data[1, ])
 [1] 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09
> cor(as.numeric(data[1, ]), data.cl, method=param)
[1] 0.9839815
> as.vector(data[1, ])
      G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5
gene1 6.44  6.3 6.51 6.36 6.49 6.39 3.58 4.39 4.25  3.7 4.09
> mode(data[1, ])
[1] "list"
>
```

コードの解説：cbind

cbind関数を用いて出力させたい順番で列(column)方向で結合(bind)したものがtmpオブジェクト。ちなみに行(row)方向で結合させたい場合は、rbind関数を用いる。

```
#本番
r <- apply(data, 1, cor, y=data.cl, method=param)#各（行）遺伝子に

#ファイルに保存
tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係数rのベクトルを結合した結果をtmpに格納。
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保存
```

R Console

```
> #ファイルに保存
> tmp <- cbind(rownames(data), data, r)  #入力データの右側に相関係数rのベクトルを結合$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し$
> rownames(data)
[1] "gene1" "gene2" "gene3"
> data
      G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5
gene1 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09
gene2 5.81 6.93 6.73 5.55 6.39 6.61 2.81 5.46 1.00 3.46 4.17
gene3 3.91 4.81 5.04 3.17 4.75 5.36 5.58 5.52 5.70 5.64 5.61
> r
     gene1      gene2      gene3
 0.9839815  0.8107810 -0.7068671
> tmp
      rownames(data) G1_1 G1_2 G1_3 G1_4 G1_5 G1_6 G2_1 G2_2 G2_3 G2_4 G2_5          r
gene1          gene1 6.44 6.30 6.51 6.36 6.49 6.39 3.58 4.39 4.25 3.70 4.09  0.9839815
gene2          gene2 5.81 6.93 6.73 5.55 6.39 6.61 2.81 5.46 1.00 3.46 4.17  0.8107810
gene3          gene3 3.91 4.81 5.04 3.17 4.75 5.36 5.58 5.52 5.70 5.64 5.61 -0.7068671
>
```

Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

• 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test 教科書p108-116

t-test：仮想データ

このウェブページを用いたDEG検出手順の一般的なグループごとの反復数指定方法です。G1群は6反復、G2群は5反復。同一群でまとまっている、という前提です。

(Rで)マイクロアレイデータ解析

- 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | samroc (Broberg_2003) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | SAM (Tusher_2001) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test (last modified 2014/05/23) NEW
- 解析 | 発現変動 | 2群間 | 対応なし | Welch t-test (last modified 2014/05/23) NEW
- 解析 | 発現変動 | 2群間 | 対応なし | Mann-Whitney U-test (last modified 2013/10/15)

① ②

解析 | 発現変動 | 2群間 | 対応なし | Student's t-test NEW

等分散性を仮定したt検定を用いて、2群間での発現変動遺伝子の同定を行うやり方を示します。

出力ファイルの「p.value」列がp値、「ranking」列がp値に基づく順位、「q.value」列が任意のFDR閾値を満たすものを調べるときに用い

2. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用しないやり方です。

```
in_f <- "sample16_log.txt"                  #入力ファイル名を指定してin_fに格納
out_f <- "hoge2.txt"                        #出力ファイル名を指定してout_fに格納
param_G1 <- 6                               #G1群のサンプル数を指定
param_G2 <- 5                               #G2群のサンプル数を指定

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したファイルの読み込
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.clを作成

#等分散性を仮定（var.equal=T）してt.testを行い、t統計量とp-valueの値を返す関数Students_ttestを作成。
Students_ttest <- function(x, cl){
    x.class1 <- x[(cl == 1)]                #ラベルが1のものをx.class1に格納
    x.class2 <- x[(cl == 2)]                #ラベルが2のものをx.class2に格納
    if((sd(x.class1)+sd(x.class2)) == 0){#両方の群の標準偏差が共に0の場合は計算できないので...
        stat <- 0                           #統計量を0
        pval <- 1                           #p値を1
        return(c(stat, pval))               #として結果を返す
    }
    else{                                   #G1，G2どちらかの群の標準偏差が0(上記条件以外)の場合は
```

• 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test 教科書p108-116

t-test：仮想データ

このウェブページを用いたDEG検出手順の一般的なグループごとの反復数指定方法です。G1群は6反復、G2群は5反復。同一群でまとまっている、という前提です。gene1のような比較するグループ間（G1群 対 G2群）で明らかに発現の異なる遺伝子（DEG）の*p*値は0に近い値となり、明らかにDEGではないもの（non-DEG）の*p*値は1に近い値になるという感覚は重要。

2. サンプルデータ16のsample16_log.txt(対数変換後のデータ)の場合:

クラスラベル情報ファイル(sample16_cl.txt)を利用しないやり方です。

```
in_f <- "sample16_log.txt"                    #入力ファイル名を指定
out_f <- "hoge2.txt"                          #出力ファイル名を指定
param_G1 <- 6                                 #G1群のサンプル数を指
param_G2 <- 5                                 #G2群のサンプル数を指

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したファイルの読み込
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.clを作成

#等分散性を仮定（var.equal=T）してt testを行い
Students_ttest <- function(x, cl){
    x.class1 <- x[(cl == 1)]                  #ラベ
    x.class2 <- x[(cl == 2)]                  #ラベ
    if((sd(x.class1)+sd(x.class2)) == 0){#両方の群の標準偏差が共に0の場合は計算できないので...
        stat <- 0                             #統計量を0
        pval <- 1                             #p値を1
        return(c(stat, pval))                 #として結果を返す
```

rownan	G1_1	G1_2	G1_3	G1_4	G1_5	G1_6	G2_1	G2_2	G2_3	G2_4	G2_5
gene1	6.44	6.30	6.51	6.36	6.49	6.39	3.58	4.39	4.25	3.70	4.09
gene2	5.81	6.93	6.73	5.55	6.39	6.61	2.81	5.46	1.00	3.46	4.17
gene3	3.91	4.81	5.04	3.17	4.75	5.36	5.58	5.52	5.70	5.64	5.61

rownar	G1_1	G1_2	G1_3	G1_4	G1_5	G1_6	G2_1	G2_2	G2_3	G2_4	G2_5	p.value	q.value	ranking
gene1	6.44	6.30	6.51	6.36	6.49	6.39	3.58	4.39	4.25	3.70	4.09	4.77E-08	1.43E-07	1
gene2	5.81	6.93	6.73	5.55	6.39	6.61	2.81	5.46	1.00	3.46	4.17	0.002465	0.003697	2
gene3	3.91	4.81	5.04	3.17	4.75	5.36	5.58	5.52	5.70	5.64	5.61	0.015006	0.015006	3

t-test：乱数

N = 10,000遺伝子（行）からなる遺伝子発現行列（各群3サンプル）を入力として、遺伝子ごとにt-testを実行し、$p < 0.05$を満たす遺伝子数を眺めることを通じて多重比較問題を実感する

解析 | 発現変動 | 2群間 | 対応なし | Student's t-test NEW

等分散性を仮定したt検定を用いて、2群間での発現変動遺伝子の同定を行うやり方を示します。

3. サンプルデータ22のsample22.txtの場合:

10000行×6列分の標準正規分布に従う乱数です。G1群3サンプル vs. G2群3サンプルの2群間比較として解析を行っています。乱数を発生させただけのデータなので、発現変動遺伝子(DEG)がない全てがnon-DEGのデータです。

```
in_f <- "sample22.txt"                    #入力ファイル名を指定してin_fに格納
out_f <- "hoge3.txt"                      #出力ファイル名を指定してout_fに格納
param_G1 <- 3                             #G1群のサンプル数を指定
param_G2 <- 3                             #G2

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.
data.cl <- c(rep(1, param_G1), rep(2, para

#等分散性を仮定（var.equal=T）してt.testを行
Students_ttest <- function(x, cl){
    x.class1 <- x[(cl == 1)]              #ラ
    x.class2 <- x[(cl == 2)]              #ラ
    if((sd(x.class1)+sd(x.class2)) == 0){#
        stat <- 0                         #統
        pval <- 1                         #p値
        return(c(stat, pval))             #とりで結果を返す
    }
    else{                                 #G1, G2どちらかの群の標準偏差が0(上記条件以外)の場合に
        hoge <- t.test(x.class1, x.class2, var.equal=T)#通常のt検定を行って、
        return(c(hoge$statistic, hoge$p.value))#統計量とp値を結果として返す
    }
```

	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3
gene_1	−0.4458	0.9471	0.3047	−0.6163	−0.2745	0.0957
gene_2	−1.2059	−0.3745	0.2449	1.0468	−1.6776	−0.0616
gene_3	0.0411	−0.2297	−0.6740	1.9160	−1.2744	1.8971
gene_4	0.6394	−0.0078	0.7516	−0.3792	1.7884	−1.2589
gene_5	−0.7866	1.1538	−0.0819	0.5820	−0.5949	2.5911
gene_6	−0.3855	−2.4745	1.1373	−1.6164	0.4605	−0.6176
gene_7	−0.4759	−1.8536	0.0646	1.7821	−0.0340	1.3479
gene_8	0.7198	0.8058	−2.3137	−0.2549	−0.6822	−0.6548
gene_9	−0.0185	−1.5977	−2.0602	1.1553	0.6388	2.0476
gene_10	−1.3731	0.0591	0.2841	−0.0105	0.2229	0.2152
gene_11	−0.9824	−0.5004	−1.3967	−1.3212	0.3319	−2.1954

t-test：乱数

data.clオブジェクトは、テンプレートパターンのようなもの。入力データに相当するdataオブジェクトの1-3列がG1群、4-6列がG2群由来サンプルだということを指し示すクラスラベル情報

3. サンプルデータ22のsample22.txtの場合:

10000行×6列分の 標準正規分布に従う乱数です。G1群3サンプル vs. G2群3サンプル
います。乱数を発生させただけのデータなので、発現変動遺伝子(DEG)がない全てが non-DEGのデータです。

```
in_f <- "sample22.txt"                    #入力ファイル名を指定してin_fに格納
out_f <- "hoge3
param_G1 <- 3
param_G2 <- 3

#入力ファイルの読
data <- read.ta
data.cl <- c(re

#等分散性を仮定
Students_ttest
    x.class1 <-
    x.class2 <-
    if((sd(x.cl
        stat <-
        pval <-
        return(
    }
    else{
        hoge <-
        return(
    }
```

R Console

```
> getwd()
[1] "C:/Users/kadota/Desktop/hoge"
> list.files(pattern="22")
[1] "sample22.txt"
> in_f <- "sample22.txt"                    #入力ファイル名を指定してin_fに$
> out_f <- "hoge3.txt"                      #出力ファイル名を指定してout_fに$
> param_G1 <- 3                             #G1群のサンプル数を指定
> param_G2 <- 3                             #G2群のサンプル数を指定
>
> #入力ファイルの読み込みとラベル情報の作成
> data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#$
> data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2とした$
> head(data, n=3)
           G1_rep1    G1_rep2    G1_rep3   G2_rep1    G2_rep2     G2_rep3
gene_1 -0.44577826  0.9470852  0.3047168 -0.616254 -0.2744629  0.09569090
gene_2 -1.20585657 -0.3744989  0.2449209  1.046830 -1.6776099 -0.06155844
gene_3  0.04112631 -0.2297141 -0.6740500  1.916020 -1.2743934  1.89711799
> data.cl
[1] 1 1 1 2 2 2
>
```

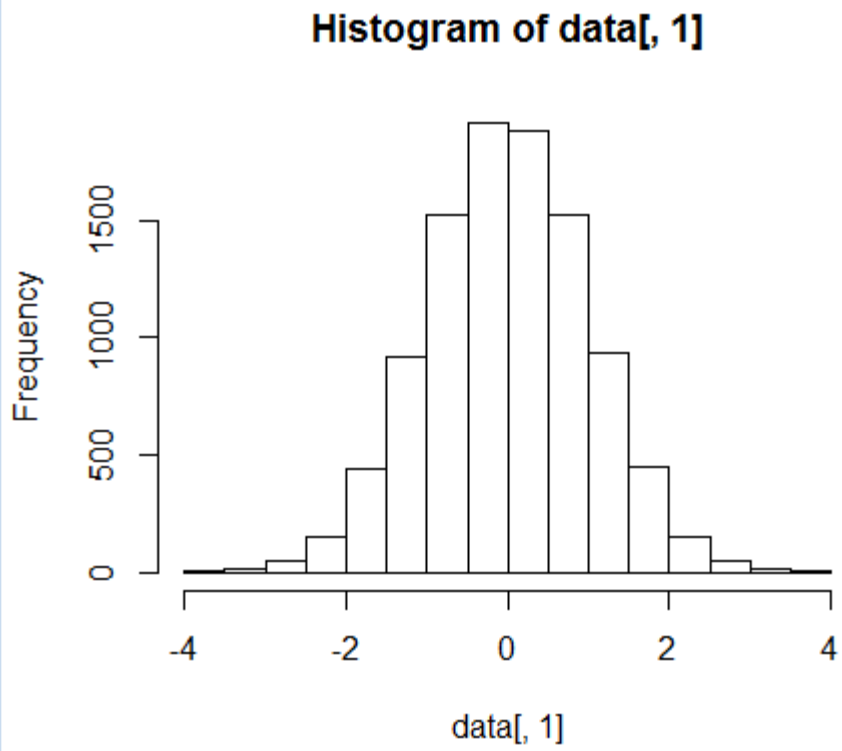
確かに（標準）正規分布乱数になっている

t-test：乱数

R Console

```
> summary(data)
    G1_rep1            G1_rep2            G1_rep3
 Min.   :-3.561788   Min.   :-4.6296   Min.   :-3.97209
 1st Qu.:-0.673850   1st Qu.:-0.6694   1st Qu.:-0.65026
 Median :-0.001270   Median :-0.0038   Median : 0.01834
 Mean   : 0.001152   Mean   :-0.0036   Mean   : 0
 3rd Qu.: 0.679024   3rd Qu.: 0.6595   3rd Qu.: 0
 Max.   : 3.739140   Max.   : 3.9571   Max.   : 4
    G2_rep1            G2_rep2            G2_rep
 Min.   :-3.63480   Min.   :-4.096237   Min.   :-
 1st Qu.:-0.68904   1st Qu.:-0.676368   1st Qu.:-
 Median :-0.01723   Median :-0.012488   Median :-
 Mean   :-0.01278   Mean   :-0.003493   Mean   :
 3rd Qu.: 0.65439   3rd Qu.: 0.659750   3rd Qu.:
 Max.   : 3.83720   Max.   : 3.514435   Max.   :
> hist(data[, 1])
>
```

R Graphics: Device 2 (ACTIVE)



t-test：乱数

特定の条件を満たす
列のみ取り出すやり方

R Console

```
> head(data, n=4)
          G1_rep1      G1_rep2    G1_rep3    G2_rep1    G2_rep2     G2_rep3
gene_1 -0.44577826  0.947085174  0.3047168 -0.6162540 -0.2744629  0.09569090
gene_2 -1.20585657 -0.374498928  0.2449209  1.0468297 -1.6776099 -0.06155844
gene_3  0.04112631 -0.229714096 -0.6740500  1.9160200 -1.2743934  1.89711799
gene_4  0.63938841 -0.007755371  0.7515938 -0.3791546  1.7883961 -1.25888353
> data[1, ]
          G1_rep1   G1_rep2   G1_rep3   G2_rep1    G2_rep2   G2_rep3
gene_1 -0.4457783 0.9470852 0.3047168 -0.616254 -0.2744629 0.0956909
> data.cl == 1
[1]  TRUE  TRUE  TRUE FALSE FALSE FALSE
> data.cl == 2
[1] FALSE FALSE FALSE  TRUE  TRUE  TRUE
> data[1, data.cl==1]
          G1_rep1   G1_rep2   G1_rep3
gene_1 -0.4457783 0.9470852 0.3047168
> data[1, data.cl==2]
         G2_rep1    G2_rep2   G2_rep3
gene_1 -0.616254 -0.2744629 0.0956909
>
```

t-test：乱数

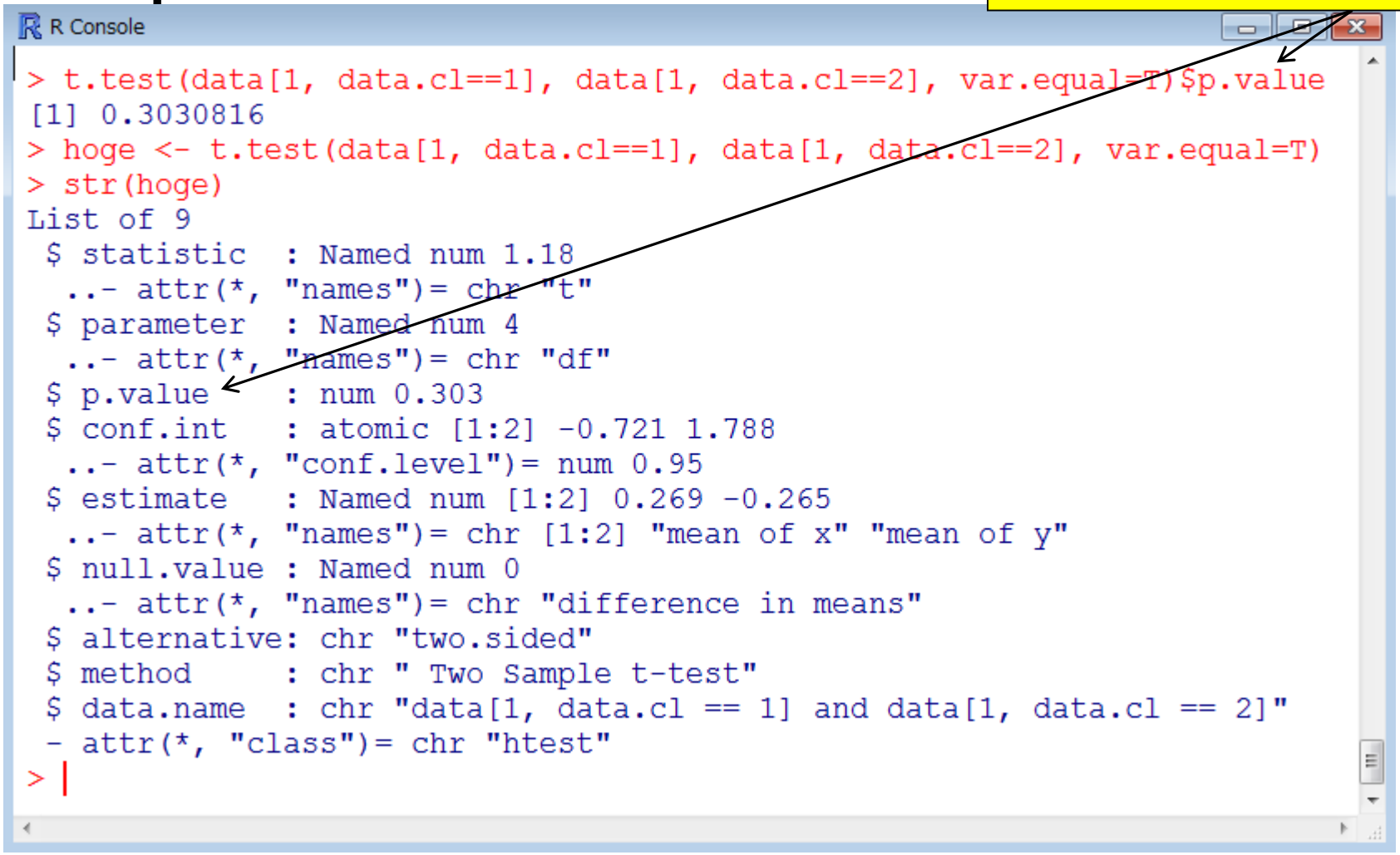
有意水準 α = 0.05とすると、1行目の遺伝子の*p*値は0.05未満ではない。2行目、3行目、…。

R Console

```
> data[1, data.cl==1]
          G1_rep1   G1_rep2   G1_rep3
gene_1 -0.4457783 0.9470852 0.3047168
> data[1, data.cl==2]
         G2_rep1    G2_rep2   G2_rep3
gene_1 -0.616254 -0.2744629 0.0956909
> t.test(data[1, data.cl==1], data[1, data.cl==2], var.equal=T)

        Two Sample t-test

data:  data[1, data.cl == 1] and data[1, data.cl == 2]
t = 1.1808, df = 4, p-value = 0.3031
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.7211295  1.7884960
sample estimates:
 mean of x  mean of y
 0.2686746 -0.2650087

> t.test(data[1, data.cl==1], data[1, data.cl==2], var.equal=T)$p.value
[1] 0.3030816
>
```

Tips

str関数を利用すれば、t.test関数に限らず、出力結果からどのような情報を取り出せるかを知ることができます。



t-test：乱数

出力ファイル(hoge3.txt)は、入力ファイル情報の右側にp.value, q.value, rankingという列の情報が追加されたものです。

3. サンプルデータ22のsample22.txtの場合:

10000行×6列分の標準正規分布に従う乱数です。G1群3サンプル vs. G2群3サンプルの2群間比較として解析を行っています。乱数を発生させただけのデータなので、発現変動遺伝子(DEG)がない全てがnon-DEGのデータです。

```
    x.class2 <- x[(cl == 2)]              #ラベルが2のものをx.class2に格納
    if((sd(x.class1)+sd(x.class2)) == 0){#両方の群の標準偏差が共に0の場合は計算できないので..
        stat <- 0                         #統計量を0
        pval <- 1                         #p値を1
        return(c(stat, pval))             #として結果を返す
    }
    else{                                 #G1, G2どちらかの群の標準偏差が0(上記条件以外)の場合に
        hoge <- t.test(x.class1, x.class2, var.equal=T)#通常のt検定を行って、
        return(c(hoge$statistic, hoge$p.value))#統計量とp値を結果として返す
    }
}

#本番
out <- t(apply(data, 1, Students_ttest, data.cl))#各（行）遺伝子についてt検定を行った結果のt
p.value <- out[,2]                        #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)                  #p.valueでランキングした結果をrankingに格納

#ファイルに保存
tmp <- cbind(rownames(data), data, p.value, q.value, ranking)#入力データの右側にp.value、q
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイ
```

t-test：乱数

p.value列かranking列で昇順にソートすると、発現変動順になる。

R Console

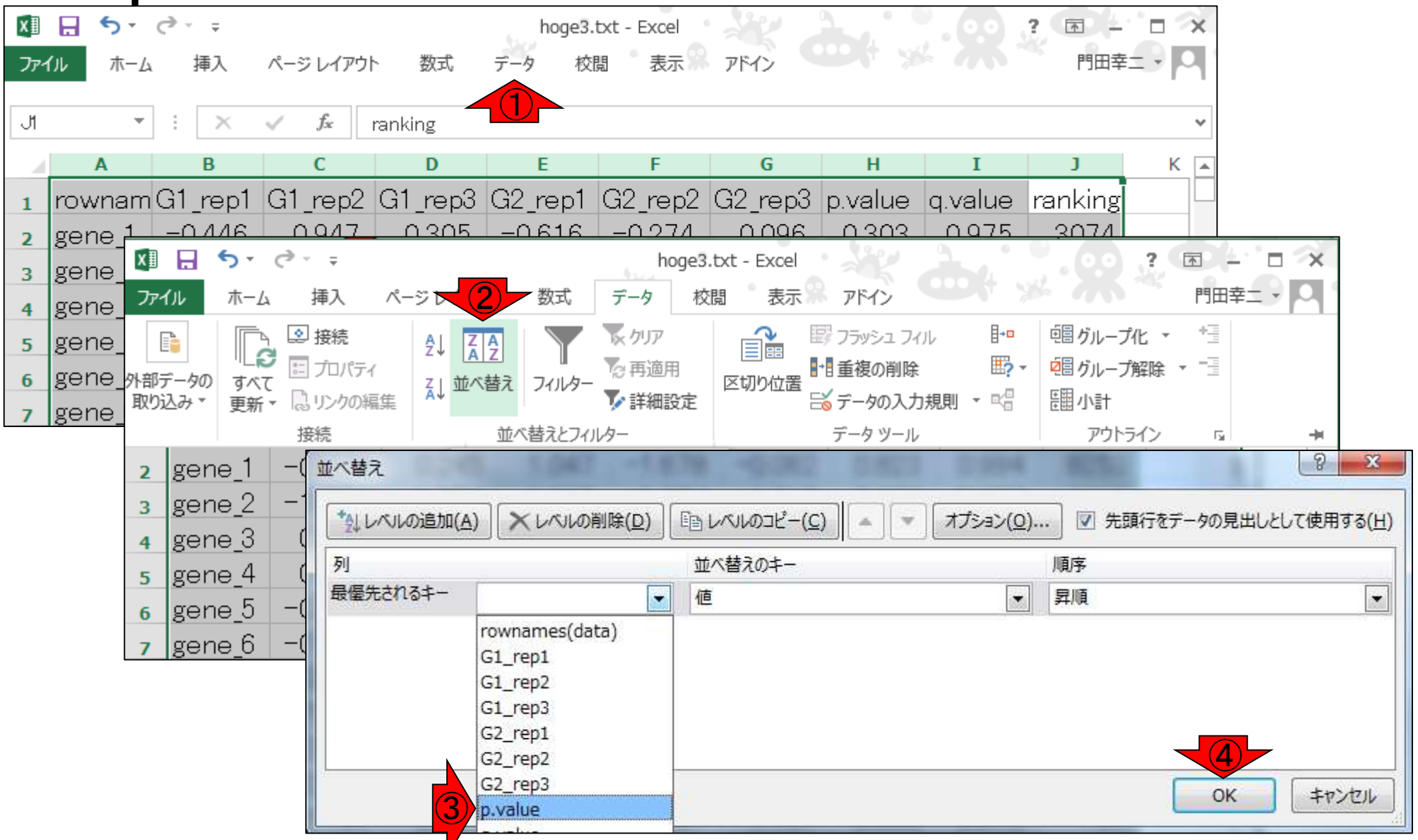
```
> #ファイルに保存
> tmp <- cbind(rownames(data), data, p.value, q.value, ranking)#入力デ$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tm$
> head(tmp, n=3)
       rownames(data)     G1_rep1    G1_rep2    G1_rep3   G2_rep1
gene_1         gene_1 -0.44577826  0.9470852  0.3047168 -0.616254
gene_2         gene_2 -1.20585657 -0.3744989  0.2449209  1.046830
gene_3         gene_3  0.04112631 -0.2297141 -0.6740500  1.916020
          G2_rep2     G2_rep3   p.value   q.value ranking
gene_1 -0.2744629  0.09569090 0.3030816 0.9750872    3074
gene_2 -1.6776099 -0.0
gene_3 -1.2743934  1.8
>
```

hoge3.txt

rownam	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking
gene_1	−0.446	0.947	0.305	−0.616	−0.274	0.096	0.303	0.975	3074
gene_2	−1.206	−0.374	0.245	1.047	−1.678	−0.062	0.823	0.994	8253
gene_3	0.041	−0.230	−0.674	1.916	−1.274	1.897	0.353	0.977	3615
gene_4	0.639	−0.008	0.752	−0.379	1.788	−1.259	0.683	0.994	6828
gene_5	−0.787	1.154	−0.082	0.582	−0.595	2.591	0.522	0.994	5209
gene_6	−0.385	−2.474	1.137	−1.616	0.460	−0.618	0.989	0.998	9914
gene_7	−0.476	−1.854	0.065	1.782	−0.034	1.348	0.087	0.975	852
gene_8	0.720	0.806	−2.314	−0.255	−0.682	−0.655	0.809	0.994	8112
gene_9	−0.019	−1.598	−2.060	1.155	0.639	2.048	0.028	0.975	277
gene_10	−1.373	0.059	0.284	−0.011	0.223	0.215	0.407	0.989	4111
gene_11	−0.982	−0.500	−1.397	−1.321	0.332	−2.195	0.903	0.994	9075
gene_12	−0.554	−1.795	−0.657	0.391	−0.387	1.137	0.080	0.975	796

EXCELでのソートの仕方

Tips



Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

DEGの存在しないnon-DEGのみからなるデータなので妥当

多重比較問題のイントロ

rownames(c	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking
gene_2313	0.172	0.422	0.234	−1.211	−1.399	−1.564	0.0002	0.9025	1
gene_7754	1.514	1.426	1.814	−0.324	−0.615	−0.275	0.0002	0.9025	2
gene_1175	−1.437	−1.029	−1.326	1.238	1.336	0.780	0.0003	0.9025	3
gene_766	−1.162	−0.936	−1.521	0.538	0.775	0.899	0.0006	0.9025	4
gene_9001	0.648	0.386	0.489	−0.737	−0.610	−0.428	0.0007	0.9025	5
gene_5866	−1.956	−2.084	−1.932	−0.132	−0.001	0.611	0.0008	0.9025	6
gene_5818	−0.999	−1.526	−1.013	0.860	0.705	1.237	0.0008	0.9025	7
gene_4882	1.911	1.145	0.782	−0.007	−0.983	0.408	0.0498	0.9751	490
gene_8919	0.508	1.239	0.396	−0.149	−0.415	0.135	0.0498	0.9751	491
gene_2545	−1.248	−2.039	−0.106	0.366	0.674	0.332	0.0499	0.9751	492
gene_8229	0.262	0.250	0.715	−0.427	−0.675	0.081	0.0500	0.9751	493
gene_9729	−1.113	−1.714	−0.291	−0.042	0.654	0.123	0.0500	0.9751	494
gene_2484	0.296	0.249	2.137	−0.718	−0.787	−1.059	0.0501	0.9751	495
gene_7406	−0.700	0.393	−1.023	0.136	0.709	1.119	0.0997	0.9751	957
gene_924	0.473	1.117	0.998	0.647	−0.102	−0.597	0.0998	0.9751	958
gene_872	0.236	−0.057	1.111	−1.812	0.031	−1.014	0.0999	0.9751	959
gene_7666	0.112	1.531	1.352	0.491	−0.575	−1.307	0.1003	0.9751	960
gene_4154	1.346	1.255	−0.047	−0.114	0.102	−0.876	0.1003	0.9751	961
gene_8055	−0.590	−0.454	0.163	0.145	0.138	0.543	0.1004	0.9751	962
gene_724	−0.442	1.969	1.071	0.767	1.595	0.236	0.9997	0.9999	9998
gene_287	1.677	0.395	−1.040	−0.120	0.370	0.783	0.9998	0.9999	9999
gene_9776	0.942	−0.389	−0.424	−0.843	−0.763	1.735	1.0000	1.0000	10000

①

$p < 0.05$を満たす遺伝子数は492個

②

$p < 0.10$を満たす遺伝子数は959個

Type-I error（false positive）

ランダムデータの場合

- 有意水準αでN回の検定（多重比較）を行うと、（$N \times \alpha$）個のFalse Positiveが得られる。
- 10000個の遺伝子（N=10000）に対して$p < 0.05$を満たすものを調べる（有意水準αを0.05に設定することと同義）と（$N \times \alpha$）個程度が本当は発現変動遺伝子(Differentially Expressed Genes; DEGs)でないにもかかわらず発現変動遺伝子と判断されてしまう。

p値だけである程度判断できる

- うれしくない結果：「実際に得られた発現変動遺伝子数 ≒（解析遺伝子数N× 設定した有意水準 α）個」
  - **このデータ中には「発現変動遺伝子（DEG）はない」**と判断する。
- うれしい結果：「実際に得られた発現変動遺伝子数 >>（解析遺伝子数N× 設定した有意水準 α）個」
  - **このデータ中には「真の発現変動遺伝子が存在する」**ことが期待される。
- 実際に利用されているRパッケージの多くは、（多重比較を考慮した補正後のp-valueに相当する）q-valueの値を出力する
  - （p値利用時の有意水準 αに相当する）False Discovery Rate（FDR）の閾値を満たす遺伝子数を頼りに発現変動遺伝子の有無を判断する

ここではq-valueとしているが、adjusted p-valueと呼ばれるものと同じです。Rパッケージ出力結果でもadjPやPadjやFDRなどの列がありますが、それに相当する議論です。

多重比較問題：FDRって何?

- ***p***–value (false positive rate; FPR)
  - 本当はDEGではないにもかかわらずDEGと判定してしまう確率
  - 全遺伝子に占めるnon–DEGの割合（分母は**遺伝子総数**）
  - 例：10,000個のnon–DEGからなる遺伝子をp–value < 0.05で検定すると、10,000 × 0.05 = 500個程度のnon–DEGを間違ってDEGと判定することに相当
    - 実際のDEG検出結果が900個だった場合：500個は偽物で400個は本物と判断
    - 実際のDEG検出結果が510個だった場合：500個は偽物で10個は本物と判断
    - 実際のDEG検出結果が500個以下の場合：全て偽物と判断
- ***q***–value （false discovery rate: FDR）
  - DEGと判定した中に含まれるnon–DEGの割合
  - DEG中に占めるnon–DEGの割合（分母は**DEGと判定された数**）
  - non–DEGの期待値を計算できれば、p値でも上位x個でもDEGと判定する手段はなんでもよい。以下は10,000遺伝子の検定結果でのFDR計算例
    - p < 0.001を満たすDEG数が100個の場合：FDR = 10,000 × 0.001/100 = 0.1
    - p < 0.01を満たすDEG数が400個の場合：FDR = 10,000 × 0.01/400 = 0.25
    - p < 0.05を満たすDEG数が926個の場合：FDR = 10,000 × 0.05/926 = 0.54

Benjamini and Hochberg *J. Roy. Stat. Soc. B*, **57**: 289–300, 1995.

多重比較問題：FDRって何?

DEG数に関するよりよい結果を得たい場合には、***p***–valueではなく***q***–valueを利用しましょう。（閾値を有意水準 α ではなくFDRで設定しましょう。）

- DEGかnon–DEGかを判定する閾値を決める問題
  - 有意水準5%というのが***p*–value** < 0.05に相当
  - False discovery rate (FDR) 5%というのが***q*–value** < 0.05に相当
- **発現変動ランキング結果は不変**なので上位x個という決め打ちの場合にはこの問題とは無関係

rownames(c	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking
gene_2313	0.172	0.422	0.234	−1.211	−1.399	−1.564	0.0002	0.9025	1
gene_7754	1.514	1.426	1.814	−0.324	−0.615	−0.275	0.0002	0.9025	2
gene_1175	−1.437	−1.029	−1.326	1.238	1.336	0.780	0.0003	0.9025	3
gene_766	−1.162	−0.936	−1.521	0.538	0.775	0.899	0.0006	0.9025	4
gene_9001	0.648	0.386	0.489	−0.737	−0.610	−0.428	0.0007	0.9025	5
gene_5866	−1.956	−2.084	−1.932	−0.132	−0.001	0.611	0.0008	0.9025	6
gene_5818	−0.999	−1.526	−1.013	0.860	0.705	1.237	0.0008	0.9025	7

多重比較問題

q < 0.05を満たす遺伝子数は0個。DEGの存在しないnon-DEGのみからなるデータなので妥当

rownames(c	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking
gene_2313	0.172	0.422	0.234	−1.211	−1.399	−1.564	0.0002	0.9025	1
gene_7754	1.514	1.426	1.814	−0.324	−0.615	−0.275	0.0002	0.9025	2
gene_1175	−1.437	−1.029	−1.326	1.238	1.336	0.780	0.0003	0.9025	3
gene_766	−1.162	−0.936	−1.521	0.538	0.775	0.899	0.0006	0.9025	4
gene_9001	0.648	0.386	0.489	−0.737	−0.610	−0.428	0.0007	0.9025	5
gene_5866	−1.956	−2.084	−1.932	−0.132	−0.001	0.611	0.0008	0.9025	6
gene_5818	−0.999	−1.526	−1.013	0.860	0.705	1.237	0.0008	0.9025	7
gene_4882	1.911	1.145	0.782	−0.007	−0.983	0.408	0.0498	0.9751	490
gene_8919	0.508	1.239	0.396	−0.149	−0.415	0.135	0.0498	0.9751	491
gene_2545	−1.248	−2.039	−0.106	0.366	0.674	0.332	0.0499	0.9751	492
gene_8229	0.262	0.250	0.715	−0.427	−0.675	0.081	0.0500	0.9751	493
gene_9729	−1.113	−1.714	−0.291	−0.042	0.654	0.123	0.0500	0.9751	494
gene_2484	0.296	0.249	2.137	−0.718	−0.787	−1.059	0.0501	0.9751	495
gene_7406	−0.700	0.393	−1.023	0.136	0.709	1.119	0.0997	0.9751	957
gene_924	0.473	1.117	0.998	0.647	−0.102	−0.597	0.0998	0.9751	958
gene_872	0.236	−0.057	1.111	−1.812	0.031	−1.014	0.0999	0.9751	959
gene_7666	0.112	1.531	1.352	0.491	−0.575	−1.307	0.1003	0.9751	960
gene_4154	1.346	1.255	−0.047	−0.114	0.102	−0.876	0.1003	0.9751	961
gene_8055	−0.590	−0.454	0.163	0.145	0.138	0.543	0.1004	0.9751	962
gene_724	−0.442	1.969	1.071	0.767	1.595	0.236	0.9997	0.9999	9998
gene_287	1.677	0.395	−1.040	−0.120	0.370	0.783	0.9998	0.9999	9999
gene_9776	0.942	−0.389	−0.424	−0.843	−0.763	1.735	1.0000	1.0000	10000

①

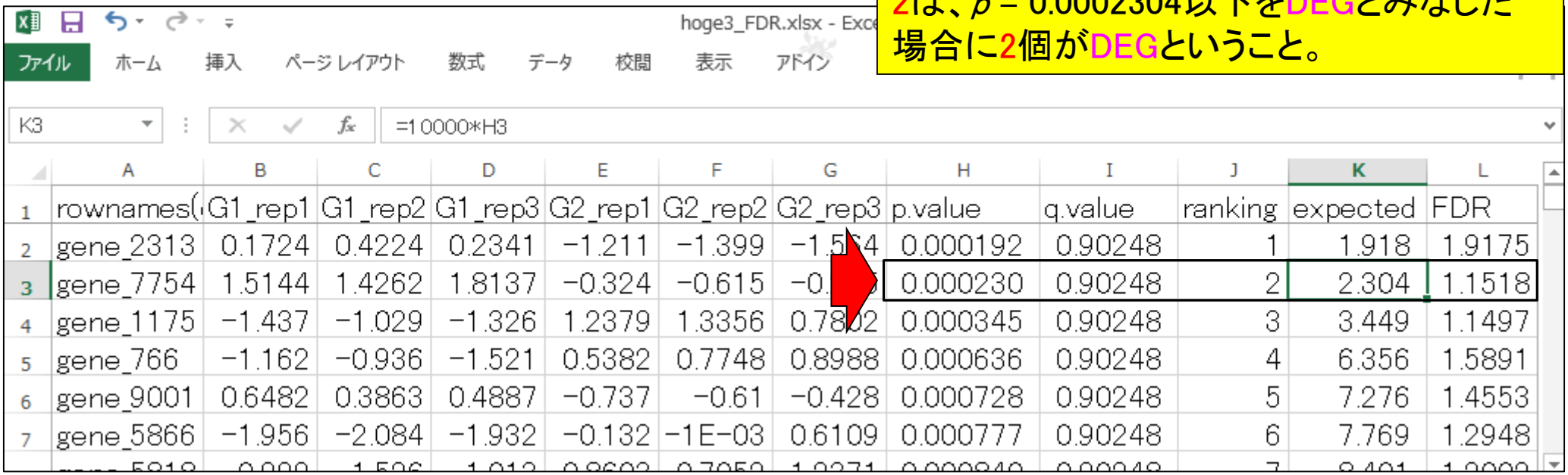
p < 0.05を満たす遺伝子数は492個

②

p < 0.10を満たす遺伝子数は959個

FDR

expected = 全遺伝子数 × p–value =
expected = 10,000 × 0.0002304 = 2.304。
FDR = 偽物検出割合 =
expected/observed = 2.304 / 2 = 1.1518。
2は、p = 0.0002304以下をDEGとみなした場合に2個がDEGということ。

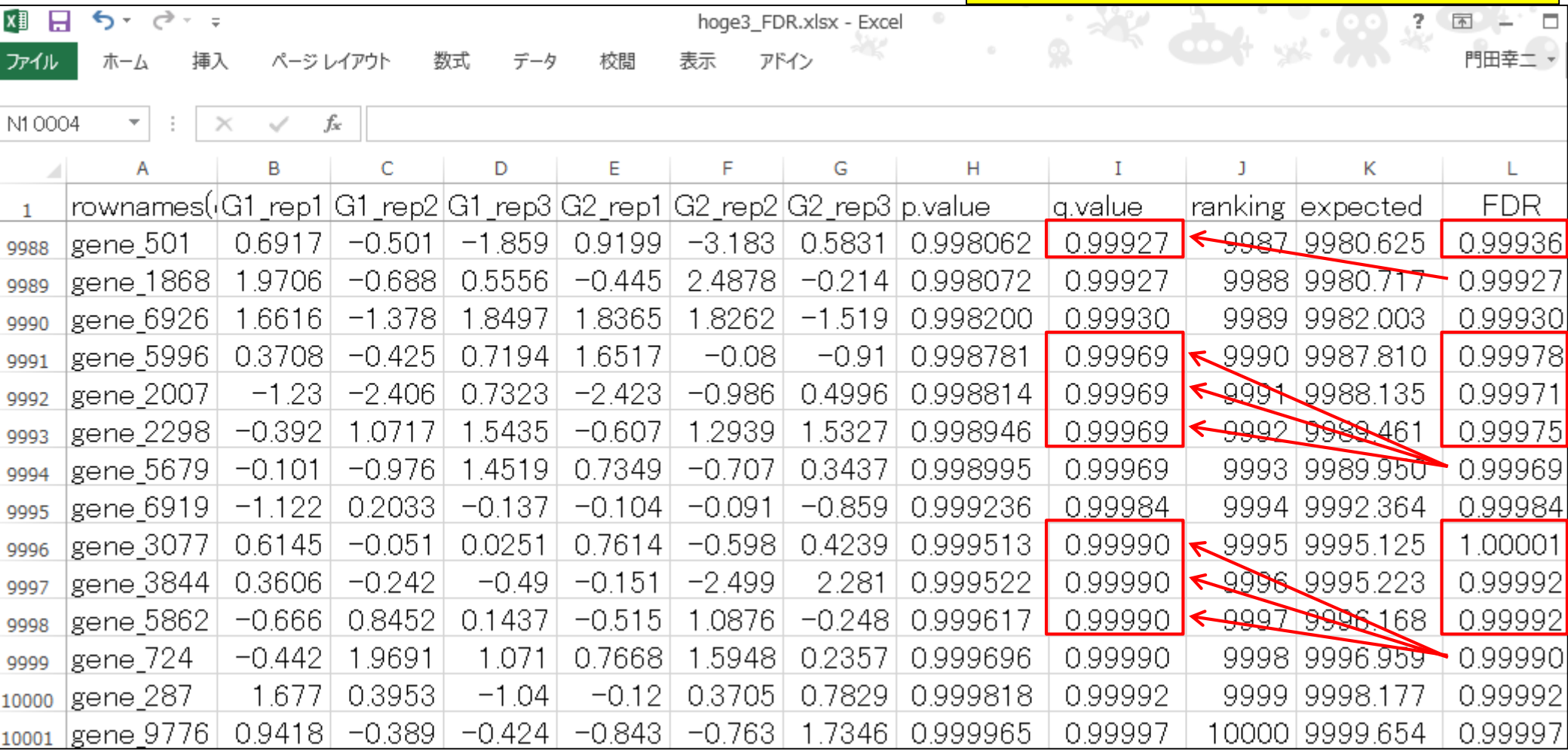


	A	B	C	D	E	F	G	H	I	J	K	L
1	rownames(	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking	expected	FDR
2	gene_2313	0.1724	0.4224	0.2341	−1.211	−1.399	−1.5[illegible]4	0.000192	0.90248	1	1.918	1.9175
3	gene_7754	1.5144	1.4262	1.8137	−0.324	−0.615	−0.[illegible]	0.000230	0.90248	2	2.304	1.1518
4	gene_1175	−1.437	−1.029	−1.326	1.2379	1.3356	0.78[illegible]2	0.000345	0.90248	3	3.449	1.1497
5	gene_766	−1.162	−0.936	−1.521	0.5382	0.7748	0.8988	0.000636	0.90248	4	6.356	1.5891
6	gene_9001	0.6482	0.3863	0.4887	−0.737	−0.61	−0.428	0.000728	0.90248	5	7.276	1.4553
7	gene_5866	−1.956	−2.084	−1.932	−0.132	−1E−03	0.6109	0.000777	0.90248	6	7.769	1.2948

基本的にこの2つは同じものという理解でよい。より正確には、FDR列の情報をもとに値の分布が滑らかになるように細工しているのがq.value列の数値

FDR

p.valueが高いもののFDR値から順に見ていって、FDR値が低くなるように置換していったものがq.value列の値



	A	B	C	D	E	F	G	H	I	J	K	L
1	rownames(	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking	expected	FDR
9988	gene_501	0.6917	−0.501	−1.859	0.9199	−3.183	0.5831	0.998062	0.99927	9987	9980.625	0.99936
9989	gene_1868	1.9706	−0.688	0.5556	−0.445	2.4878	−0.214	0.998072	0.99927	9988	9980.717	0.99927
9990	gene_6926	1.6616	−1.378	1.8497	1.8365	1.8262	−1.519	0.998200	0.99930	9989	9982.003	0.99930
9991	gene_5996	0.3708	−0.425	0.7194	1.6517	−0.08	−0.91	0.998781	0.99969	9990	9987.810	0.99978
9992	gene_2007	−1.23	−2.406	0.7323	−2.423	−0.986	0.4996	0.998814	0.99969	9991	9988.135	0.99971
9993	gene_2298	−0.392	1.0717	1.5435	−0.607	1.2939	1.5327	0.998946	0.99969	9992	9989.461	0.99975
9994	gene_5679	−0.101	−0.976	1.4519	0.7349	−0.707	0.3437	0.998995	0.99969	9993	9989.950	0.99969
9995	gene_6919	−1.122	0.2033	−0.137	−0.104	−0.091	−0.859	0.999236	0.99984	9994	9992.364	0.99984
9996	gene_3077	0.6145	−0.051	0.0251	0.7614	−0.598	0.4239	0.999513	0.99990	9995	9995.125	1.00001
9997	gene_3844	0.3606	−0.242	−0.49	−0.151	−2.499	2.281	0.999522	0.99990	9996	9995.223	0.99992
9998	gene_5862	−0.666	0.8452	0.1437	−0.515	1.0876	−0.248	0.999617	0.99990	9997	9996.168	0.99992
9999	gene_724	−0.442	1.9691	1.071	0.7668	1.5948	0.2357	0.999696	0.99990	9998	9996.959	0.99990
10000	gene_287	1.677	0.3953	−1.04	−0.12	0.3705	0.7829	0.999818	0.99992	9999	9998.177	0.99992
10001	gene_9776	0.9418	−0.389	−0.424	−0.843	−0.763	1.7346	0.999965	0.99997	10000	9999.654	0.99997

教科書p115　参考

・FDR = 偽物検出割合
・FDR = expected/observed

自力でq-value (FDR)計算

1	rownames(	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value	q.value	ranking	expected	FDR
491	gene_4882	1.911	1.1447	0.7818	−0.007	−0.983	0.4081	0.049820	0.97509	490	498.198	1.01673
492	gene_8919	0.5084	1.2393	0.3958	−0.149	−0.415	0.1354	0.049845	0.97509	491	498.453	1.01518
493	gene_2545	−1.248	−2.039	−0.106	0.3658	0.6739	0.3315	0.049911	0.97509	492	499.114	1.01446
494	gene_8229	0.262	0.2501	0.715	−0.427	−0.675	0.0809	0.050047	0.97509	493	500.468	1.01515
495	gene_9729	−1.113	−1.714	−0.291	−0.042	0.6544	0.1234	0.050049	0.97509	494	500.489	1.01314
496	gene_2484	0.2956	0.2492	2.1367	−0.718	−0.787	−1.059	0.050095	0.97509	495	500.949	1.01202

R Console

```
> head(p.value)
   gene_1    gene_2    gene_3    gene_4    gene_5    gene_6
0.3030816 0.8226226 0.3532968 0.6832627 0.5216252 0.9894427
> observed <- sum(p.value < 0.05)
> observed
[1] 492
> length(p.value)
[1] 10000
> expected <- length(p.value)*0.05
> expected
[1] 500
> FDR <- expected/observed
> FDR
[1] 1.01626
>
```

自力でq-value (FDR)計算

*p*値計算結果が手元にあれば(つまりp.valueオブジェクトがあれば)このコードを実行することによってFDRの概要がわかります

(Rで)マイクロアレイデータ解析

(last modified 2014/05/23, since 2005)

- 書籍 | トランスクリプトーム解析 | 3.2.1 クラスタリング(データ変換や距離の定義など) (last modified
- 書籍 | トランスクリプトーム解析 | 3.2.2 実験デザイン、データ分布、統計解析との関係 (last modified
- 書籍 | トランスクリプトーム解析 | 3.2.3 多重比較問題 (last modified 2014/04/19)
- 書籍 | トランスクリプトーム解析 | 3.2.4 各種プロット(M-A plotや平均-分散プロットなど) (last modifie
- 書籍 | トランスクリプトーム解析 | 4.2.1 2群間比較 (last modified 2014/04/19)

書籍 | トランスクリプトーム解析 | 3.2.3 多重比較問題

シリーズ Useful R 第7巻トランスクリプトーム解析のp111-121のRコードです。Windowsの場合、コピペは「CTRLキーとALTキーを押しながら枠内で左クリック」でコード内を全選択できます。

この項目

p111の

前項か

p111下:

p115-116の網掛け部分:

```
threshold <- c(0.001, 0.01, 0.03, 0.05, 0.10)
res <- NULL
for(i in 1:length(threshold)){
  observed <- sum(p.value < threshold[i])
  expected <- nrow(data)*threshold[i]
  FDR <- expected/observed
  res <- rbind(res, c(threshold[i], observed, expected, FDR))
}
colnames(res) <- c("threshold", "observed", "expected", "FDR")
```

自力でq-value (FDR)計算

ここで指定しているのはp-valueの閾値（つまり有意水準）です

p115-116の網掛け部分:

```
threshold <- c(0.001, 0.01, 0.03, 0.05, 0.10)
res <- NULL
for(i in 1:length(threshold)){
  ob
  ex
  FD
  re
}
coln
```

R Console

```
> threshold <- c(0.001, 0.01, 0.03, 0.05, 0.10)
> res <- NULL
> for(i in 1:length(threshold)){
+   observed <- sum(p.value < threshold[i])
+   expected <- nrow(data)*threshold[i]
+   FDR <- expected/observed
+   res <- rbind(res, c(threshold[i], observed, expected, FD$
+ }
> colnames(res) <- c("threshold", "observed", "expected", "F$
> res
     threshold observed expected       FDR
[1,]     0.001        8       10 1.2500000
[2,]     0.010      104      100 0.9615385
[3,]     0.030      295      300 1.0169492
[4,]     0.050      492      500 1.0162602
[5,]     0.100      959     1000 1.0427529
> 
```

Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

t-test：DEGを含むデータ

10,000個中1,000個がG1群で高発現の2群間比較用シミュレーションデータ。確かにG1群で高発現になっていることがわかります

解析 | 発現変動 | 2群間 | 対応なし | Student's t-test NEW

等分散性を仮定したt検定を用いて、2群間での発現変動遺伝子の同定を行うやり方を示します。

4. サンプルデータ23のsample23.txtの場合:

最初の3サンプルがG1群、残りの3サンプルがG2群の標準正規分布に従う乱数からなるシミュレーションデータです。乱数発生後に、さらに最初の10%分についてG1群に相当するところのみ数値を+3している（つまり10%がG1群で高発現というシミュレーションデータを作成している）

```
in_f <- "sample23.txt"                    #入力ファイル名を指定してin_fに格納
out_f <- "hoge4.txt"                      #出力ファイル名を指定してout_fに格納
param_G1 <- 3
param_G2 <- 3

#入力ファイルの
data <- read.
data.cl <- c(

#等分散性を仮定
Students_ttes
    x.class1
    x.class2
    if((sd(x.
```

R Console

```
> in_f <- "sample23.txt"                    #入力ファイル名を指定してin$
> out_f <- "hoge4.txt"                      #出力ファイル名を指定してou$
> param_G1 <- 3                             #G1群のサンプル数を指定
> param_G2 <- 3                             #G2群のサンプル数を指定
>
> #入力ファイルの読み込みとラベル情報の作成
> data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote$
> data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2$
> head(data)
         G1_rep1   G1_rep2  G1_rep3    G2_rep1    G2_rep2     G2_rep3
gene_1 2.554222 3.9470852 3.304717 -0.6162540 -0.2744629  0.09569090
gene_2 1.794143 2.6255011 3.244921  1.0468297 -1.6776099 -0.06155844
gene_3 3.041126 2.7702859 2.325950  1.9160200 -1.2743934  1.89711799
gene_4 3.639388 2.9922446 3.751594 -0.3791546  1.7883961 -1.25888353
gene_5 2.213446 4.1538056 2.918114  0.5819835 -0.5949475  2.59110024
gene_6 2.614511 0.5255313 4.137273 -1.6164433  0.4604567 -0.61764893
>
```

t-test：DEGを含むデータ

①p < 0.05を満たす遺伝子数は1,226個。期待値は500個なので、(1,226 − 500)個程度が本物だと判断する。②q < 0.20を満たす遺伝子数は388個。FDR = 0.20なので、388*0.2 = 77.6個は偽物で残りの80%は本物だと判断する。

4. サンプルデータ23のsample23.txtの場合:

最初の3サンプルがG1群、残りの3サンプルがG2群の標準正規分布に従う乱数からなるシ
データです。乱数発生後に、さらに最初の10%分についてG1群に相当するところのみ数値
(つまり10%がG1群で高発現というシミュレーションデータを作成している)

```
in_f <- "sample23.txt"                  #入力ファイル名を指定してin_fに
out_f <- "hoge4.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 3                           #G1群のサンプル数を指定
```

R Console

```
> #以下は(こんなこともできますという)おまけ
> #(G1群 vs. G2群) t-testでp-value < 0.05を満たす遺伝子数を表示さ$
> param4 <- 0.05                                    #閾値を指定
> sum(p.value < param4)                             #p.valueが(param4)未満と$
[1] 1226
> sum(q.value < 0.05)                               #FDR < 0.05を満たす要素数$
[1] 0
> sum(q.value < 0.10)                               #FDR < 0.10を満たす要素数$
[1] 0
> sum(q.value < 0.15)                               #FDR < 0.15を満たす要素数$
[1] 110
> sum(q.value < 0.20)                               #FDR < 0.20を満たす要素数$
[1] 388
> sum(q.value < 0.25)                               #FDR < 0.25を満たす要素数$
[1] 627
>
```

①

②

Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

• 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) 教科書p167-

発現変動解析(limma)

limmaという発現変動解析用Rパッケージを用いてDEG検出を行います。

- 解析 | 発現変動 | 2群間 | 発現変動遺伝子の割合を調べる (Ploner_2006) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | について (last modified 2011/08/02)
- 解析 | 発現変動 | 2群間 | 対応なし | WAD (Kadota_2008) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | Random forest (Diaz-Uriarte_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | shrinkage t (Opgen-Rhein_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | layer ranking algorithm (Chen_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | fdr2d (Ploner_2006) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | Rank products (Breitling_2004) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | samroc (Broberg_2003) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | SAM (Tusher_2001) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | Studet
- 解析 | 発現変動 | 2群間 | 対応なし | Welch
- 解析 | 発現変動 | 2群間 | 対応なし | Mann-
- 解析 | 発現変動 | 2群間 | 対応なし | パター
- 解析 | 発現変動 | 2群間 | 対応あり | につい
- 解析 | 発現変動 | 2群間 | 対応あり | SAM
- 解析 | 発現変動 | 2群間 | 対応あり | SAM
- 解析 | 発現変動 | 2群間 | 対応あり | 時系列

解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004)

limmaパッケージを用いて2群間比較を行うやり方を示します。
この方法は経験ベイズと表現されたり、moderated t statisticと表現されたりしているようです。
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し、以下をコピペ

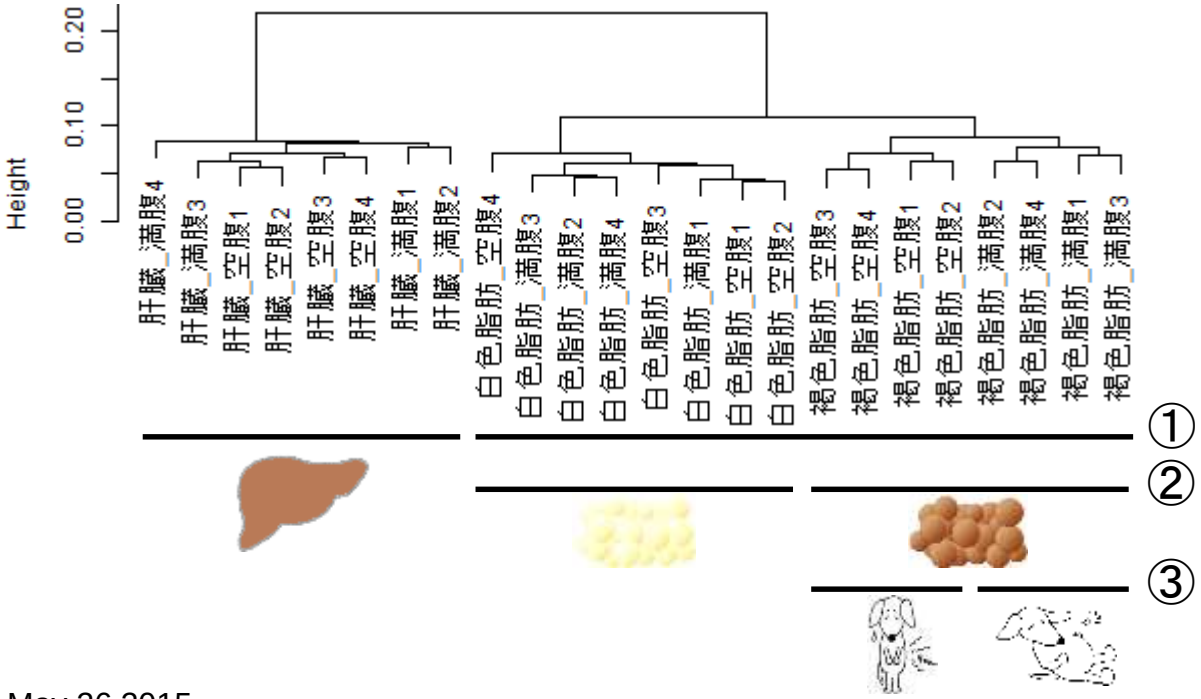
1. サンプルデータ20の31,099 probesets×8 samplesのdata_rma_2_LIV.txt(G1群4サンプル vs. G2群4サンプル)の場合:

```
in_f <- "data_rma_2_LIV.txt"            #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 4                           #G1群のサンプル数を指定
param_G2 <- 4                           #G2群のサンプル数を指定

#必要なパッケージをロード
library(limma)                          #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したフ
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.clを

#本番
#design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignlに格納
design <- model.matrix(~data.cl)        #デザイン行列を作成した結果をdesignlに格納
```

発現変動解析(limma)

GSE7623データを用い、様々な2群間比較を行い、クラスタリング結果とDEG検出結果の関係をみてみよう

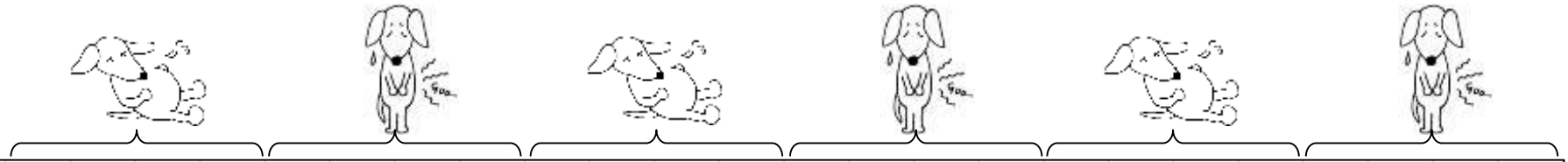
- Nakai et al., Biosci Biotechnol Biochem., 72: 139–148, 2008
  - GSE7623、GPL1355 (Affymetrix Rat Genome 230 2.0 Array)、31,099 probesets
  - ラット24サンプル：Brown adipose tissue (褐色脂肪組織; BAT) 8サンプル、White adipose tissue (白色脂肪組織; WAT) 8サンプル、Liver (肝臓; LIV) 8サンプル
    - BAT 8サンプル：通常(BAT_fed) 4サンプル　対　24時間絶食(BAT_fas) 4サンプル
    - WAT 8サンプル：通常(WAT_fed) 4サンプル　対　24時間絶食(WAT_fas) 4サンプル
    - LIV 8サンプル：通常(LIV_fed) 4サンプル　対　24時間絶食(LIV_fas) 4サンプル



rcode_clustering_png.txtの実行結果。
①肝臓と脂肪間で大きく二つのクラスターに分かれている。
②脂肪の中でも白色脂肪と褐色脂肪に分かれている。
③褐色脂肪は空腹(24時間絶食)と満腹(通常)できれいに分かれている。

発現変動解析(limma)

解析1の予想：DEGなし。解析2~4の予想：DEGあり。予想されるDEG数：解析2 < 解析3 < 解析4



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2	WAT_fed3	WAT_fed4	WAT_fas1	WAT_fas2	WAT_fas3	WAT_fas4	LIV_fed1	LIV_fed2	LIV_fed3	LIV_fed4	LIV_fas1	LIV_fas2	LIV_fas3	LIV_fas4
1367452_at																								
1367453_at																								
1367454_at																								
1367455_at																								
1367456_at																								
1367457_at																								
1367458_at																								
1367459_at																								
1367460_at																								
1367461_at																								
...																								
解析1	G1	G1	G2	G2																				
解析2	G1	G1			G2	G2																		
解析3	G1	G1							G2	G2														
解析4	G1	G1															G2	G2						

発現変動解析(limma)

rcode_limma_basic.txt。解析1用。テンプレートからの変更点および追加点。Macのヒトは区切り文字に注意

1. サンプルデータ20の31,099 probesets×8 s

```
in_f <- "data_rma_2_LIV.txt"
out_f <- "hoge1.txt"
param_G1 <- 4
param_G2 <- 4

#必要なパッケージをロード
library(limma)

#入力ファイルの読み込みとラベル情報
data <- read.table(in_f, header
data.cl <- c(rep(1, param_G1),

#本番
#design <- model.matrix(~ as.fa
design <- model.matrix(~data.cl
fit <- lmFit(data, design)
out <- eBayes(fit)
p.value <- out$p.value[,ncol(de
q.value <- p.adjust(p.value, me
ranking <- rank(p.value)
sum(q.value < 0.05)
```

```
############################
###  解析1 (Analysis1)  ###
############################
in_f <- "data_mas_EN.txt"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"               #出力ファイル名を指定してout_fに格納
param_G1 <- 2                      #G1群のサンプル数を指定
param_G2 <- 2                      #G2群のサンプル数を指定

#必要なパッケージをロード
library(limma)                     #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定した
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.cl

#サブセットの作成（解析したいデータのみにする）
posi <- c(1,2,3,4)                 #元の発現行列上での列番号を指定
data <- data[,posi]                #サブセットを抽出

#本番
#design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignに格納
design <- model.matrix(~data.cl)   #デザイン行列を作成した結果をdesignに格納
fit <- lmFit(data, design)         #モデル構築(ばらつきの程度を見積もっている)
out <- eBayes(fit)                 #検定(経験ベイズ)
p.value <- out$p.value[,ncol(design)]  #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)           #p.valueでランキングした結果をrankingに格納
sum(q.value < 0.05)                #FDR < 0.05を満たす遺伝子数を表示

#ファイルに保存
tmp <- cbind(rownames(data), data, p.value, q.value, ranking)#入力データの右側にp.valu
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```

rcode_limma_basic.txt

入力ファイル（data_mas_EN.txt）
読み込み後のdataオブジェクト
は24サンプルからなる

```
############################
###  解析1 (Analysis1)  ###
############################
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param_G1 <- 2                            #G1群のサンプル数を指定
param_G2 <- 2                            #G2群のサンプル数を指定

#必要なパッケージをロード
library(limma)                           #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t"
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2

#サブセットの作成（解析したいデータのみにする）
posi <- c(1,2,3,4)                       #元の発現行列上での列
data <- data[,posi]                      #サブセットを抽出

#本番
#design <- model.matrix(~ as.factor(data.cl))#デザイン行列を
design <- model.matrix(~data.cl)         #デザイン行列を作成し
fit <- lmFit(data, design)               #モデル構築(ばらつき
out <- eBayes(fit)                       #検定(経験ベイズ)
p.value <- out$p.value[,ncol(design)]    #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)                 #p.valueでランキング
sum(q.value < 0.05)                      #FDR < 0.05を満たす遺

#ファイルに保存
tmp <- cbind(rownames(data), data, p.value, q.value, rankin
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.nam
```

R Console

```
> ############################
> ###  解析1 (Analysis1)  ###
> ############################
> in_f <- "data_mas_EN.txt"                #入力ファイ$
> out_f <- "hoge1.txt"                     #出力ファイ$
> param_G1 <- 2                            #G1群のサン$
> param_G2 <- 2                            #G2群のサン$
>
> #必要なパッケージをロード
> library(limma)                           #パッケージ$
>
> #入力ファイルの読み込みとラベル情報の作成
> data <- read.table(in_f, header=TRUE, row.names=1,$
> data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G$
> dim(data)
[1] 31099    24
> colnames(data)
 [1] "BAT_fed1" "BAT_fed2" "BAT_fed3" "BAT_fed4"
 [5] "BAT_fas1" "BAT_fas2" "BAT_fas3" "BAT_fas4"
 [9] "WAT_fed1" "WAT_fed2" "WAT_fed3" "WAT_fed4"
[13] "WAT_fas1" "WAT_fas2" "WAT_fas3" "WAT_fas4"
[17] "LIV_fed1" "LIV_fed2" "LIV_fed3" "LIV_fed4"
[21] "LIV_fas1" "LIV_fas2" "LIV_fas3" "LIV_fas4"
>
```

rcode_limma_basic.txt

posiで指定した列番号のみからなるサブセットを抽出できていることがわかる

```
############################
###  解析1 (Analysis1)  ###
############################
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                     #出力ファイル名を指定してout_fに格納
param_G1 <- 2                            #G1群のサンプル数を指定
param_G2 <- 2                            #G2群のサンプル数を指定

#必要なパッケージをロード
library(limma)                           #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定した
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群

#サブセットの作成（解析したいデータのみにする）
posi <- c(1,2,3,4)                       #元の発現行列上
data <- data[,posi]                      #サブセットを抽

#本番
#design <- model.matrix(~ as.factor(data.cl))#デザイン
design <- model.matrix(~data.cl)         #デザイン行列を
fit <- lmFit(data, design)               #モデル構築(ば
out <- eBayes(fit)                       #検定(経験ベイ
p.value <- out$p.value[,ncol(design)]    #p値をp.valueに
q.value <- p.adjust(p.value, method="BH")#q値をq.value
ranking <- rank(p.value)                 #p.valueでラン
sum(q.value < 0.05)                      #FDR < 0.05を満

#ファイルに保存
tmp <- cbind(rownames(data), data, p.value, q.value, r
write.table(tmp, out_f, sep="¥t", append=F, quote=F, r
```

R Console

```
> #サブセットの作成(解析したいデータのみにする)
> posi <- c(1,2,3,4)                     #元の発現行$
> data <- data[,posi]                    #サブセット$
> dim(data)
[1] 31099     4
> colnames(data)
[1] "BAT_fed1" "BAT_fed2" "BAT_fed3" "BAT_fed4"
> head(data)
           BAT_fed1 BAT_fed2 BAT_fed3 BAT_fed4
1367452_at 12.78446 12.44708 12.80591 12.30472
1367453_at 11.80125 12.15293 11.94223 11.96848
1367454_at 11.38990 11.16076 11.14599 11.21209
1367455_at 12.36435 12.52974 12.43257 12.60401
1367456_at 13.44849 13.54305 13.55279 13.62980
1367457_at 10.40403 10.69632 10.47508 10.45579
> posi
[1] 1 2 3 4
>
```

発現変動解析(limma)

data.clで指定している情報は、グループラベル情報（どの列がどの群由来かということ）

	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2	3	4	1	2	3	4
1367452_at																
1367453_at																
1367454_at																
1367455_at																
1367458_at																
1367459_at																
1367460_at																
1367461_at																
...																
解析1	G1	G1	G2	G2												
解析2	G1	G1			G2	G2										
解析3	G1	G1							G2	G2						
解析4	G1	G1														

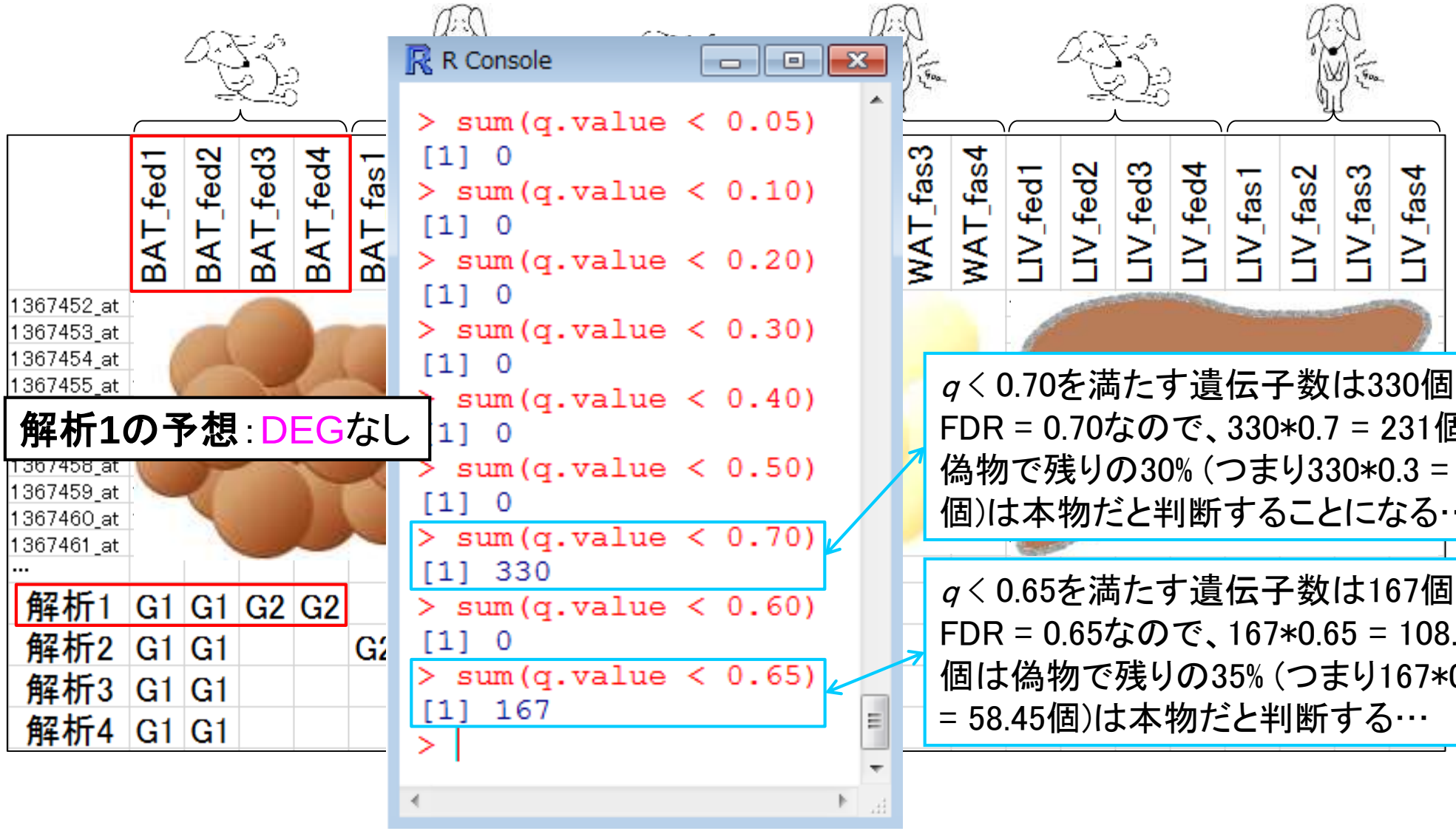
解析1の予想：DEGなし

R Console

```
> #サブセットの作成(解析したいデータのみにする)
> posi <- c(1,2,3,4)                    #元の発$
> data <- data[,posi]                   #サブセ$
> dim(data)
[1] 31099     4
> colnames(data)
[1] "BAT_fed1" "BAT_fed2" "BAT_fed3" "BAT_fed4"
> head(data)
           BAT_fed1 BAT_fed2 BAT_fed3 BAT_fed4
1367452_at 12.78446 12.44708 12.80591 12.30472
1367453_at 11.80125 12.15293 11.94223 11.96848
1367454_at 11.38990 11.16076 11.14599 11.21209
1367455_at 12.36435 12.52974 12.43257 12.60401
1367456_at 13.44849 13.54305 13.55279 13.62980
1367457_at 10.40403 10.69632 10.47508 10.45579
> posi
[1] 1 2 3 4
> data.cl
[1] 1 1 2 2
>
```

発現変動解析(limma)

通常（私）は、0.05～0.30あたりのFDR閾値を調査→DEGがないと判断



解析1の予想：DEGなし

q < 0.70を満たす遺伝子数は330個。FDR = 0.70なので、330*0.7 = 231個は偽物で残りの30%（つまり330*0.3 = 99個）は本物だと判断することになる…

q < 0.65を満たす遺伝子数は167個。FDR = 0.65なので、167*0.65 = 108.55個は偽物で残りの35%（つまり167*0.35 = 58.45個）は本物だと判断する…

rcode_limma_basic.txt

rcode_limma_basic.txtで動作確認をしてから、param_posiのように変更予定箇所を上のほうに移動して、解析2-4用のコードを作成する（のが門田流）

```
############################
###  解析1 (Analysis1)  ###
############################
in_f <- "data_mas_EN.txt"
out_f <- "hoge1.txt"
param_G1 <- 2
param_G2 <- 2

#必要なパッケージをロード
library(limma)

#入力ファイルの読み込みとラベル情報
data <- read.table(in_f, header=TR
data.cl <- c(rep(1, param_G1), rep

#サブセットの作成（解析したいデータ
posi <- c(1,2,3,4)
data <- data[,posi]

#本番
#design <- model.matrix(~ as.facto
design <- model.matrix(~data.cl)
fit <- lmFit(data, design)
out <- eBayes(fit)
p.value <- out$p.value[,ncol(desig
q.value <- p.adjust(p.value, metho
ranking <- rank(p.value)
sum(q.value < 0.05)

#ファイルに保存
tmp <- cbind(rownames(data), data,
write.table(tmp, out_f, sep="¥t",
```

rcode_limma_all.txt（の一部）

```
############################
###  解析1 (Analysis1)  ###
############################
in_f <- "data_mas_EN.txt"              #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                   #出力ファイル名を指定してout_fに格納
param_G1 <- 2                          #G1群のサンプル数を指定
param_G2 <- 2                          #G2群のサンプル数を指定
param_posi <- c(1,2,3,4)               #元の発現行列上での列番号を指定

#必要なパッケージをロード
library(limma)                         #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成、そしてサブセットの作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定した
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.c
data <- data[,param_posi]              #サブセットを抽出
colnames(data)                         #サブセット抽出後のサンプル名を表示

#本番
design <- model.matrix(~data.cl)       #デザイン行列を作成した結果をdesignに格納
fit <- lmFit(data, design)             #モデル構築(ばらつきの程度を見積もっている)
out <- eBayes(fit)                     #検定(経験ベイズ)
p.value <- out$p.value[,ncol(design)]  #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)               #p.valueでランキングした結果をrankingに格納
sum(q.value < 0.05)                    #FDR < 0.05を満たす遺伝子数を表示
sum(q.value < 0.10)                    #FDR < 0.10を満たす遺伝子数を表示
sum(q.value < 0.30)                    #FDR < 0.30を満たす遺伝子数を表示
sum(q.value < 0.50)                    #FDR < 0.50を満たす遺伝子数を表示

#ファイルに保存
tmp <- cbind(rownames(data), data, p.value, q.value, ranking)#入力データの右側にp.val
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```

解析2用のコード

発現変動解析（ ）

	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2
1367452_at						
1367453_at						
1367454_at						
1367455_at						
1367458_at						
1367459_at						
1367460_at						
1367461_at						
…						
解析1	G1	G1	G2	G2		
解析2	G1	G1			G2	G2
解析3	G1	G1				
解析4	G1	G1				

解析2の予想：DEGあり

rcode_limma_all.txt（の一部）

```
############################↓
###  解析2 (Analysis2)  ###↓
############################↓
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定してin_fに格納↓
out_f <- "hoge2.txt"                     #出力ファイル名を指定してout_fに格納↓
param_G1 <- 2                            #G1群のサンプル数を指定↓
param_G2 <- 2                            #G2群のサンプル数を指定↓
param_posi <- c(1,2,5,6)                 #元の発現行列上での列番号を指定↓
↓
#必要なパッケージをロード↓
library(limma)                           #パッケージの読み込み↓
↓
#入力ファイルの読み込みとラベル情報の作成、そしてサブセットの作成↓
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定し
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.
data <- data[,param_posi]                #サブセットを抽出↓
colnames(data)                           #サブセット抽出後のサンプル名を表示↓
↓
#本番↓
design <- model.matrix(~data.cl)         #デザイン行列を作成した結果をdesignに格納↓
fit <- lmFit(data, design)               #モデル構築(ばらつきの程度を見積もっている)↓
out <- eBayes(fit)                       #検定(経験ベイズ)↓
p.value <- out$p.value[,ncol(design)]    #p値をp.valueに格納↓
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納↓
ranking <- rank(p.value)                 #p.valueでランキングした結果をrankingに格納↓
sum(q.value < 0.05)                      #FDR < 0.05を満たす遺伝子数を表示↓
sum(q.value < 0.10)                      #FDR < 0.10を満たす遺伝子数を表示↓
sum(q.value < 0.30)                      #FDR < 0.30を満たす遺伝子数を表示↓
sum(q.value < 0.50)                      #FDR < 0.50を満たす遺伝子数を表示↓
↓
#ファイルに保存↓
tmp <- cbind(rownames(data), data, p.value, q.value, ranking)#入力データの右側にp.va
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定し
```

発現変動解析(limma)

通常（私）は、0.05～0.30あたりのFDR閾値を調査→DEGがあると判断




	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	LIV_fas4
1367452_at								
1367453_at								
1367454_at								
1367455_at								
1367458_at								
1367459_at								
1367460_at								
1367461_at								
…								
解析1	G1	G1	G2	G2				
解析2	G1	G1			G2	G2		
解析3	G1	G1						
解析4	G1	G1						

解析2の予想：DEGあり

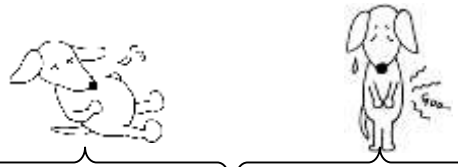
R Console

```
> colnames(data)                                    #サブ$
[1] "BAT_fed1" "BAT_fed2" "BAT_fas1" "BAT_fas2"
>
> #本番
> design <- model.matrix(~data.cl)                  #デザ$
> fit <- lmFit(data, design)                        #モデ$
> out <- eBayes(fit)                                #検定$
> p.value <- out$p.value
> q.value <- p.adjust(p
> ranking <- rank(p.valu
> sum(q.value < 0.05)
[1] 0
> sum(q.value < 0.10)
[1] 38
> sum(q.value < 0.30)                               #FDR $
[1] 2375
> sum(q.value < 0.50)                               #FDR $
[1] 8993
>
> #ファイルに保存
> tmp <- cbind(rownames(data), data, p.value, $
> write.table(tmp, out_f, sep="\t", append=F, $
>
```

q＜0.1を満たす遺伝子数は38個。FDR = 0.1なので、38*0.1 = 3.8個は偽物で残りの90%（つまり38*0.9 = 34.2個）は本物だと判断することになる…

発現変動解析(limma)

通常（私）は、0.05～0.30あたりの
FDR閾値を調査→DEGがあると判断



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	
1367452_at										
1367453_at										
1367454_at										
1367455_at										
…										
解析1	G1	G1	G2	G2						
解析2	G1	G1			G2	G2				
解析3	G1	G1							G2	G2
解析4	G1	G1								G2 G2

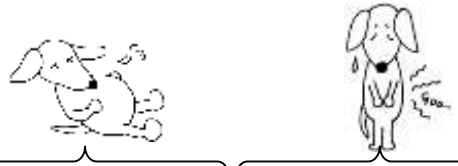
解析3の予想：DEGあり

R Console

```
> colnames(data)                                  #サブ$
[1] "BAT_fed1" "BAT_fed2" "WAT_fed1" "WAT_fed2"
>
> #本番
> design <- model.matrix(~data.cl)                #デザ$
> fit <- lmFit(data, design)                      #モデ$
> out <- eBayes(fit)                              #検定$
> p.value <- out$p.value[,ncol(design)]           #p値$
> q.value <- p.adjust(p.value, method="BH")#q$
> ranking <- rank(p.value)                        #p.va$
> sum(q.value < 0.05)                             #FDR $
[1] 0
> sum(q.value < 0.10)                             #FDR $
[1] 0
> sum(q.value < 0.30)                             #FDR $
[1] 4786
> sum(q.value < 0.50)                             #FDR $
[1] 12733
```

発現変動解析(limma)

通常（私）は、0.05～0.30あたりの
FDR閾値を調査→DEGがあると判断



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1									
1367452_at																		
1367453_at																		
1367454_at																		
1367455_at																		
1367458_at																		
1367459_at																		
1367460_at																		
1367461_at																		
…																		
解析1	G1	G1	G2	G2														
解析2	G1	G1			G2	G2												
解析3	G1	G1							G2	G2								
解析4	G1	G1													G2	G2		

解析4の予想：DEGあり

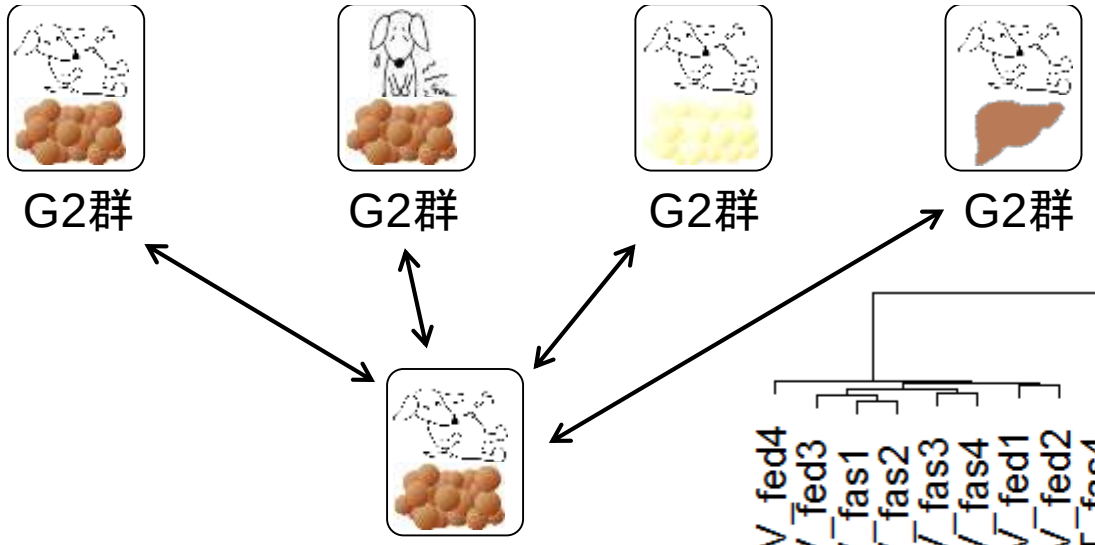
R Console

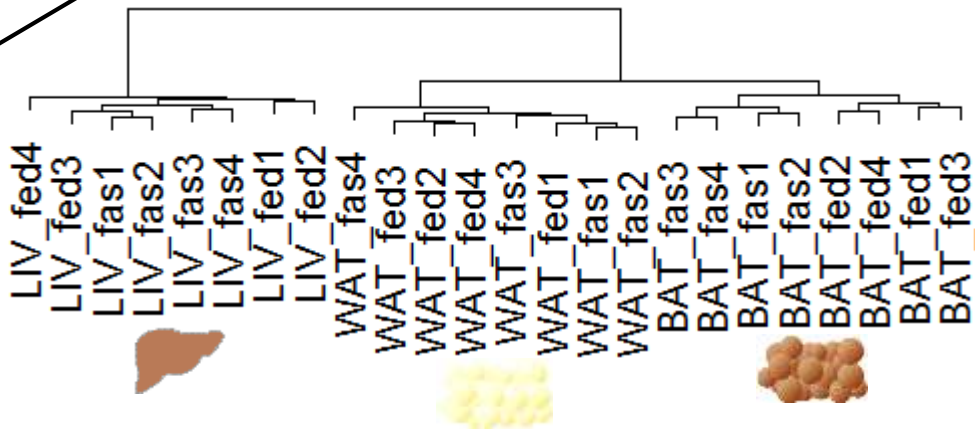
```
> colnames(data)                                  #サブ$
[1] "BAT_fed1" "BAT_fed2" "LIV_fed1" "LIV_fed2"
>
> #本番
> design <- model.matrix(~data.cl)                #デザ$
> fit <- lmFit(data, design)                      #モデ$
> out <- eBayes(fit)                              #検定$
> p.value <- out$p.value[,ncol(design)]           #p値$
> q.value <- p.adjust(p.value, method="BH")#q$
> ranking <- rank(p.value)                        #p.va$
> sum(q.value < 0.05)                             #FDR $
[1] 2892
> sum(q.value < 0.10)                             #FDR $
[1] 5829
> sum(q.value < 0.30)                             #FDR $
[1] 13771
> sum(q.value < 0.50)                             #FDR $
[1] 19355
```

DEG検出結果まとめ

解析1の予想：DEGなし。解析2~4の予想：DEGあり。予想されるDEG数：解析2 < 解析3 < 解析4

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR < 0.05	0	0	0	2892
FDR < 0.10	0	38	0	5829
FDR < 0.30	0	2375	4786	13771
FDR < 0.50	0	8993	12733	19355





Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

参考

4.と5.と6.は実質的に同じです

Tips：関数の利用法

解析 | 発現変動 | 2群間 | 対応なし | Student's t-test NEW

等分散性を仮定し

出力ファイルの「p.v
るものです。実用上
のに相当します。p
のと同じです。
「ファイル」−「ディレ

1. サンプルデータ1

クラスラベル情報

```
in_f1 <- "sam
in_f2 <- "sam
out_f <- "hog

#入力ファイル(
data <- read.
hoge <- read.
data.cl <- ho

#等分散性を仮定
```

4. サンプルデータ23のsample23.txtの場合:

最初の3サンプルがG1群、残りの
レーションデータです。乱数発生
数値を+3している(つまり10%が

```
in_f <- "sample23.txt"
out_f <- "hoge4.txt"
param_G1 <- 3
param_G2 <- 3

#入力ファイルの読み込みとラ
data <- read.table(in_f,
data.cl <- c(rep(1, para

#等分散性を仮定 (var.equal
Students_ttest <- functi
    x.class1 <- x[(cl ==
    x.class2 <- x[(cl ==
    if((sd(x.class1)+sd(x.class2))
        stat <- 0
        pval <- 1
        return(c(stat, pval))
    }
    else{
        hoge <- t.test(x.class1, x.
        return(c(hoge$statistic, ho
    }
```

5. サンプルデータ23のsample23.txtの場合:

4.と同じですが、関数の定義の仕方が異なります。

```
in_f <- "sample23.txt"                  #入力ファイル名を指定してin_fに格納
out_f <- "hoge5.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 3                           #G1群のサンプル数を指定
param_G2 <- 3                           #G2群のサンプル数を指定

#必要な関数などをロード
source("http://www.iu.a.u-tokyo.ac.jp/~kadota/R/R_functions.R")#Student'

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#i
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベ

#本番
out <- t
```

6. サンプルデータ23のsample23.txtの場合:

5.とほぼ同じですが、作業ディレクトリ中にStudents_ttest関数を含むR_functions.Rという名前のファイルが存在するという前提です。

```
in_f <- "sample23.txt"                  #入力ファイル名を指定してin_fに格納
out_f <- "hoge6.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 3                           #G1群のサンプル数を指定
param_G2 <- 3                           #G2群のサンプル数を指定

#必要な関数などをロード
source("R_functions.R")                 #Student's t-testを行うStudents_tt

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#i
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベ

#本番
out <- t(apply(data, 1, Students_ttest, data.cl))#各（行）遺伝子についてt検
```

参考

ネット接続環境であれば、ここで提供している関数を利用可能

Tips：関数の利用法

5. サンブルデータ23のsample23.txtの場合:

4.と同じですが、関数の定義の仕方が異なります。

```
in_f <- "sample23.txt"                  #入力ファイル名を指定してin_fに格納
out_f <- "hoge5.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 3                           #G1群のサンプル数を指定
param_G2 <- 3                           #G2群のサンプル数を指定

#必要な関数などをロード
source("http://www.iu.a.u-tokyo.ac.jp/~kadota/R/R_functions.R")#Student'

#入力ファイルの読み込みとラベル情報の作成
data <
data.c

#本番
out <-
```

http://www.iu.a.u-tokyo.ac.jp/~kadota/R/R_functions.R

iu.a.u-tokyo.ac.jp

```
############################
###  Student's t-test    ###
############################
Students_ttest <- function(x, cl){
    x.class1 <- x[(cl == 1)]                    #ラベルが1のものをx.class1に格納
    x.class2 <- x[(cl == 2)]                    #ラベルが2のものをx.class2に格納
    if((sd(x.class1)+sd(x.class2)) == 0){       #両方の群の標準偏差が共に0の場合は計算できないので...
        stat <- 0                               #統計量を0
        pval <- 1                               #p値を1
        return(c(stat, pval))                   #として結果を結果として返す
    }
    else{                                       #A, Bどちらかの群の標準偏差が0(上記条件以外)の場合は
        hoge <- t.test(x.class1, x.class2, var.equal=T)  #t検定を行って、
        return(c(hoge$statistic, hoge$p.value))  #統計量とp値を結果として返す
    }
}
```

参考

Tips：関数の利用法

ネット接続環境でなくても、一旦R_functions.Rファイルを作業ディレクトリにダウンロードしておけばStudents_ttest関数を利用可能

6. サンプルデータ23のsample23.txtの場合:

5.とほぼ同じですが、作業ディレクトリ中にStudents_ttest関数を含むR_functions.Rという名前のファイルが存在するという前提です。

```
in_f <- "sample23.txt"                  #入力ファイル名を指定してin_fに格納
out_f <- "hoge6.txt"                    #出力ファイル名を指定してout_fに格納
param_G1 <- 3                           #G1群のサンプル数を指定
param_G2 <- 3                           #G2群のサンプル数を指定

#必要な関数などをロード
source("R_functions.R")                 #Student's t-testを行うStudents_tt

#入力ファイル
data <- read
data.cl <- c

#本番
out <- t(app
```

C:¥Users¥kadota¥Desktop¥hoge¥R_functions.R - EmEditor

ファイル(F) 編集(E) 検索(S) 表示(V) ツール(T) ウィンドウ(W) ヘルプ(H)

R_functions.R

```
################################↓
###  Student's t-test     ###↓
################################↓
Students_ttest <- function(x, cl){↓
    x.class1 <- x[(cl == 1)]                    #ラベルが1のものをx.class1に格納↓
    x.class2 <- x[(cl == 2)]                    #ラベルが2のものをx.class2に格納↓
    if((sd(x.class1)+sd(x.class2)) == 0){       #両方の群の標準偏差が共に0の場合は計
        stat <- 0                               #統計量を0↓
        pval <- 1                               #p値を1↓
        return(c(stat, pval))                   #として結果を結果として返す↓
    }↓
    else{                                       #A, Bどちらかの群の標準偏差が0(上記条
        hoge <- t.test(x.class1, x.class2, var.equal=T)   #t検定を行って↓
```

• 解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006)

参考

多数の方法があります

limma以外にも様々なパッケージがあります。ウェブページでリストアップされていない方法(例：右下のFCROS)も多数あり。ここではlimmaと似た系列のIBMT法を紹介。

- 解析 | クラスタリング | 非階層的 | 主成分分析(PCA) (last modified 2012/04/13)
- 解析 | 発現変動 | 2群間 | 発現変動遺伝子の割合を調べる (Ploner_2006) (last modified 2013
- 解析 | 発現変動 | 2群間 | 対応なし | について (last modified 2011/08/02)
- 解析 | 発現変動 | 2群間 | 対応なし | WAD (Kadota_2008) (last modified 2015/03/30)
- 解析 | 発現変動 | 2群間 | 対応なし | Random forest (Diaz-Uriarte_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | shrinkage t (Opgen-Rhein_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | layer ranking algorithm (Chen_2007) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | fdr2d (Ploner_2006) (l modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006) modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | Rank products (Breitli 2004) (last modified 2015/02/12)
- 解析 | 発現変動 | 2群間 | 対応なし | empirical Baye (Smyth 2004) (last modified 2015/05/24) NEW
- 解析 | 発現変動 | 2群間 | 対応なし | samroc (Brober
- 解析 | 発現変動 | 2群間 | 対応なし | SAM (Tusher_2
- 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test(
- 解析 | 発現変動 | 2群間 | 対応なし | Welch t-test(las
- 解析 | 発現変動 | 2群間 | 対応なし | Mann-Whitney
- 解析 | 発現変動 | 2群間 | 対応なし | パターンマッチ
- 解析 | 発現変動 | 2群間 | 対応あり | について(last m
- 解析 | 発現変動 | 2群間 | 対応あり | SAM (Tusher_2
- 解析 | 発現変動 | 2群間 | 対応あり | SAM (Tusher_2
- 解析 | 発現変動 | 2群間 | 対応あり | 時系列 | について
- 解析 | 発現変動 | 2群間 | 対応あり | 時系列 | maSig
- 解析 | 発現変動 | 3群間 | 対応なし | について(last m
- 解析 | 発現変動 | 3群間 | 対応なし | Mulcom (Isella
- 解析 | 発現変動 | 3群間 | 対応なし | limma (Smyth
- 解析 | 発現変動 | 3群間 | 対応なし | 一元配置分散
- 解析 | 発現変動 | 3群間 | 対応なし | Kruskal-Wallis

BMC Bioinformatics. 2014 Jan 15;15:14. doi: 10.1186/1471-2105-15-14.

Fold change rank ordering statistics: a new method for detecting differentially expressed genes.

Dembélé D[1], Kastner P.

Author information

Abstract

BACKGROUND: Different methods have been proposed for analyzing differentially expressed (DE) genes in microarray data. Methods based on statistical tests that incorporate expression level variability are used more commonly than those based on fold change (FC). However, FC based results are more reproducible and biologically relevant.

RESULTS: We propose a new method based on fold change rank ordering statistics (FCROS). We exploit the variation in calculated FC levels using combinatorial pairs of biological conditions in the datasets. A statistic is associated with the ranks of the FC values for each gene, and the resulting probability is used to identify the DE genes within an error level. The FCROS method is deterministic, requires a low computational runtime and also solves the problem of multiple tests which usually arises with microarray datasets.

CONCLUSION: We compared the performance of FCROS with those of other methods using synthetic and real microarray datasets. We found that FCROS is well suited for DE gene identification from noisy datasets when compared with existing FC based methods.

参考

多数の方法があります

IBMT法は、limmaパッケージ中の関数を内部的に用いています。limmaを基本としつつ、改良を加えた関数部分のみ提供しているという解釈でもよい。

http://eh3.uc.edu/r/ibmtR.R

eh3.uc.edu

```
IBMT<-function(mdata,testcol)
{ ##########################################################################
# Function for IBMT (Intensity-based Moderated T-statistic) # Written by: Maureen Sartor,
University of Cincinnati, 2006
##########################################################################
## This function adjusts the T-statistics and p-v
microarrays. The method contains elements simil
function in limma and to the Cyber-T ## program
hierarchical Bayesian method. ## Local regressio
to ## determine the prior degrees of freedom and
variance for each gene dependent on average sp
T-statistic uses a weighted average of prior and
degrees of freedom are simply the sum of ## pric
Please acknowledge your use of IBMT in publicat
Tomlinson CR, Wesselkamper SC, Sivaganesan S
Intensity-based hierarchical Bayes method impro
genes in microarray experiments. BMC Bioinform
mdata and testcol ## "mdata" should be a list ob
                                    named sigma, A
                                    integer or vect
                                    function is to b
                                    nput), with the
                                    alue for IBMT #
                                    variance for IBM
for IBMT ## ## Example Function Call: ## IBMT.r
further help on implementing function, contact sa
##########################################################################
library("stats") library("limma") logVAR<-log(mda
numgenes<-length(logVAR[df>0]) df[df==0]<-NA
egpred<-loessFit(eg,mdata$Amean,iterations=1,sp
trigamma(df/2) print("Local regression fit") mean
priordf<-vector(); testd0<-vector() for (i in 1:(nur
abs(mean.myfct-trigamma(testd0[i]/2)) if(i>2) { i
```

IBMT法の実体であるIBMT関数の読み込みを行っている

解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006)

IBMT法 (Sartor et al., 2006)の方法を用いて2群間で発現の異なる遺伝子をランキング。empirical Bayes (Smyth_2004)の改良版という位置づけですね。a novel Bayesian moderated-Tと書いてますし。

「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し、以下をコピペ

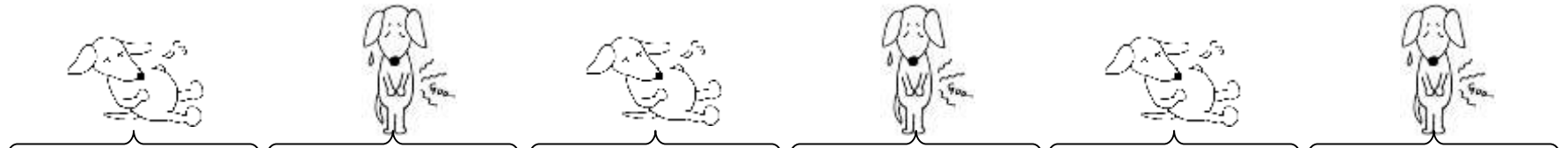
1. サンプルデータ20の31,099 probesets×8 samplesのdata_rma_2_LIV.txt(G1群4サンプル vs. G2群4サンプル)の場合:

```
in_f <- "data_rma_2_LIV.txt"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                  #出力ファイル名を指定してout_fに格納
param_G1 <- 4                         #G1群のサンプル数を指定
param_G2 <- 4                         #G2群のサンプル数を指定

#必要なパッケージなどをロード
library(limma)                        #パッケージの読み込み
source("http://eh3.uc.edu/r/ibmtR.R") #IBMTのRスクリプトの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.

#本番
#design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignに格
design <- model.matrix(~data.cl)      #デザイン行列を作成した結果をdesignに格納
fit <- lmFit(data, design)            #モデル構築(ばらつきの程度を見積もっている)
fit$Amean<-rowMeans(data)             #おまじない
fit <- IBMT(fit,2)                    #IBMTプログラムの実行
p.value <- fit$IBMT.p                 #p値をp.valueに格納
```

参考

発現変動解析(IBMT)

IBMT法はlimma系なので、全体的な傾向は変わりません。そして違いの程度を把握してもらうのが目的です。



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2	WAT_fed3	WAT_fed4	WAT_fas1	WAT_fas2	WAT_fas3	WAT_fas4	LIV_fed1	LIV_fed2	LIV_fed3	LIV_fed4	LIV_fas1	LIV_fas2	LIV_fas3	LIV_fas4
1367452_at																								
1367453_at																								
1367454_at																								
1367455_at																								
1367456_at																								
1367457_at																								
1367458_at																								
1367459_at																								
1367460_at																								
1367461_at																								
...																								
解析1	G1	G1	G2	G2																				
解析2	G1	G1			G2	G2																		
解析3	G1	G1							G2	G2														
解析4	G1	G1															G2	G2						

解析1の予想：DEGなし
解析2~4の予想：DEGあり
予想されるDEG数：解析2 < 解析3 < 解析4

参考

発現変動解析(IBMT)

rcode_ibmt_basic.txt。解析1用。テンプレートからの変更点および追加点。Macのヒトは区切り文字に注意

解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006)

IBMT法 (Sartor et al., 2006)の方法を用いて2群
(Smyth 2004)の改良版という位置づけですね。
「ファイル」-「ディレクトリの変更」で解析したい

**1. サンプルデータ20の31,099 probesets×8 sam
ル)の場合:**

```
in_f <- "data_rma_2_LIV.txt"
out_f <- "hoge1.txt"
param_G1 <- 4
param_G2 <- 4

#必要なパッケージなどをロード
library(limma)
source("http://eh3.uc.edu/r/ibmtR

#入力ファイルの読み込みとラベル情報の
data <- read.table(in_f, header=TR
data.cl <- c(rep(1, param_G1), rep

#本番
#design <- model.matrix(~ as.facto
design <- model.matrix(~data.cl)
fit <- lmFit(data, design)
fit$Amean<-rowMeans(data)
fit <- IBMT(fit,2)
p.value <- fit$IBMT.p
```

```
##############################
###  解析1 (Analysis1)  ###
##############################
in_f <- "data_mas_EN.txt"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"               #出力ファイル名を指定してout_fに格納
param_G1 <- 2                      #G1群のサンプル数を指定
param_G2 <- 2                      #G2群のサンプル数を指定

#必要なパッケージをロード
library(limma)                     #パッケージの読み込み
source("ibmtR.R")                  #IBMTのRスクリプトの読み込み

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdat

#サブセットの作成（解析したいデータのみにする）
posi <- c(1,2,3,4)                 #元の発現行列上での列番号を指定
data <- data[,posi]                #サブセットを抽出

#本番
design <- model.matrix(~data.cl)   #デザイン行列を作成した結果をdesignに格納
fit <- lmFit(data, design)         #モデル構築(ばらつきの程度を見積もっている)
fit$Amean<-rowMeans(data)          #おまじない
fit <- IBMT(fit,2)                 #IBMTプログラムの実行
p.value <- fit$IBMT.p              #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)           #p.valueでランキングした結果をrankingに格納
sum(q.value < 0.05)                #FDR < 0.05を満たす遺伝子数を表示
```

解析結果(IBMT)

rcode_ibmt_all.txtの結果。同じDEG検出法でも入力データ（前処理法：MAS5, RMA, RobLoxBioC）が異なると結果も異なることが分かる。

MAS5

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR＜0.05	0	1927	1999	7256
FDR＜0.10	0	2891	3246	9227
FDR＜0.30	0	6729	8030	14607
FDR＜0.50	0	11491	13602	19125

RMA

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR＜0.05	0	2889	2988	8965
FDR＜0.10	0	4348	5570	10954
FDR＜0.30	0	8973	14364	16059
FDR＜0.50	0	13927	20056	20305

RobLoxBioC

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR＜0.05	0	3066	2169	9196
FDR＜0.10	0	4527	4079	11434
FDR＜0.30	0	10008	11627	17265
FDR＜0.50	0	15485	17558	21451

課題：limmaとIBMTの解析結果を考察せよ

課題：解析結果(limma)

MAS5

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR < 0.05	0	0	0	2892
FDR < 0.10	0	38	0	5829
FDR < 0.30	0	2375	4786	13771
FDR < 0.50	0	8993	12733	19355

RMA

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR < 0.05	0	2451	2514	8783
FDR < 0.10	0	3996	5025	10730
FDR < 0.30	0	8937	13271	15734
FDR < 0.50	0	13646	19417	19728

RobLoxBioC

遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR < 0.05	0	1832	1025	8438
FDR < 0.10	0	3654	3234	10869
FDR < 0.30	0	9618	11097	16288
FDR < 0.50	0	14731	16576	20145

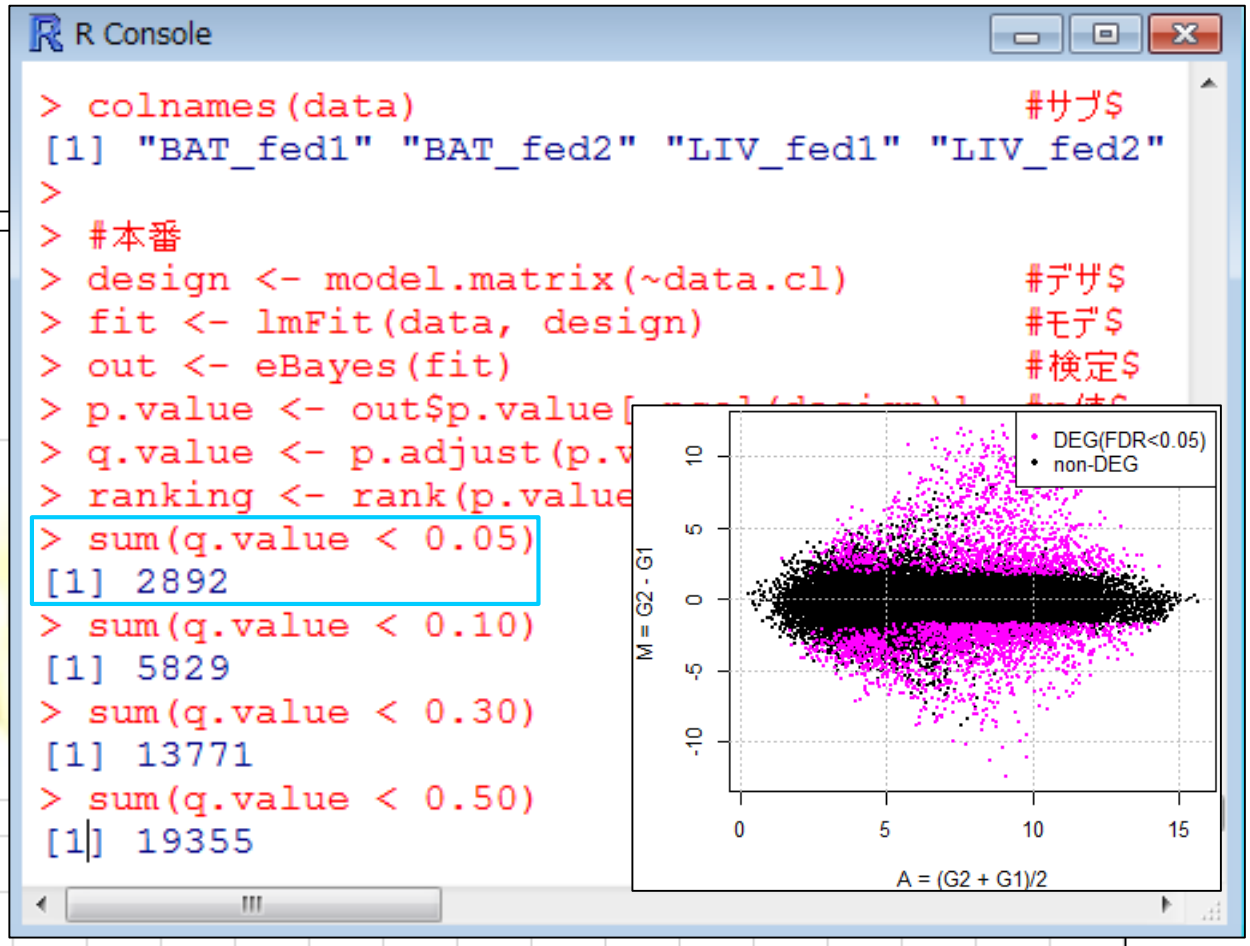
Contents

- 実験デザイン（教科書の§3.2.2）
- 2群間比較：発現変動遺伝子（DEG）検出
  - パターンマッチング法（相関係数の利用）
    - コードの中身をおさらい、apply関数の基本的な利用法など
  - 多重比較問題とFalse Discovery Rate (FDR)
    - 正規分布乱数由来のDEGが存在しないデータでStudent's t-test
    - 10% DEGが存在する正規乱数でデータ（10,000個中1,000個がDEG）でStudent's t-test
  - 発現変動解析用Rパッケージの利用（§4.2.1, p167-）
    - limmaパッケージ (Smyth GK, *SAGMB*, 2004)
    - 関数の利用法
    - IBMT法 (Sartor et al., *BMC Bioinformatics*, 2006)
    - 課題
  - 描画（M-A plot）
    - 作成法
    - 同一群内のばらつき（前処理法間の違い）

limma解析結果（解析4のFDRが0.05を満たす2,892 probesets）のM-A plotを描画しよう

M-A plot

	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	
1367452_at										
1367453_at										
1367454_at										
1367455_at										
1367456_at										
1367457_at										
1367458_at										
1367459_at										
1367460_at										
1367461_at										
...										
解析1	G1	G1	G2	G2						
解析2	G1	G1			G2	G2				
解析3	G1	G1							G2	G2
解析4	G1	G1								G2 G2



M-A plot

rcode_limma_MAplot_basic.txtは、②の例題7をテンプレートにしています。最終形は右下の図ですが、順を追って説明していきます。

- 解析 | 発現変動 | 2群間 | 対応なし | IBMT (Sartor_2006) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | Rank products (Breitling_2004) (last modified 2013/06/02)
- 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | samroc (Broberg_2003) (last modified 2014/02/03)
- 解析 | 発現変動 | 2群間 | 対応なし | SAM (Tusher_2001) (last modified 2014/02/03)

①

解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004)

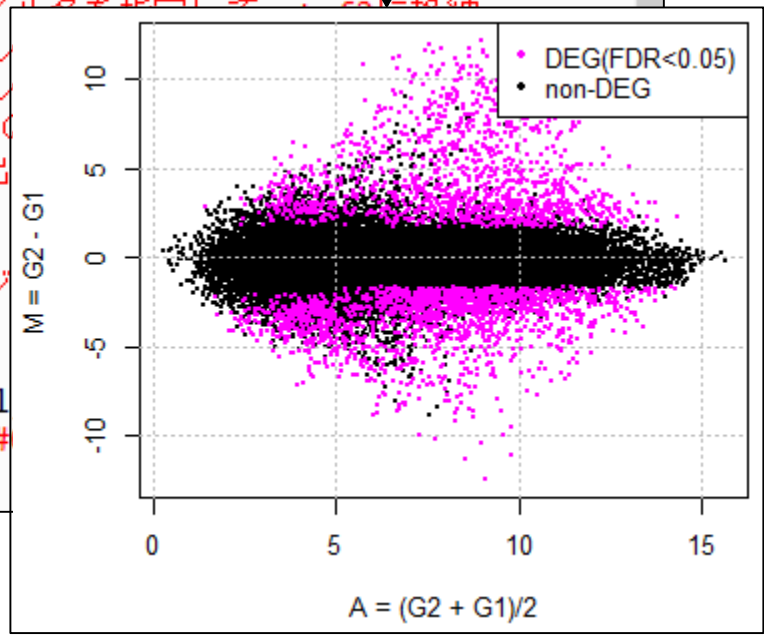
②

limmaパッ
この方法
「ファイル」

7. サンプルデータ20の31,099 probesets×8 samplesのdata_rma_2_LIV.txt(G1群4サンプル vs. G2群4サンプル)の場合:

M-A plotのpngファイルを生成しています。limmaパッケージ中のtopTable関数やplotMA関数を使わないやり方です。テキストファイルのほうは、M-A plotのM値とA値も出力させるようにしています。

```
in_f <- "data_rma_2_LIV.txt"                #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge7.txt"                       #出力ファイル名を指定してout_f1に格納
out_f2 <- "hoge7.png"                       #出力ファイ
param_G1 <- 4                               #G1群のサン
param_G2 <- 4                               #G2群のサン
param_FDR <- 0.05                           #DEG検出時
param_fig <- c(400, 380)                    #ファイル出

#必要なパッケージをロード
library(limma)                              #パッケージ

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#
```

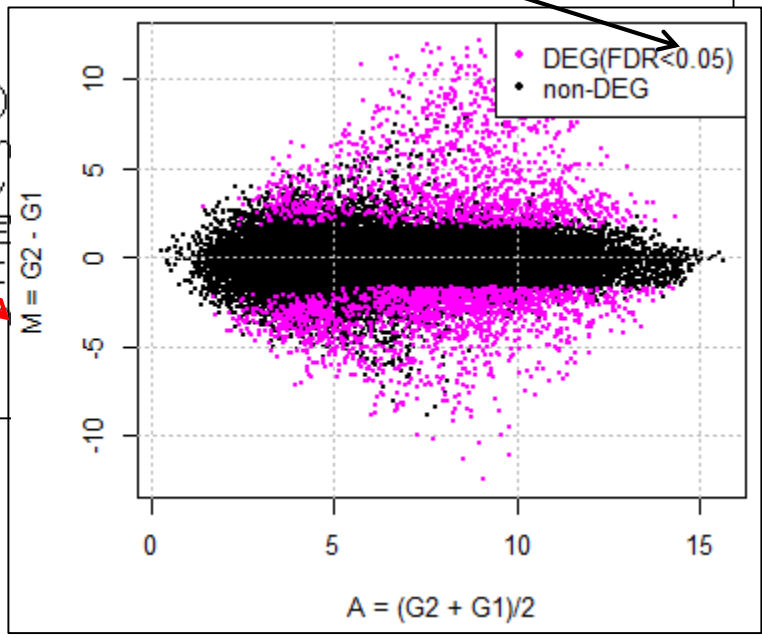


M-A plotの説明。横軸のAは平均発現レベル、縦軸のMは$\log_2$(G2/G1)に相当

```
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定してin_fに
out_f1 <- "hoge1.txt"                    #出力ファイル名を指定してout_f
out_f2 <- "hoge1.png"                    #出力ファイル名を指定してout_f2
param_G1 <- 2                            #G1群のサンプル数を指定
param_G2 <- 2                            #G2群のサンプル数を指定
param_posi <- c(1,2,17,18)               #元の発現行列上での列番号を指定
param_FDR <- 0.05                        #DEG検出時のfalse discovery rate (FDR)閾値を指定
param_fig <- c(400, 380)                 #ファイル出力時の横幅と縦幅を指定(単位はピクセル)

#必要なパッケージをロード
library(

#入力ファ
data <- r
data.cl <
data <- d
colnames(

#本番(DE
design <-
fit <- lm
out <- eB
p.value <
```

```
mean_G1 <- apply(as.matrix(data[,data.cl==1]), 1, mean)#遺伝子ごとにG1群の平均を計算した結果をmean_G1に格納
mean_G2 <- apply(as.matrix(data[,data.cl==2]), 1, mean)#遺伝子ごとにG2群の平均を計算した結果をmean_G2に格納
M <- mean_G2 - mean_G1                   #M-A plotのM値(y軸の値)に相当するものをMに格納
A <- (mean_G1 + mean_G2)/2               #M-A plotのA値(x軸の値)に相当するものをAに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(data), data, M, A, p.value, q.value, ranking)#入力データの右側にDEG検出結果を結合した
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保存

#ファイルに保存(M-A plot)
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch
grid(col="gray", lty="dotted")           #指定したパラメータでグリッ
```

```
R Console
> getwd()
[1] "C:/Users/kadota/Desktop/hoge/GSE7623"
> list.files(pattern="mas_EN")
[1] "data_mas_EN.txt"
>
```

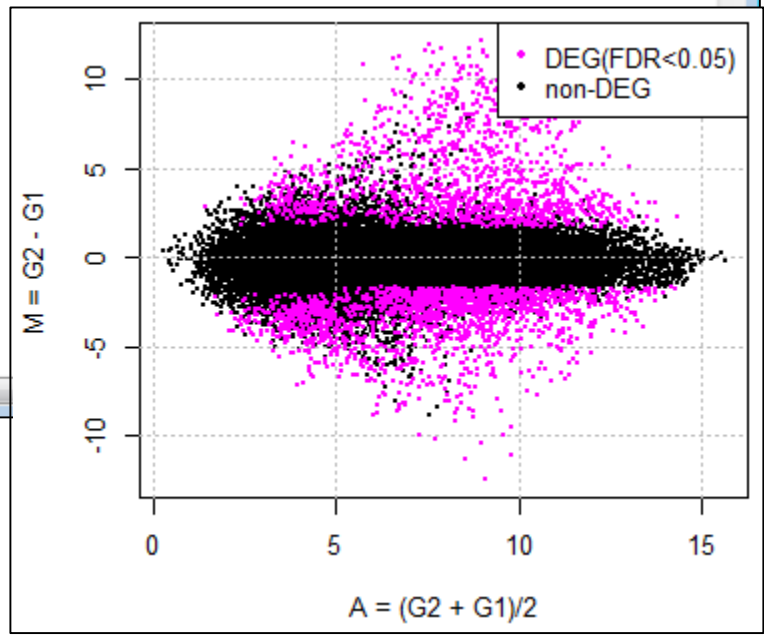


mean_G1とmean_G2は、単にグループごとの平均値を算出しているだけ

```
mean_G1 <- apply(as.matrix(data[,data.cl==1]), 1, mean)#遺伝子ごとにG1群の平均
mean_G2 <- apply(as.matrix(data[,data.cl==2]), 1, mean)#遺伝子ごとにG2群の平均を計算した結果をmean_G2に格納
M <- mean_G2 - mean_G1                    #M-A plotのM値(y軸の値)に相当するものをMに格納
A <- (mean_G1 + mean_G2)/2                #M-A plotのA値(x軸の値)に相当するものをAに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(data), data, M, A, p.value, q.value, ranking)#入力データの右側にDEG検出結果を結合した
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保存
```

R Console

```
> head(tmp)
          rownames(data) BAT_fed1 BAT_fed2 LIV_fed1 LIV_fed2         M        A   p.value   q.value ranking
1367452_at     1367452_at 12.78446 12.44708 12.19593 11.95968 -0.5379634 12.34679 0.1613769 0.3365743   14911
1367453_at     1367453_at 11.80125 12.15293 11.49419 11.49189 -0.4840521 11.73506 0.1846049 0.3647413   15740
1367454_at     1367454_at 11.38990 11.16076 11.66812 12.23333  0.6753955 11.61303 0.1322055 0.2991158   13745
1367455_at     1367455_at 12.36435 12.52974 12.80589 12.96296  0.4373753 12.66573 0.1991063 0.3812952   16239
1367456_at     1367456_at 13.44849 13.54305 13.38086 13.58722 -0.0117270 13.48990 0.9686717 0.9832814   30636
1367457_at     1367457_at 10.40403 10.69632 10.78228 10.61002  0.1459767 10.62316 0.6482376 0.7815199   25794
> head(cbind(mean_G1, mean_G2))
           mean_G1  mean_G2
1367452_at 12.61577 12.07781
1367453_at 11.97709 11.49304
1367454_at 11.27533 11.95072
1367455_at 12.44705 12.88442
1367456_at 13.49577 13.48404
1367457_at 10.55017 10.69615
> (12.78446 + 12.44708)/2
[1] 12.61577
>
```



横軸のAは平均発現レベル、
縦軸のMは$\log_2$(G2/G1)に相当

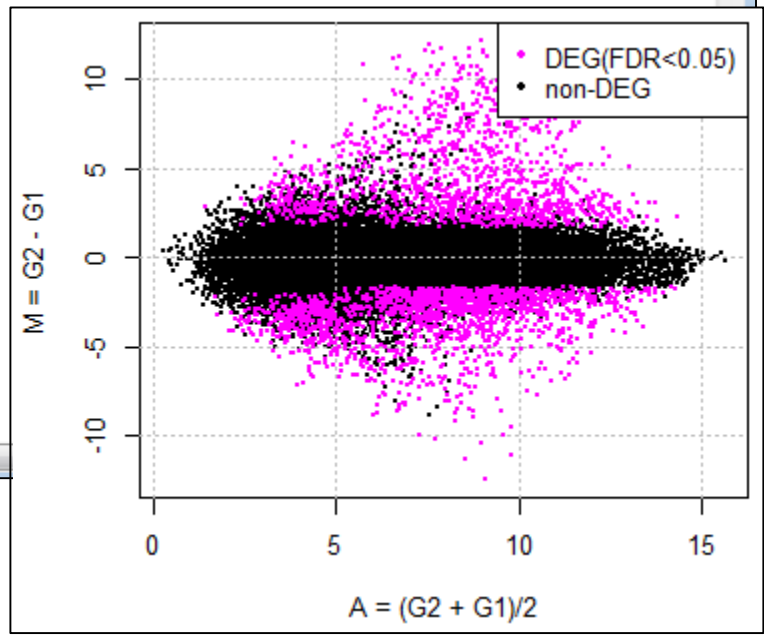
①(mean_G2 − mean_G1)の計算結果を縦軸のMとして計算できるのは、発現レベルが対数変換後のデータだから

```
mean_G1 <- apply(as.matrix(data[,data.cl==1]), 1, mean)#遺伝子ごとにG1群の平
mean_G2 <- apply(as.matrix(data[,data.cl==2]), 1, mean)#遺伝子ごとにG2群の平
M <- mean_G2 - mean_G1                    #M-A plotのM値(y軸の値)に相当するもの
A <- (mean_G1 + mean_G2)/2                #M-A plotのA値(x軸の値)に相当するものをAに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(data), data, M, A, p.value, q.value, ranking)#入力データの右側にDEG検出結果を結合した
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定したファイル名で保存
```

R Console

```
> head(tmp)
          rownames(data) BAT_fed1 BAT_fed2 LIV_fed1 LIV_fed2          M        A   p.value   q.value ranking
1367452_at     1367452_at 12.78446 12.44708 12.19593 11.95968 -0.5379634 12.34679 0.1613769 0.3365743   14911
1367453_at     1367453_at 11.80125 12.15293 11.49419 11.49189 -0.4040521 11.73506 0.1846049 0.3647413   15740
1367454_at     1367454_at 11.38990 11.16076 11.66812 12.23333  0.6753955 11.61303 0.1322055 0.2991158   13745
1367455_at     1367455_at 12.36435 12.52974 12.80589 12.96296  0.4373753 12.66573 0.1991063 0.3812952   16239
1367456_at     1367456_at 13.44849 13.54305 13.38086 13.58722 -0.0117270 13.48990 0.9686717 0.9832814   30636
1367457_at     1367457_at 10.40403 10.69632 10.78228 10.61002  0.1459767 10.62316 0.6482376 0.7815199   25794
> head(cbind(mean_G1, mean_G2))
            mean_G1  mean_G2
1367452_at 12.61577 12.07781
1367453_at 11.97709 11.49304
1367454_at 11.27533 11.95072
1367455_at 12.44705 12.88442
1367456_at 13.49577 13.48404
1367457_at 10.55017 10.69615
> 12.07781 - 12.61577
[1] -0.53796
> (12.07781 + 12.61577)/2
[1] 12.34679
>
```

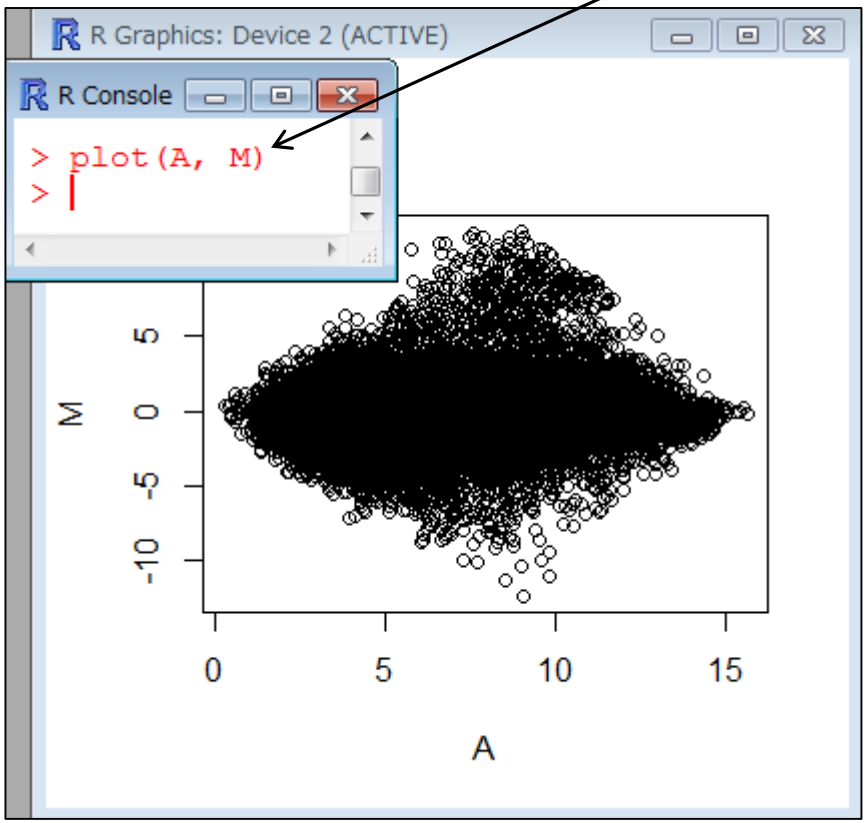
①

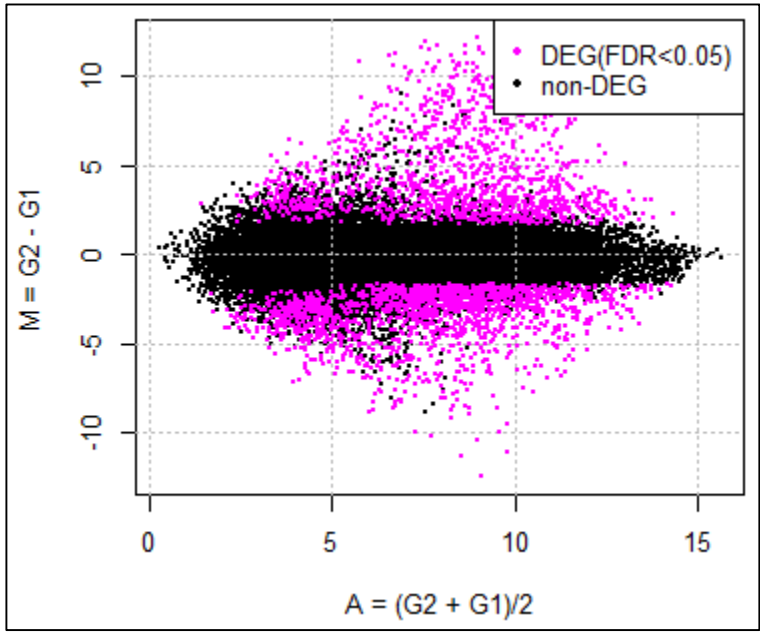


横軸のAは平均発現レベル、縦軸のMは$\log_2$(G2/G1)に相当

plotの基本形

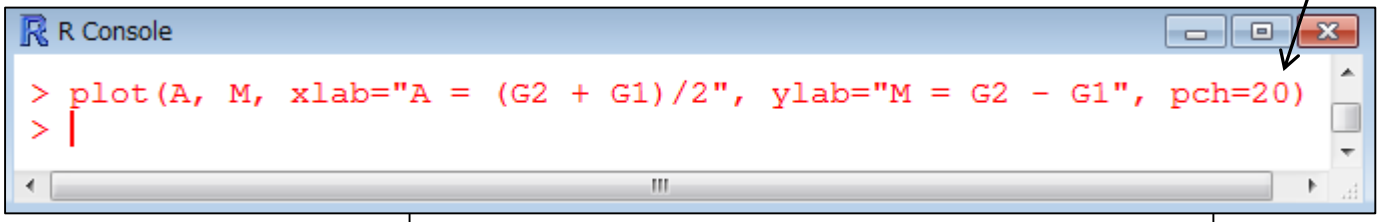
```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
```

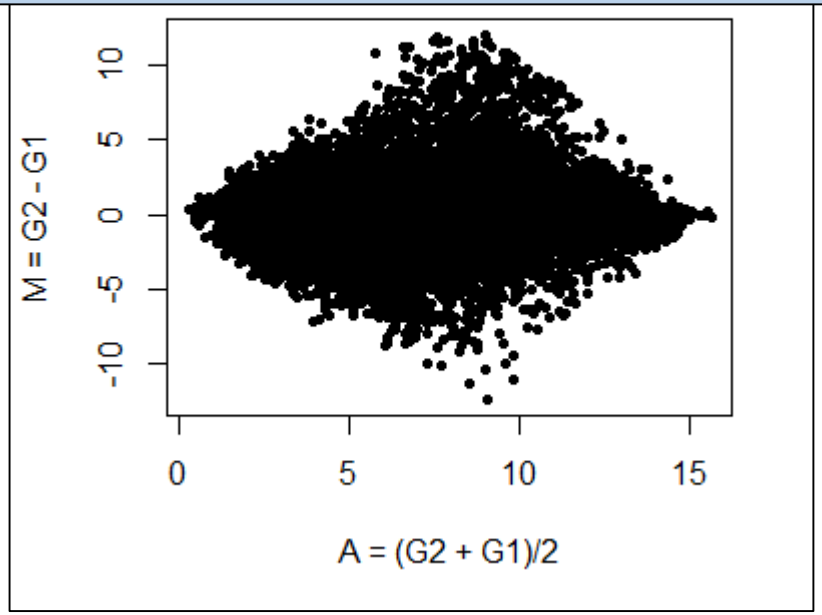


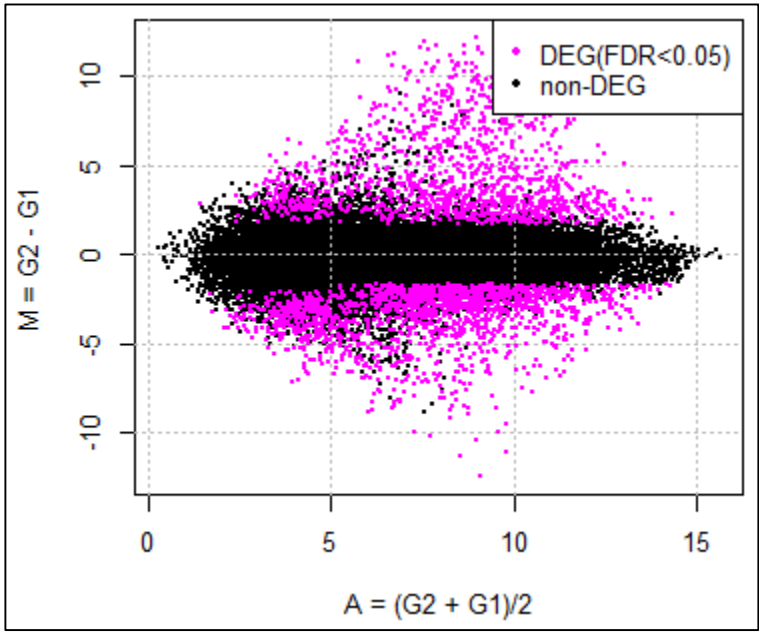


黒丸の塗りつぶしにすべく、pch=20オプションを追加

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
```

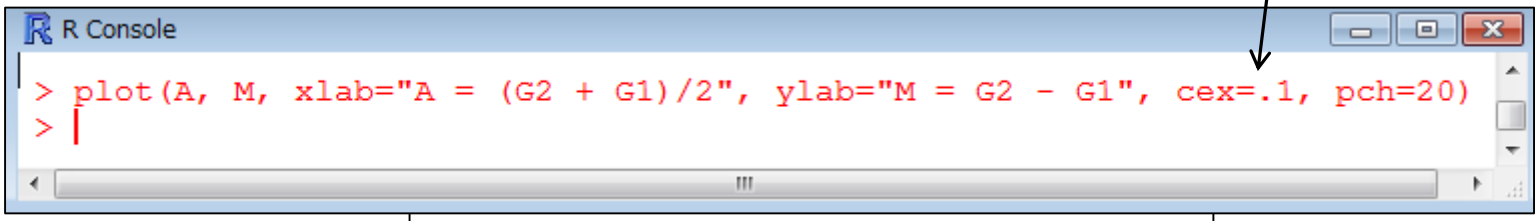


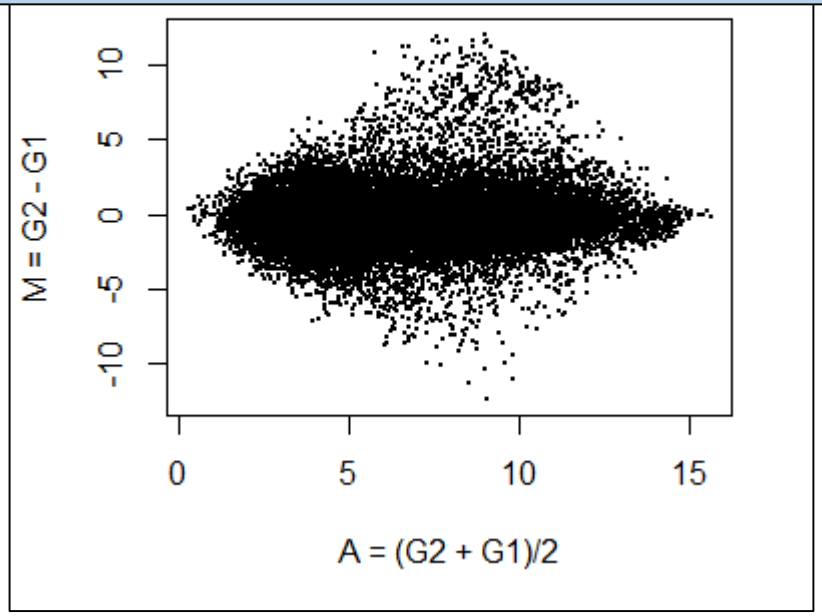


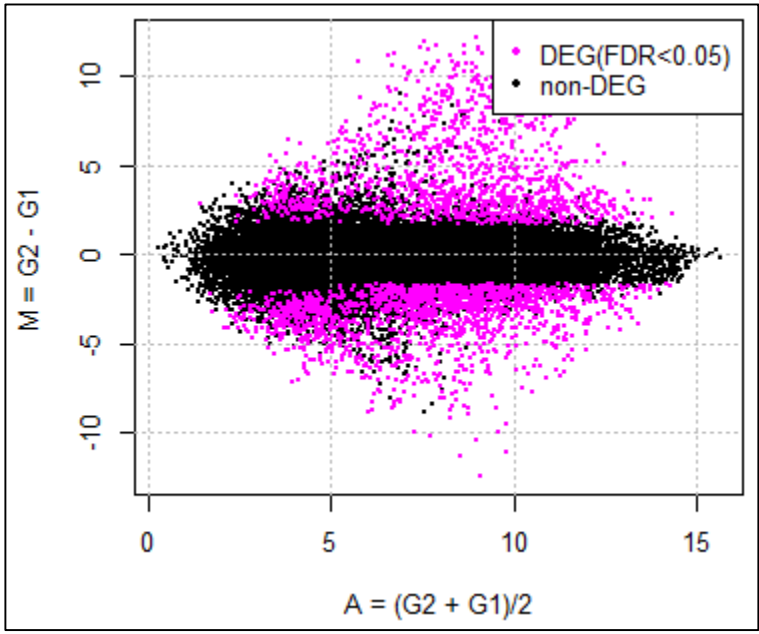


プロットの大きさをデフォルトの10%にすべく、cex=.1オプションを追加

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
```

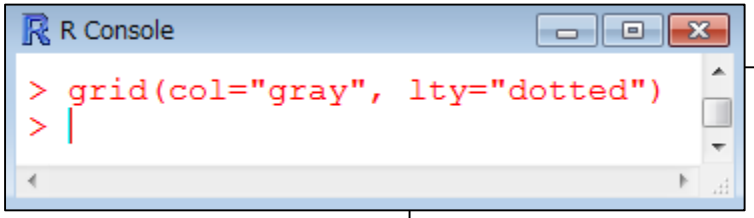


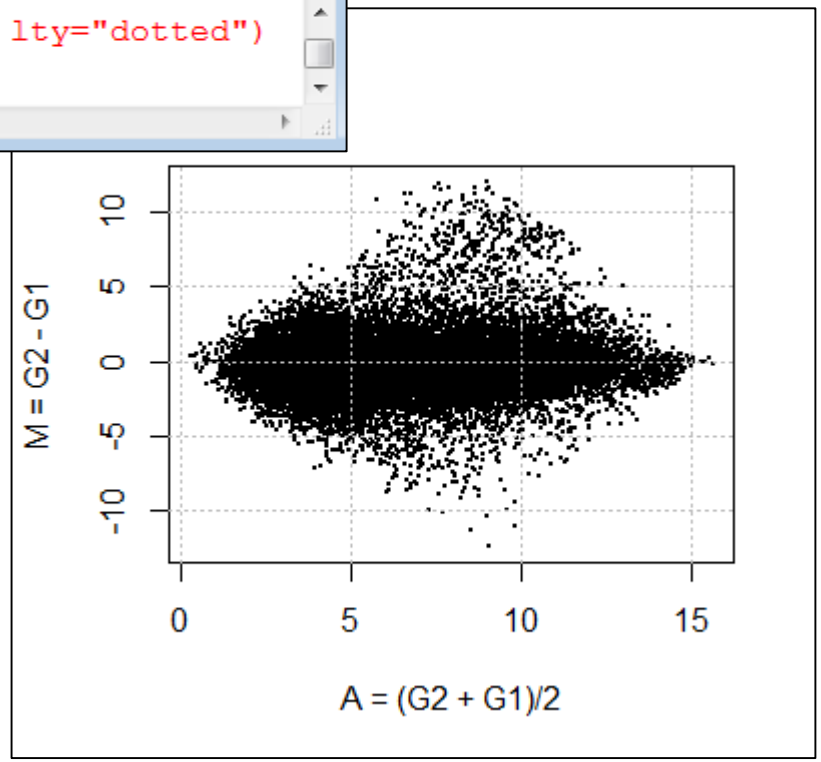


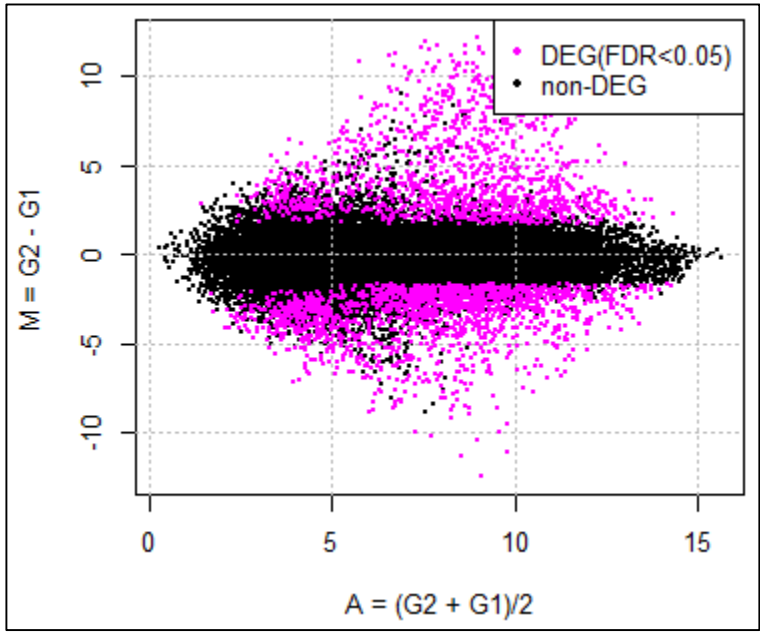


グリッド線を追加

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")                    #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                                         #おまじない↓
```





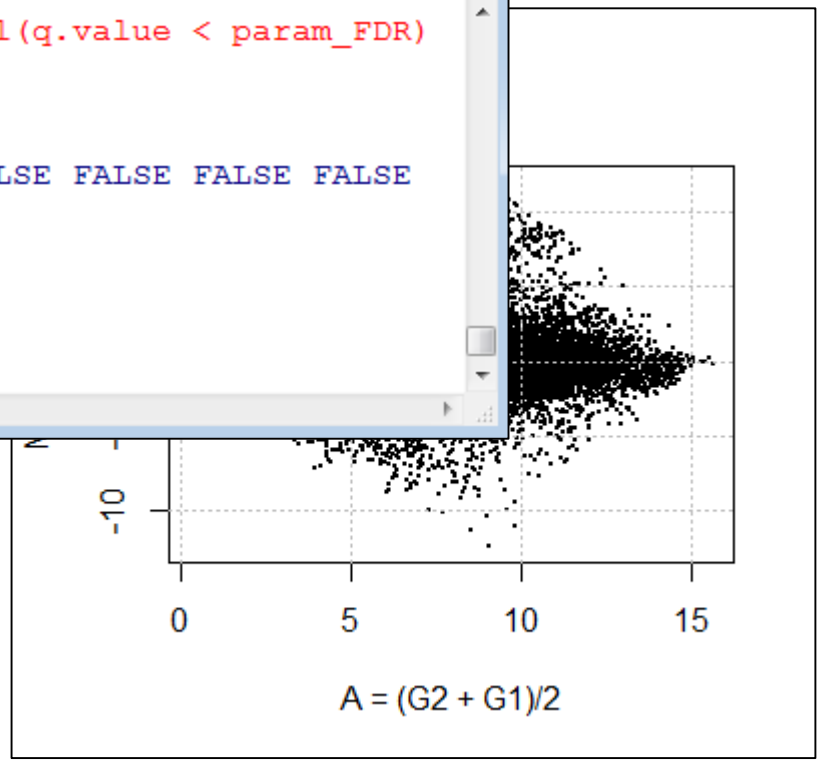


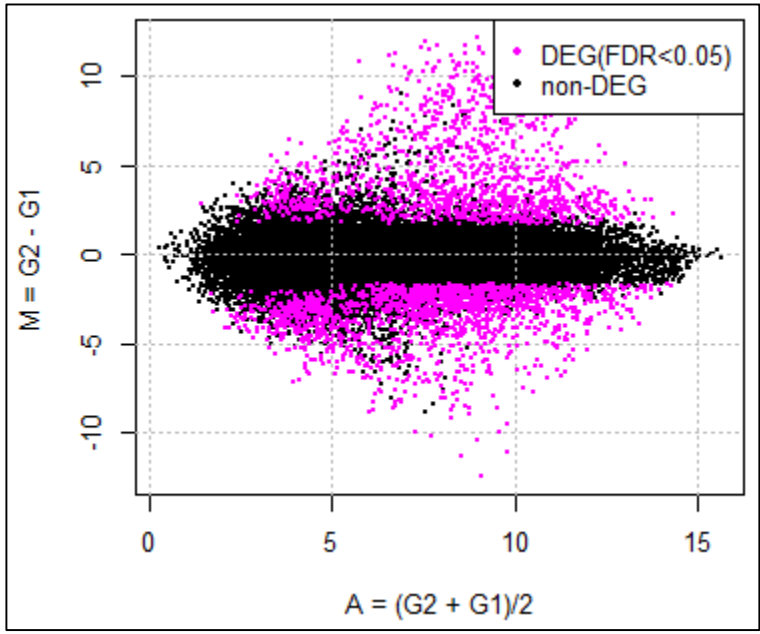
指定したFDR閾値を満たすDEGの位置情報を取得

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR)#条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
```

R Console

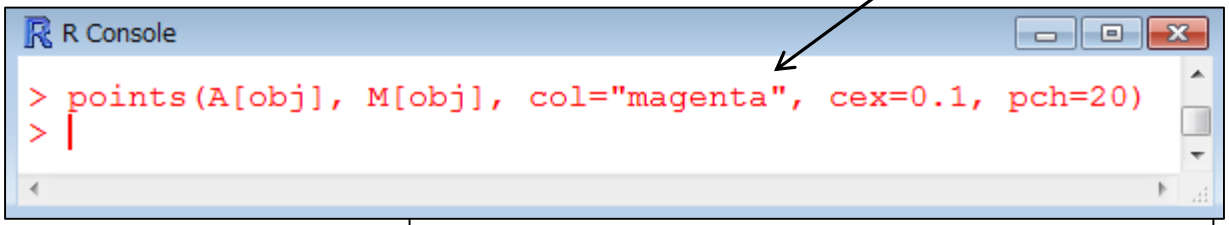
```
> obj <- as.logical(q.value < param_FDR)
> param_FDR
[1] 0.05
> head(obj)
[1] FALSE FALSE FALSE FALSE FALSE FALSE
> sum(obj)
[1] 2892
> length(obj)
[1] 31099
>
```

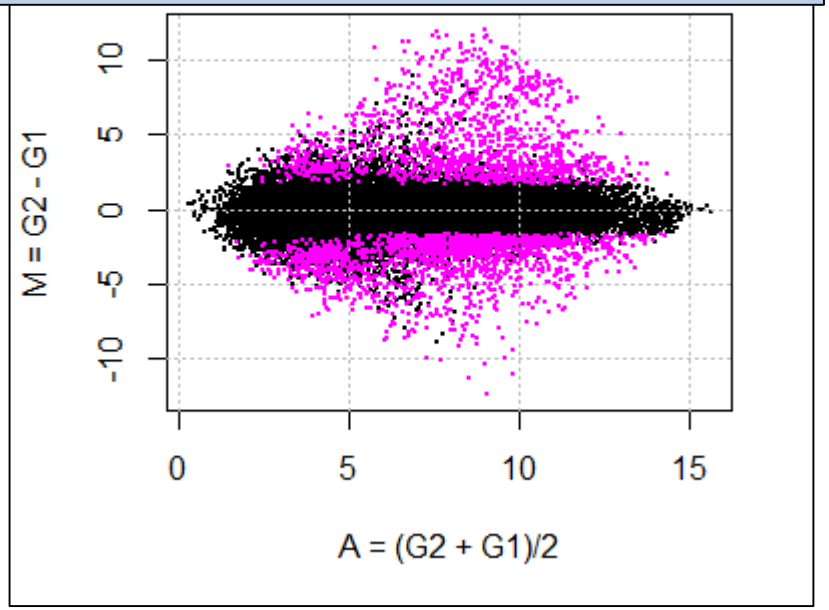


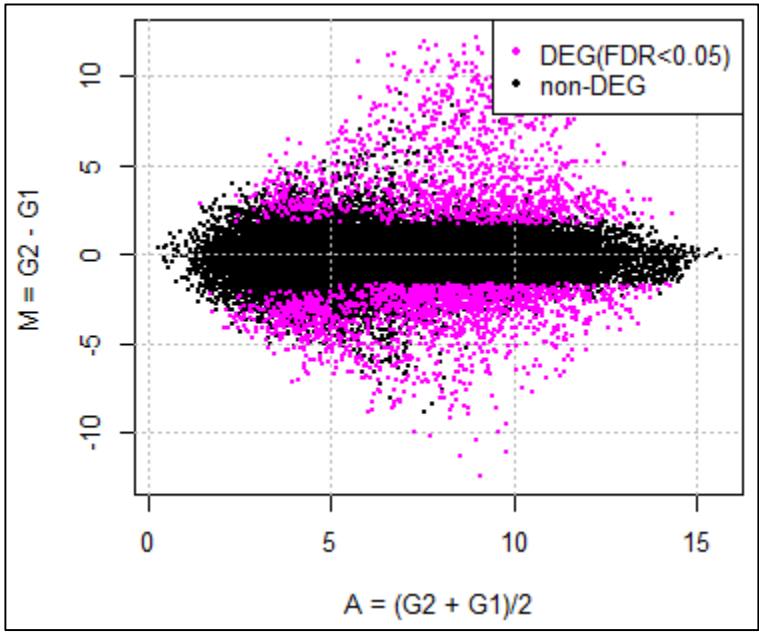


objベクトルがTRUEの場所をmagenta色で描画。cexとpchオプションの値を同じにすることで色だけを変更していることに相当

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力フ
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
```







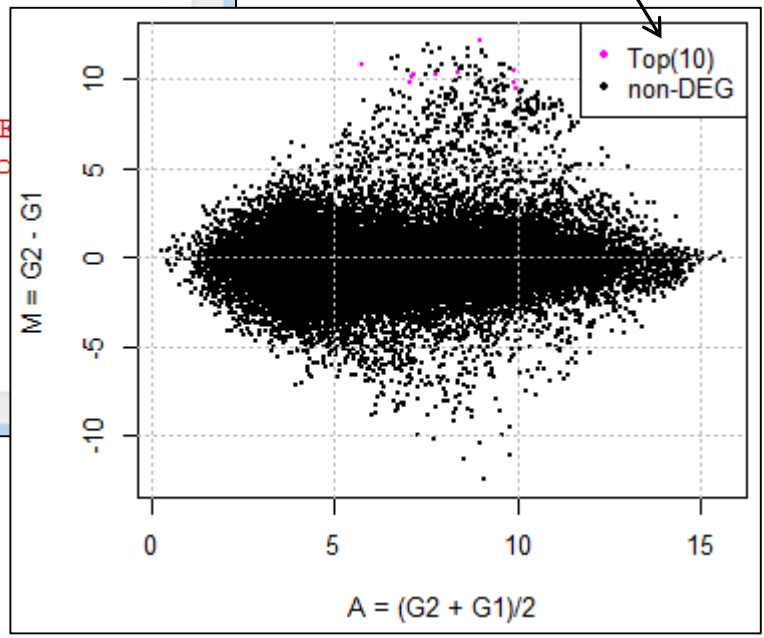
参考

上位x個のみ色を変えることも簡単にできます。rcode_limma_MAplot_basic.txtの下のほうのコードです。これは、Top10のみマゼンタ色にするやり方です。

```
##################################
#ファイルに保存(M-A plot)1
##################################
param_TOP <- 10
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示
obj <- as.logical(ranking <= param_TOP)#条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画
legend("topright", c(paste("Top(", param_TOP, ")", sep=""), "non-DEG"),#凡例を作成している
       col=c("magenta", "black"), pch=20)#凡例を作成している
dev.off()                               #おまじない
```

R Console

```
> param_TOP <- 10
> png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出$
> plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1,
> grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを$
> obj <- as.logical(ranking <= param_TOP)#条件を満たすかどうかを判定した$
> points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUE
> legend("topright", c(paste("Top(", param_TOP, ")", sep=""), "no
+        col=c("magenta", "black"), pch=20)#凡例を作成している
> dev.off()                               #おまじない
windows
      2
>
```



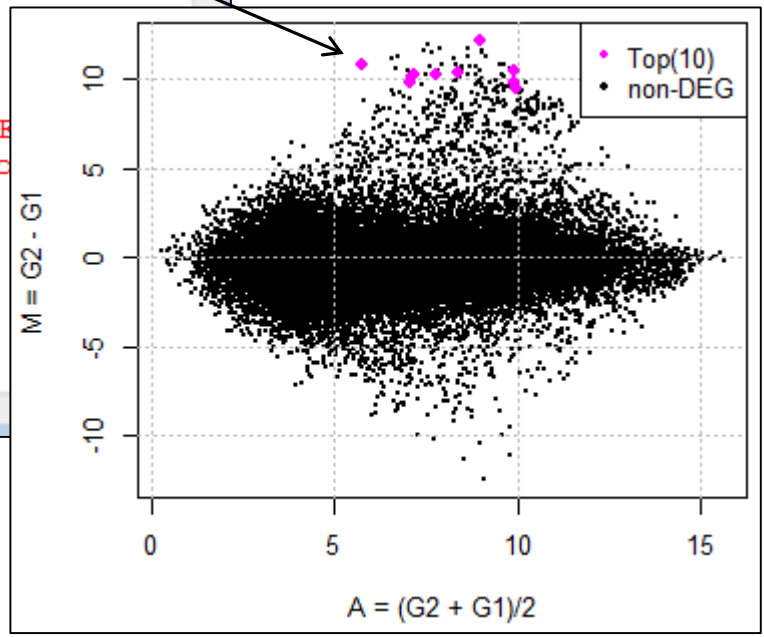
参考

上位x個のみ色を変えて大きくすることもできます。rcode_limma_MAplot_basic.txtの下のほうのコードです。cexのところの値を大きくして、通常の1.5倍の大きさにしています。

```
################################
#ファイルに保存(M-A plot)2
################################
param_TOP <- 10
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=2
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示
obj <- as.logical(ranking <= param_TOP)#条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)
points(A[obj], M[obj], col="magenta", cex=1.5, pch=20)#objがTRUEとなる要素のみ指定した色で描画
legend("topright", c(paste("Top(", param_TOP, ")", sep=""), "non-DEG"),#凡例を作成している
       col=c("magenta", "black"), pch=20)#凡例を作成している
dev.off()                               #おまじない
```

R Console

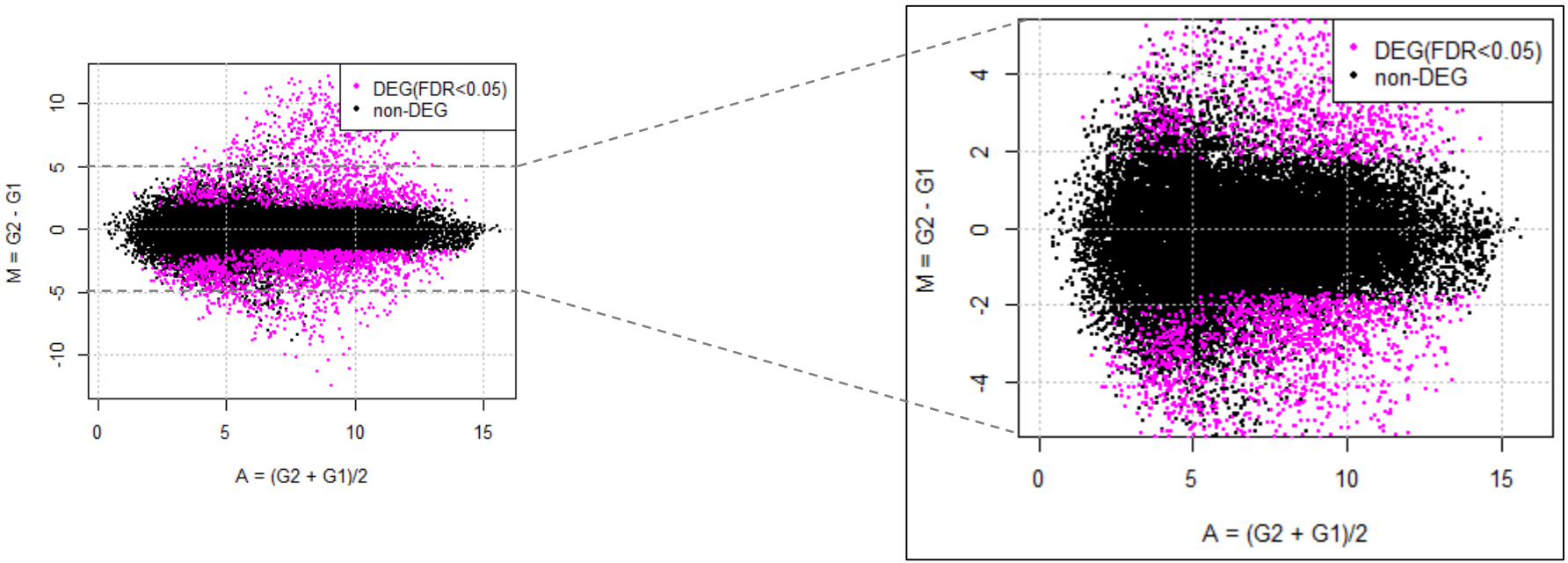
```
> param_TOP <- 10
> png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出$
> plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1,
> grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを$
> obj <- as.logical(ranking <= param_TOP)#条件を満たすかどうかを判定した$
> points(A[obj], M[obj], col="magenta", cex=1.5, pch=20)#objがTRUE
> legend("topright", c(paste("Top(", param_TOP, ")", sep=""), "no
+        col=c("magenta", "black"), pch=20)#凡例を作成している
> dev.off()                               #おまじない
windows
      2
>
```



参考

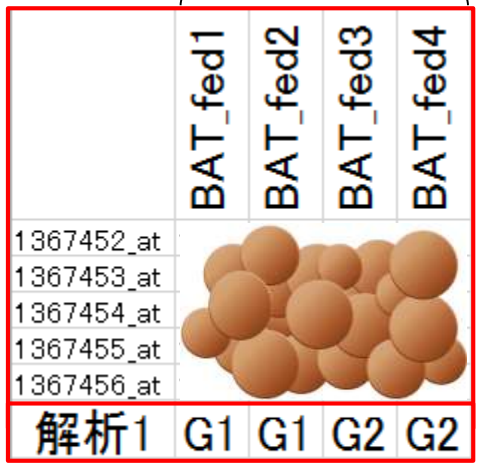
表示範囲を自在に変更可能です。
rcode_limma_MAplot_basic.txtの下のほうのコードです。

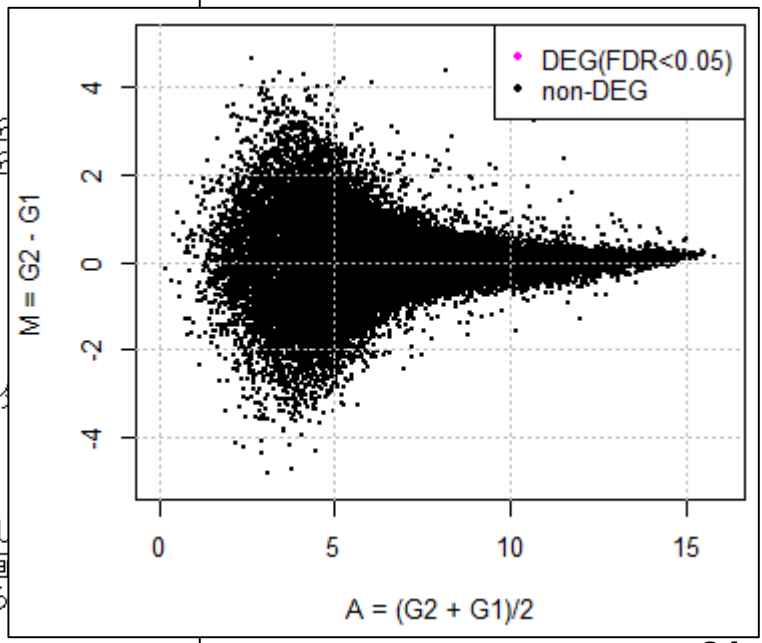
```
################################↓
#ファイルに保存(M-A plot)3↓
################################↓
param_xrange <- c(0, 16)                #M-A plotのx軸の範囲を指定↓
param_yrange <- c(-5, 5)                #M-A plotのy軸の範囲を指定↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1",    #M-A plotを描画↓
     ylim=param_yrange, xlim=param_xrange, cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRUE、non-DEGがFALSE)↓
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画↓
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している↓
       col=c("magenta", "black"), pch=20)#凡例を作成している↓
dev.off()                               #おまじない↓
↓
```



```
in_f <- "data_mas_EN.txt"                    #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge1.txt"                        #出力ファイル名を指定してout_f1に格納
out_f2 <- "hoge1.png"                        #出力ファイル名を指定してout_f2に格納
param_G1 <- 2                                #G1群のサンプル数を指定
param_G2 <- 2                                #G2群のサンプル数を指定
param_posi <- c(1,2,3,4)                   #元の発現行列上での列番号を指定
param_FDR <- 0.05                            #DEG検出時のfalse discovery rate (FDR)閾値を指定
param_fig <- c(400, 380)                     #ファイル出力時の横幅と縦幅を指定(単位はピクセル)

#必要なパッケージをロード
library(limma)                               #パッケージの読み込み

#入力ファイルの読み込みとラベル情報の作成、そしてサブセットの作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定したファイルの読み込み
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdata.clを作成
data <- data[,param_posi]                    #サブセットを抽出
colnames(data)                               #サブセット抽出後のサンプル名を表示

#本番(DEG検出)
design <- model.matrix(~data.cl)             #デザイン行列を作成した結果をdesignに格納
fit <- lmFit(data, design)                   #モデル構築(ばらつきの程度を見積もっている)
out <- eBayes(fit)                           #検定(経験ベイズ)
p.value <- out$p.value[,ncol(design)]        #p値をp.valueに格納
q.value <- p.adjust(p.value, method="BH")#q値をq.valueに格納
ranking <- rank(p.value)                     #p.valueでランキングした結果をrankingに格納
sum(q.value < param_FDR)                     #FDR < param_FDRを満たす遺伝子数を表示
mean_G1 <- apply(as.matrix(data[,data.cl==1]), 1, mean)#遺伝子ごとにG1群の平均を計算した結果を
mean_G2 <- apply(as.matrix(data[,data.cl==2]), 1, mean)#遺伝子ごとにG2群の平均を計算した結果を
M <- mean_G2 - mean_G1                       #M-A plotのM値(y軸の値)に相当するものをMに格納
A <- (mean_G1 + mean_G2)/2                   #M-A plotのA値(x軸の値)に相当するものをAに格納

#ファイルに保存(M-A plot)
param_xrange <- c(0, 16)                     #M-A plotのx軸の範囲を指定
param_yrange <- c(-5, 5)                     #M-A plotのy軸の範囲を指定
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメー
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1",     #M-A plotを描画
     ylim=param_yrange, xlim=param_xrange, cex=.1, pch=20)#M-A plotを描画
grid(col="gray", lty="dotted")               #指定したパラメータでグリッドを表示
obj <- as.logical(q.value < param_FDR) #条件を満たすかどうかを判定した結果をobjに格納(DEGがTRU
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定した色で描画
legend("topright", c(paste("DEG(FDR<", param_FDR, ")", sep=""), "non-DEG"),#凡例を作成している
       col=c("magenta", "black"), pch=20)#凡例を作成している
dev.off()                                    #おまじない
```

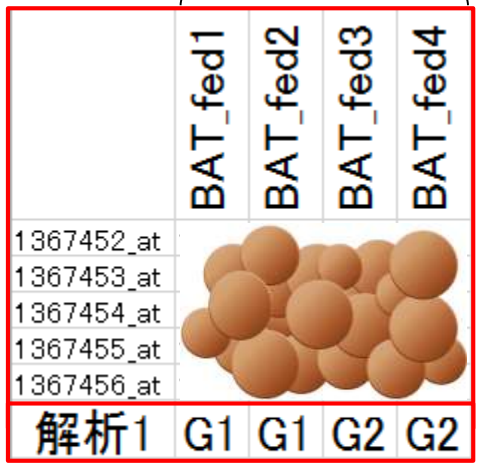
param_posiをc(5,6,7,8)に変更すればBAT_fas内（同一群内）のばらつきの程度を調べることに相当。
rcode_limma_MAplot_basic2.txtです





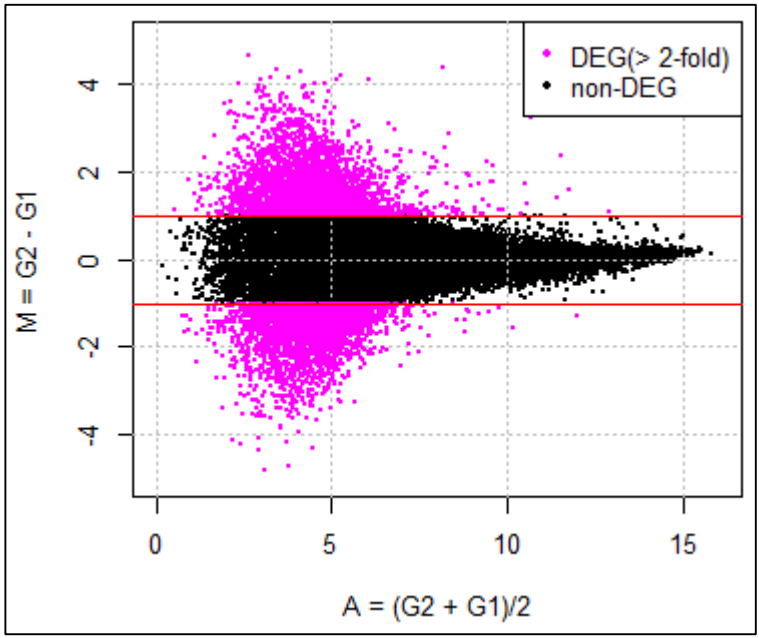
```
#################################################################
###   ファイルに保存(M-A plot) 2倍以上発現変動をマゼンタ色に
#################################################################
param_FC <- 2                          #倍率変化の閾値を指定
param_xrange <- c(0, 16)               #M-A plotのx軸の範囲を指定
param_yrange <- c(-5, 5)               #M-A plotのy軸の範囲を指定
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1",    #M-A plotを描画
     ylim=param_yrange, xlim=param_xrange, cex=.1, pch=20)#M-A plotを描画
grid(col="gray", lty="dotted")         #指定したパラメータでグリッドを表示
obj <- as.logical(abs(M) >= log2(param_FC))#条件を満たすかどうかを判定した結果を
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指
legend("topright", c(paste("DEG(> ", param_FC, "-fold)", sep=""), "non-DEG"),#凡
       col=c("magenta", "black"), pch=20)#凡例を作成している
abline(h=log2(param_FC), col="red")    #M=log2(param_FC)の直線を表示
abline(h=-log2(param_FC), col="red")   #M=-log2(param_FC)の直線を表示
dev.off()                              #おまじない
```

rcode_limma_MAplot_basic2.txt の下の方のコードです。Fold-changeによるDEG検出の危険性がよくわかります。



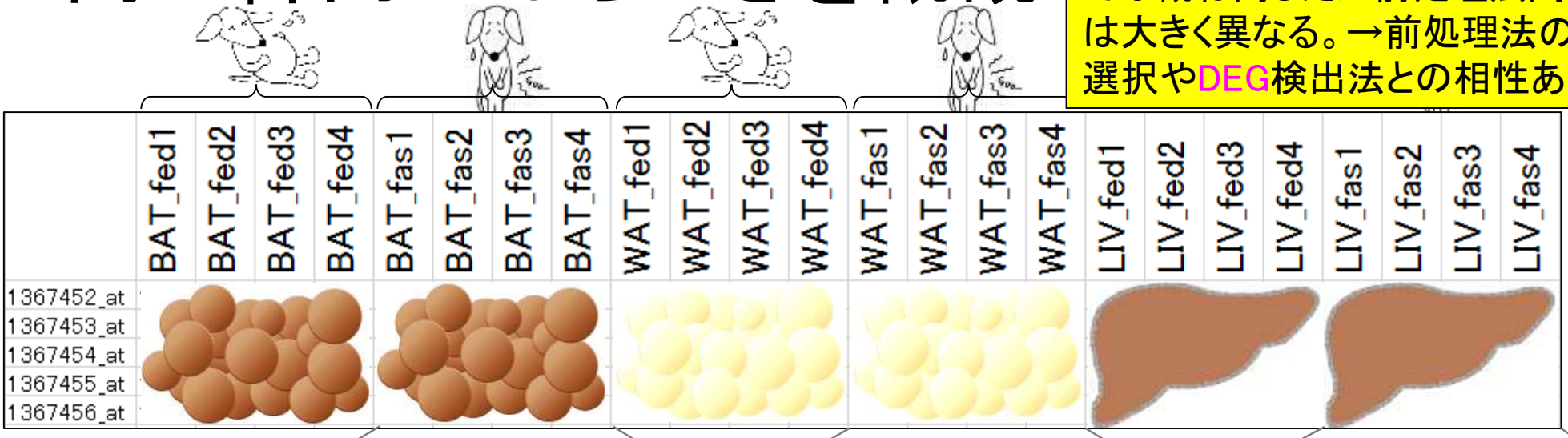
R Console

```
> param_FC <- 2                                #倍率変化の閾値を$
> param_xrange <- c(0, 16)                     #M-A plotのx軸の範$
> param_yrange <- c(-5, 5)                     #M-A plotのy軸の範$
> png(out_f2, pointsize=13, width=param_fig[1], height=para$
> plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1",    $
+      ylim=param_yrange, xlim=param_xrange, cex=.1, pch=20$
> grid(col="gray", lty="dotted")               #指定したパラメ$
> obj <- as.logical(abs(M) >= log2(param_FC))#条件を満たす$
> points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#ob$
> legend("topright", c(paste("DEG(> ", param_FC, "-fold)", $
+        col=c("magenta", "black"), pch=20)#凡例を作成して$
> abline(h=log2(param_FC), col="red")          #M=log2(param_FC)$
> abline(h=-log2(param_FC), col="red")         #M=-log2(param_FC)$
> dev.off()                                    #おまじない
null device
          1
> sum(obj)
[1] 5354
>
```

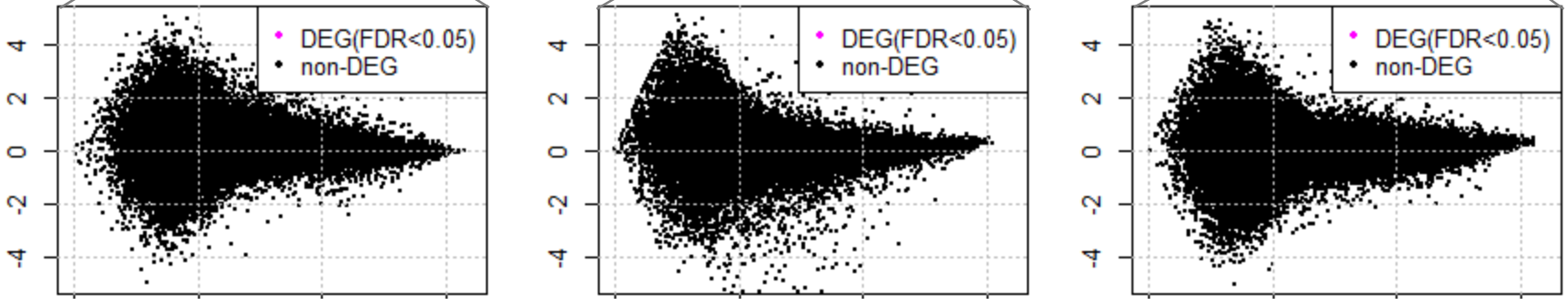


同一群内のばらつきを概観

IBMT法実行結果。同一群内のばらつきの程度は、前処理法内では概ね同じだが前処理法間では大きく異なる。→前処理法の選択やDEG検出法との相性あり



MAS5データ



RMAデータ