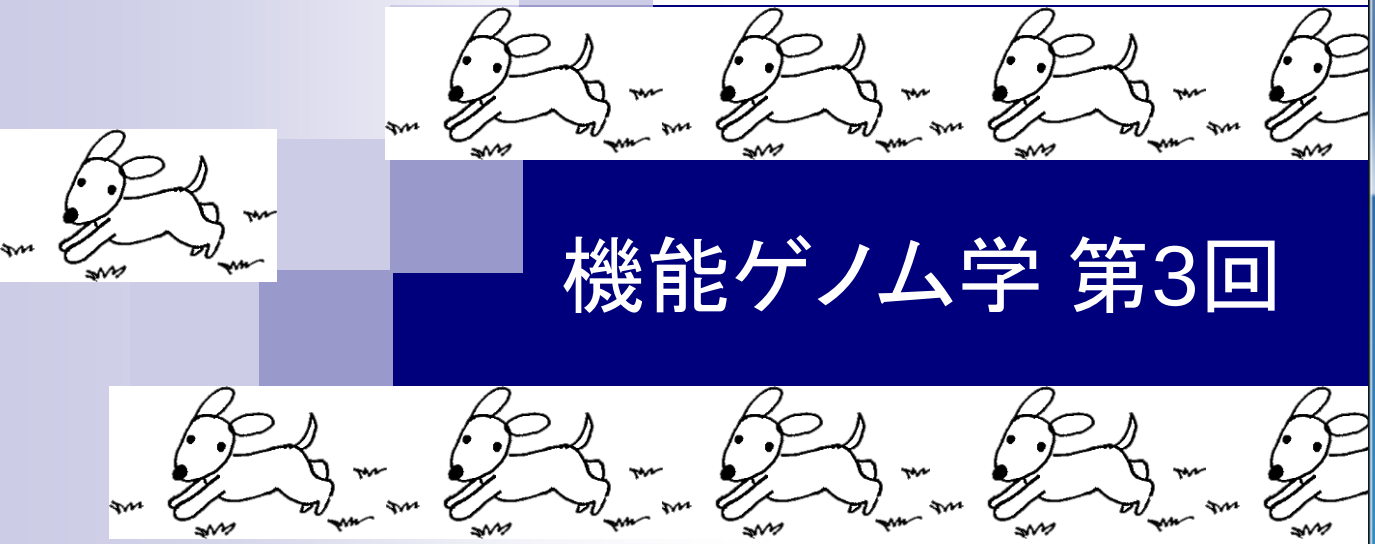
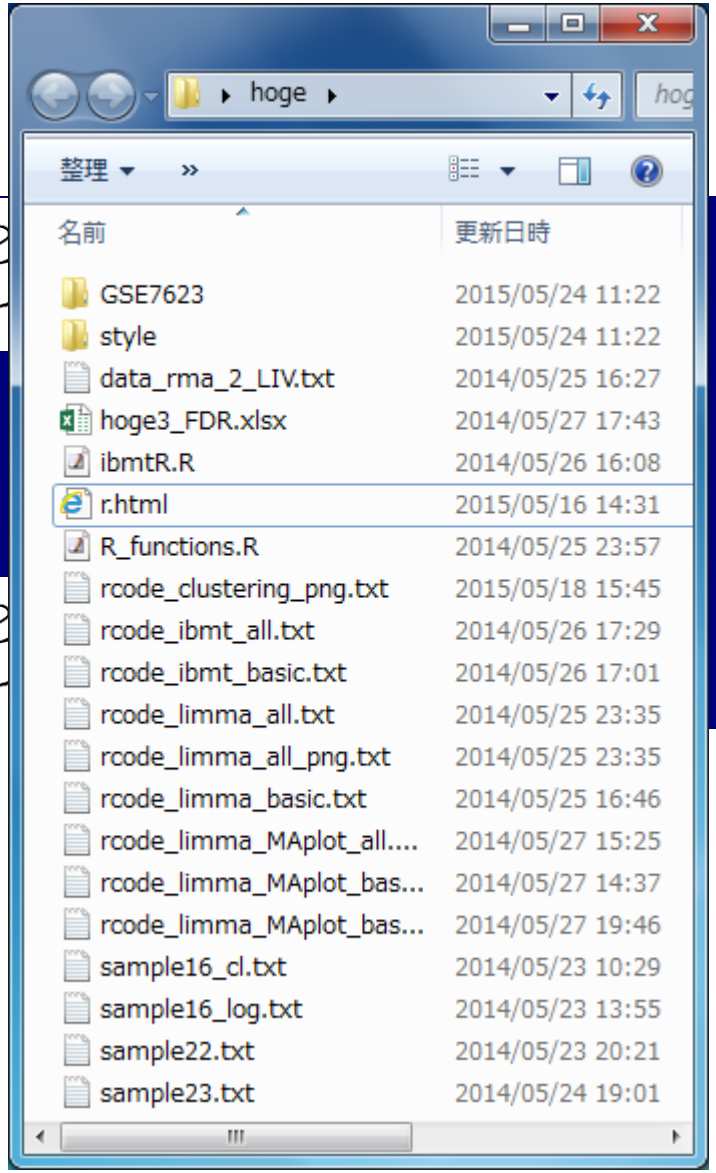
USBメモリ中のhogeフォルダをデスクトップにコピーしておいてください。



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ム
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。また、Rのスキルアップを目指します

- 第1回（2016年05月09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（2016年05月16日）
  - 数値行列作成、クラスタリング、実験デザイン
  - 教科書の3.2節周辺
- 第3回（2016年05月23日）
  - 発現変動解析（多重比較問題とFDR）、各種プロット（M-A plot）
  - 教科書の3.2節と4.2節周辺
- 第4回（2016年05月30日）
  - 機能解析（Gene Ontology解析やパスウェイ解析）

感想やコメントのところ

- transcript_nrowのところが何をカウントしているのか気になった（5/9分）
  - transcriptなので転写物をカウントしている、ということでしょう。geneとtranscriptとexonが同じというあたりが気になったのかもしれませんが、課題のgff3ファイルはバクテリア（乳酸菌）なので、実質的にgeneとtranscriptとexonが同じ（intronがない）ので違和感があるだけだと思います
- libraryをどんどんインストールすると重いので気が進まない。上手な方法はないか？（5/9分）
  - 研究室の有線LAN環境でお昼休みなどを利用すればいいのでは?!アグリバイオ事務局だと2時間程度で全て完了します。
  - ディスク容量に関する指摘だったとしても、推奨パッケージを全てインストールしたとしても5GB程度だったと思います。もちろん受け止め方はヒトそれぞれだとは思いますが…。
- GGRNAの検索結果について（5/9分；後述）
- マイクロアレイもヒトやマウスに対してまだ使われているということでしょうか？（5/9分）
  - DBの統計上はそういうことになります。
- txtファイルが文字化けして「sep="\t"」が「sep="¥t"」になってしまっていたり、Macだから?（5/16分）
  - 文字化けではなく文字コード?!の違いのためこうなります。詳細は「円マーク　バックスラッシュ」でググってください。MacintoshとWindowsの違いだから、というのは基本的に正解でいいです。
- クラスター間の距離の違いについて、具体的にどれくらい離れているとどうだとか教えて（5/16分）
  - ざっくり言えば、「群内」と「群間」のバラツキの相対的な違いの程度によって評価が異なります

GGRNAの検索結果

統合遺伝子検索 Help | Advanced search | English 旧バージョン

GGRNA ver.2

1387260_at 検索

Zoo (All organisms in RefSeq)

2016-05-18 12:09:44, GGRNA : RefSeq release 75 (Mar, 2016)

Summary:

- seq:TTTTCTATATAGTTCCTTGCCTTAA (100) ①
- seq:TGCTCTTTGCCTTCTAAAAAGCCAT (4)
- seq:GTTTATTCCCAAGTATGCCTTTAAG (6)
- seq:GTGCTTGGTGAGTCGTGGTTCTAAA (3)
- seq:GAGACTGGATCTTCTATCATTCCAA (5)
- seq:GAAGACCAGAGTTCCCTTGAATTGT (3)
- seq:ATAAGCCTGGTTTATTTCTGTAACA (2)
- seq:AGTCTGTTATGCACTGTGGTTTCAG (95)
- seq:AATCTATTTTTGTCTTCCGATCTAC (53)
- seq:AAAGTTCTCCAACTGCTGCATACTT (4)
- seq:AAAGACCACCTTGTATGCTCTTTGC (1)
- **INTERSECTION (1)**

Results:

検索語に色がつきます。重なると色が濃く表示されます。

Rattus norvegicus Kruppel-like factor 4 (gut) (Klf4), mRNA. (2393 bp)
aatcacagaacagatggggtct**gagactggatcttctatcattccaa**taccaaatccgacttgaacaagactggactt
acaaaatgccaaggggtgactggaagtttgtggatatcagggtatacattaaatcagtgacctgggggggagg**gaaga**
ccagagttcccttgaattgtgcttcaatgatgcaatatacatgg**aaagaccaccttgtatgctctttgccttctaaa**
aagccattatgacgtcagaggaagaggaagcaattcaggtacagaacgtgttctaatagcctaaacgatg**gtgcttg**
gtgagtcgtggttctaaaggtaccaaacgggggagcc**aaagttctccaactgctgcatactt**tgacaaggaa**aa**
tctatttttgtcttccgatctacatttatgacctaagtcaggtaa**ataagcctggtttatttctgtaaca**ttttttatgca
gac**agtctgttatgcactgtggtttcag**atgtgcaataatttgtacaatg**gtttattcccaagtatgccttaag**ca
gaacaaatgtgt**ttttctatatagttccttgccttaa**taaatatgtaatataaatttaa...
position 1791 1919 1968 1983 2071 2115 2150 2197 2236 2283 2322
Synonym: GKLF
NM_053713.1 - Rattus norvegicus (Norway rat) - NCBI - UCSC - RefEx(expression)

①この数値は何?についてですが、GGRNAはRefSeqという様々な生物種の転写物配列集合を高速に検索するものであり、このプローブ配列と同じ配列を持つものがDB中に計100個あるというのが私の認識です。

- seq:TTTTCTATATAGTTCCTTGCCTTAA (100)
- **INTERSECTION (100)**

Results:

検索語に色がつきます。重なると色が濃く表示されます。

PREDICTED: Pantholops hodgsonii Krueppel-like factor 4-like (LOC102317792), mRNA. (1472 bp)
attcaggtacagaaaatggtgtttaatagcctaaacgatggtgcttggtaagtcttggttctaaaggtaccaaatgaggcc
aaagttctcaaactgctgcatactttgacaaggaaaatctatttttgtcttccgatccgatctacatttatgacctaagtcagg
taaatacacctggtttacttaatctttttatgcagacagtctgttatgcactgtggtttcaaatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacgaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctatttttgtatatttgtaaactgcaaagtaaaaaaatgaacattttgtggagtttgtattttgcatactcaagg
tgagaattaagttttaaataaacctataatattttatctgaaaaaaaaaca
position 1292
XM_005977571.1 - Pantholops hodgsonii (chiru) - NCBI

PREDICTED: Acinonyx jubatus Kruppel-like factor 4 (gut) (KLF4), mRNA. (1809 bp)
acagaaaatgtgttttaataacctaaacgatggtgcttggtgagtcttggttctaaaggtaccaaatgaggaagccaaagt
tctcaaactgctgcatacttttgcatactgtgacaatgaaaatctatttttgtctgccgatctacatttatgacagaaatcagg
taaatacacctggtttacttaacatttttatgcagacagtctgttatgcactgtggtttcagatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacaaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctatttttgtatatttgtaaactacaaagtaaaaaaaatgaacattttgtggagtttgtattttgcatactcaag
gtgagaattaagttttaaataaacctataatatttta
position 1644
XM_015074164.1 - Acinonyx jubatus (cheetah) - NCBI

PREDICTED: Leptonychotes weddellii Krueppel-like factor 4-like (LOC102749632), partial mRNA. (2167 bp)
ttcaggtacagaaaatgtttttaataacctaaacgatggtgcttggtgagtcttggttctaaaggtaccaaatgaggaagc
caaagttctcaaactgctgcatactttgacaaggaaaatctatatttgtcttccgatctacatttatgacctaagtcaggtaa
atacgcctggtttatttcttttaacatttttatgcagacagtctgttatgcactgtggtttcagatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacaaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctatttttgtatatttgtaaactacaaagtaaaaaaatgaacattttgtggagtttgtattttgcatactcaagg
tgagaattaagttttaaataaacctataatattttatctgaa
position 1997
XM_006750182.1 - Leptonychotes weddellii (Weddell seal) - NCBI

GGRNAの検索結果

統合遺伝子検索 Help | Advanced search | English 旧バージョン

GGRNA ver.2

1387260_at 検索

Zoo (All organisms in RefSeq)

2016-05-18 12:09:44, GGRNA : RefSeq release 75 (Mar, 2016)

Summary:

- seq:TTTTCTATATAGTTCCTTGCCTTAA (100)
- seq:TGCTCTTTGCCTTCTAAAAAGCCAT (4)
- seq:GTTTATTCCCAAGTATGCCTTTAAG (6)
- seq:GTGCTTGGTGAGTCGTGGTTCTAAA (3)
- seq:GAGACTGGATCTTCTATCATTCCAA (5)
- seq:GAAGACCAGAGTTCCCTTGAATTGT (3)
- seq:ATAAGCCTGGTTTATTTCTGTAACA (2)
- seq:AGTCTGTTATGCACTGTGGTTTCAG (95)
- seq:AATCTATTTTTGTCTTCCGATCTAC (53)
- seq:AAAGTTCTCCAACTGCTGCATACTT (4)
- seq:AAAGACCACCTTGTATGCTCTTTGC (1)
- **INTERSECTION (1)**

Results:

①

検索語に**色がつきます**。重なると**色が濃く表示されます**。

Rattus norvegicus Kruppel-like factor 4 (gut) (Klf4), mRNA. (2393 bp)

aatcacagaacagatggggtct**gagactggatcttctatcattccaa**taccaaatccgacttgaacaagactggactt
acaaaatgccaaggggtgactggaagtttgtggatatcagggtatacattaaatcagtgacctgggggggagg**gaaga**
ccagagttcccttgaattgtgcttcaatgatgcaatatacatgg**aaagaccaccttgtatgctctttgccttctaaa**
aagccattatgacgtcagaggaagaggaagcaattcaggtacagaacgtgttctaatagcctaaacgatg**gtgcttg**
gtgagtcgtggttctaaaggtaccaaacgggggagcc**aaagttctccaactgctgcatactt**tgacaaggaa**aa**
tctatttttgtcttccgatctacatttatgacctaagtcaggtaa**ataagcctggtttatttctgtaaca**tttttatgca
gac**agtctgttatgcactgtggtttcag**atgtgcaataatttgtacaatg**gtttattcccaagtatgccttaag**ca
gaacaaatgtgt**ttttctatatagttccttgccttaa**taaatatgtaatataaatttaa...

position 1791 1919 1968 1983 2071 2115 2150 2197 2236 2283 2322

Synonym: GKLF

NM_053713.1 - Rattus norvegicus (Norway rat) - NCBI - UCSC - RefEx(expression)

①なぜプローブの重なり毎に色分けされているか?についてですが、例えば「重複領域をもつプローブのシグナル強度は似ているだろう」といった予想が立ちますし、そのあたりの確認や重複効果の補正などを行う必要性についてひらめくかもしれません。

- INTERSECTION (100)

Results:

検索語に**色がつきます**。重なると**色が濃く表示されます**。

PREDICTED: Pantholops hodgsonii Kruppel-like factor 4-like (LOC102317792), mRNA. (1472 bp)

attcaggtacagaaaatggtgtttaatagcctaaacgatggtgcttggtaagtcttggttctaaaggtaccaaatgaggcc
aaagttctcaaactgctgcatactttgacaaggaaaatctatttttgtcttccgatccgatctacatttatgacctaagtcagg
taaatacacctggtttacttaatctttttatgcagacagtctgttatgcactgtggtttcaaatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacgaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctattttgtatatttgtaaactgcaaagtaaaaaaatgaacattttgtggagtttgtattttgcatactcaagg
tgagaattaagttttaaataaacctataatattttatctgaaaaaaaaaca

position 1292

XM_005977571.1 - Pantholops hodgsonii (chiru) - NCBI

PREDICTED: Acinonyx jubatus Kruppel-like factor 4 (gut) (KLF4), mRNA. (1809 bp)

acagaaaatgtgttttaataacctaaacgatggtgcttggtgagtcttggttctaaaggtaccaaatgaggaagccaaagt
tctcaaactgctgcatacttttgcatactgtgacaatgaaaatctatttttgtctgccgatctacatttatgacagaaatcagg
taaatacacctggtttacttaacatttttatgcagacagtctgttatgcactgtggtttcagatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacaaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctattttgtatatttgtaaactacaaagtaaaaaaaatgaacattttgtggagtttgtattttgcatactcaag
gtgagaattaagttttaaataaacctataatatttta

position 1644

XM_015074164.1 - Acinonyx jubatus (cheetah) - NCBI

PREDICTED: Leptonychotes weddellii Kruppel-like factor 4-like (LOC102749632), partial mRNA. (2167 bp)

ttcaggtacagaaaatgtttttaataacctaaacgatggtgcttggtgagtcttggttctaaaggtaccaaatgaggaagc
caaagttctcaaactgctgcatactttgacaaggaaaatctatatttgtcttccgatctacatttatgacctaagtcaggtaa
atacgcctggtttatttcttttaacatttttatgcagacagtctgttatgcactgtggtttcagatgtgcaataatttgtacaatg
gtttattcccaagtatgccttaagcagaacaaatgttt**ttttctatatagttccttgccttaa**taaatatgtaatataaattt
aagcaaacttctattttgtatatttgtaaactacaaagtaaaaaaatgaacattttgtggagtttgtattttgcatactcaagg
tgagaattaagttttaaataaacctataatattttatctgaa

position 1997

XM_006750182.1 - Leptonychotes weddellii (Weddell seal) - NCBI

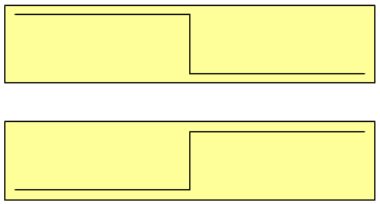
Contents

- 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）

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

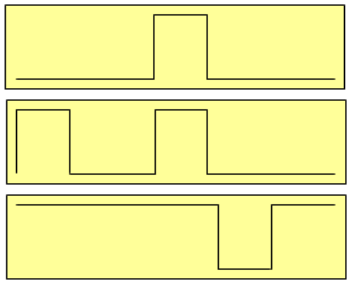
発現変動遺伝子同定



遺伝子発現行列

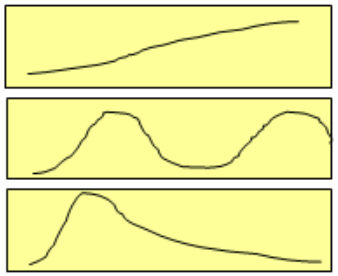
二群間比較用

	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）

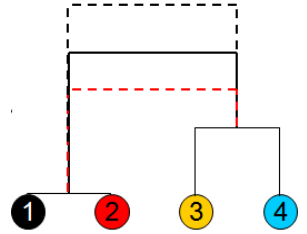
	S1	S2	S3	S4	…
gene *1*	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	…
gene *2*	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	…
…	…	…	…	…	…
gene *i*	$x_{i,1}$	$x_{i,2}$	$x_{i,3}$	$x_{i,4}$	
…	…	…	…	…	…
gene *n*	$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$x_{n,4}$	…



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*	$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$x_{n,4}$	…

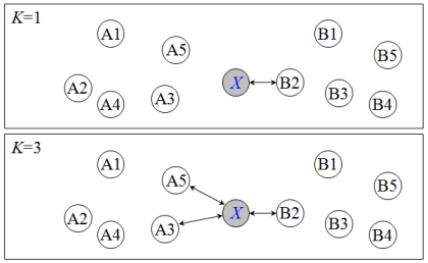
クラスタリング



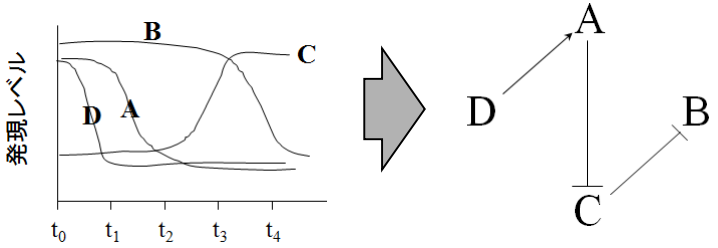
機能解析

- Gene Ontology（GO）
- パスウェイ解析

分類（診断）



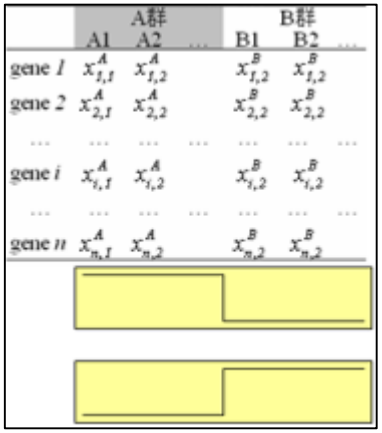
遺伝子ネットワーク推定

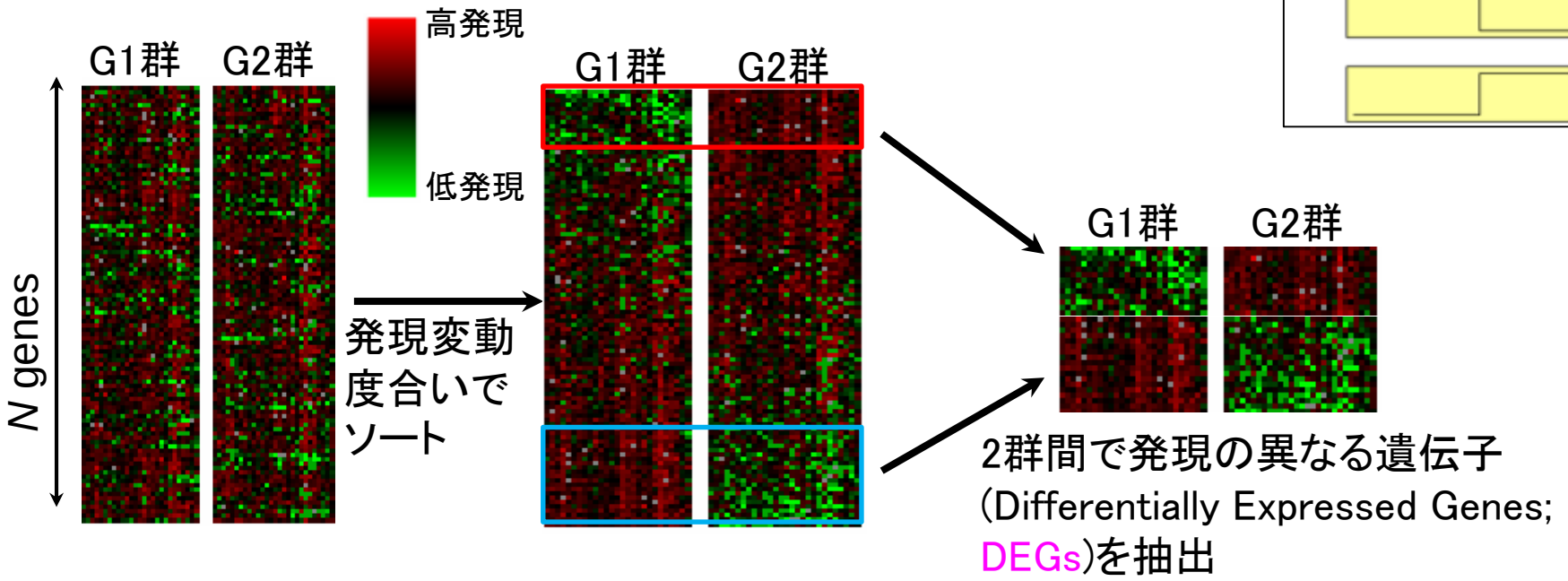


2群間比較

比較したいグループ(群)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 vs. 低温、通常 vs. 乾燥)
- 味の違い(おいしい vs. まずい)
- サンプルの状態の違い(癌 vs. 正常)



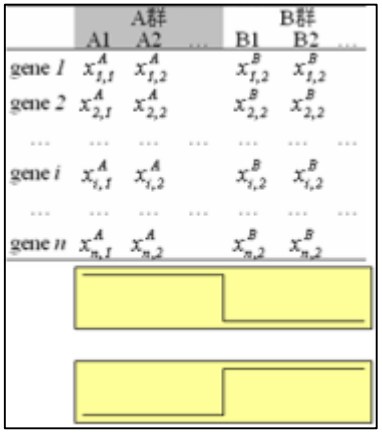


2群間比較

①理想的なパターンyと②遺伝子ベクトル間の
③相関係数rを遺伝子ごとに計算。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遺伝子をランキング

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



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	③ 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	0.98
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

①パターンマッチング法の、②例題1。③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。まだコピペしなくてよい

(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)
- 解析 | 発現変動 | 2群間 | 対
- 解析 | 発現変動 | 2群間 | 対
- 解析 | 発現変動 | 2群間 | 対
- 解析 | 発現変動 | 3群間 | 対
- 解析 | 発現変動 | 3群間 | 対
- 解析 | 発現変動 | 3群間 | 対
- 解析 | 発現変動 | 3群間 | 対
- 解析 | 発現変動 | 3群間 | 対
- 解析 | 発現変動 | 多群間 |
- 解析 | 発現変動 | 多群間 |
- 解析 | 発現変動 | 多群間 |
- 解析 | 発現変動 | 多群間 |
- 解析 | 発現変動 | 多群間 |

① ② ③

解析 | 発現変動 | 2群間 | 対応なし | パターンマッチング法

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
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
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

#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
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```

入出力の関係

①テンプレートパターン情報ファイル。2列目がテンプレートパターン。②(log変換後の)発現データファイル。赤枠で示すように、列の並びは同じ

解析 | 発現変動 | 2群間 | 対応なし | パターンマッチング法

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
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
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

#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
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```

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

①

rownar	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

②

入出力の関係

①出力ファイルは、②入力ファイルの右側に相関係数rの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

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。

②

①

```
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
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
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

#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
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```

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

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
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
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
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

コードの解説

黒枠内をコピペ。ここは、①入力ファイル情報読み込み部分。②dataオブジェクトを確認

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",
```

①

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
>
```

①

②

コードの解説

①テンプレートパターン情報の②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

②

①

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]                          #テンプレートパター$
> data.cl
 [1] 1 1 1 1 1 1 0 0 0 0 0 0
>
```

コードの解説

①テンプレートパターン情報を、②の部分で読み込んで得られた、③hogeの中身は入力ファイルと同じだが…

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("

#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.nam=1, sep="\t"
hoge <- read.table(in_f2, sep="\t", quote="")  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

#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
```

②

③

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

①

コードの解説

①欲しいのはhoge行列の2列目部分のみなので、②そこの部分のみ抽出

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
```

②

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

①

R Console

```
> hoge
①
      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
②
> hoge[, 2]
 [1] 1 1 1 1 1 1 0 0 0 0 0
>
```

Tips

もちろん、読み込み時に①row.names=1をつけてもよい。その場合は黒下線部分を②2から1に変更

```
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
>
```

①

②

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
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定し
out_f <- "hoge1.txt"                     #出力ファイル名を指定し
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, 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

```
> #本番          ①   ②   ③        ④                ⑤
> r <- apply(data, 1, cor, y=data.cl, method=param)#$
> r
     gene1      gene2      gene3
 0.9839815  0.8107810 -0.7068671
>
```

Tips：cor, as.numeric

①apply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基本手順。②as.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。③as.numericなしの場合と比較すればよい

```
in_f1 <- "sample16_log.txt"              #入力ファイル名を指定して
in_f2 <- "sample16_cl.txt"               #入力ファイル名を指定して
out_f <- "hoge1.txt"                     #出力ファイル名を指定して
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(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
>
```

①
②
③

Tips：cor, as.numeric

①ではエラーが出ているが、②as.numericをつけたことで、無事gene1とテンプレートパターン間の相関係数rを計算できていることがわかる

```
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(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
① > 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
② > cor(as.numeric(data[1, ]), data.cl, method=param)
[1] 0.9839815
>
```

Tips：データの型

①「互換性のない次元です」と言われているのは、「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」は、Rで「データの型」と呼ばれるものに相当します。②と③が同じ見栄え（型）なのでcor関数の要求を満たしています

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_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(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 ③
> 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 ①
> 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 ②
> 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, ①as.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…。②modeというデータの型を表示する関数もあるが…私は試行錯誤派です

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

#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
① > 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
② > mode(data[1, ])
[1] "list"
>
```

コードの解説：cbind

出力部分。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。尚、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
>
```

①

①

Contents

- 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

②例題2。③このウェブページを用いたDEG検出手順の、一般的な群ごとの反復数指定方法です。G1群は6反復、G2群は5反復

(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
```

③

もちろん①入力ファイルが同一群でまとまっている、という前提

t-test：仮想データ

G1群6サンプル　　G2群5サンプル

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

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。

①

```
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
```

t-test：仮想データ

①例題2の、②コピペ実行結果ファイル（hoge2.txt）を眺める。③gene1のような比較する群間（G1群 vs. 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

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

```
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
```

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

①例題3。②*N* = 10,000遺伝子（行）からなる遺伝子発現行列ファイル（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。コード全体をコピペ

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

①

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。

②

```
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))             #とりあえず結果を返す
    }
    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

教科書p108-116

①例題3。②data.clオブジェクトは、テンプレートパターンのようなもの。③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。「クラス」は、「群」と読み替えてもよい

①

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. G
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

```
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
        retur
    }
    else{
        hoge <-
        return
    }
```

②

③

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
>
```

t-test：乱数

①summary関数は、数値行列を入力として、列ごとの要約統計量を出力。確かに（標準）正規分布乱数になっている。②hist関数を用いて、最初の③1列目のみのヒストグラムも描画して確認している

R Console

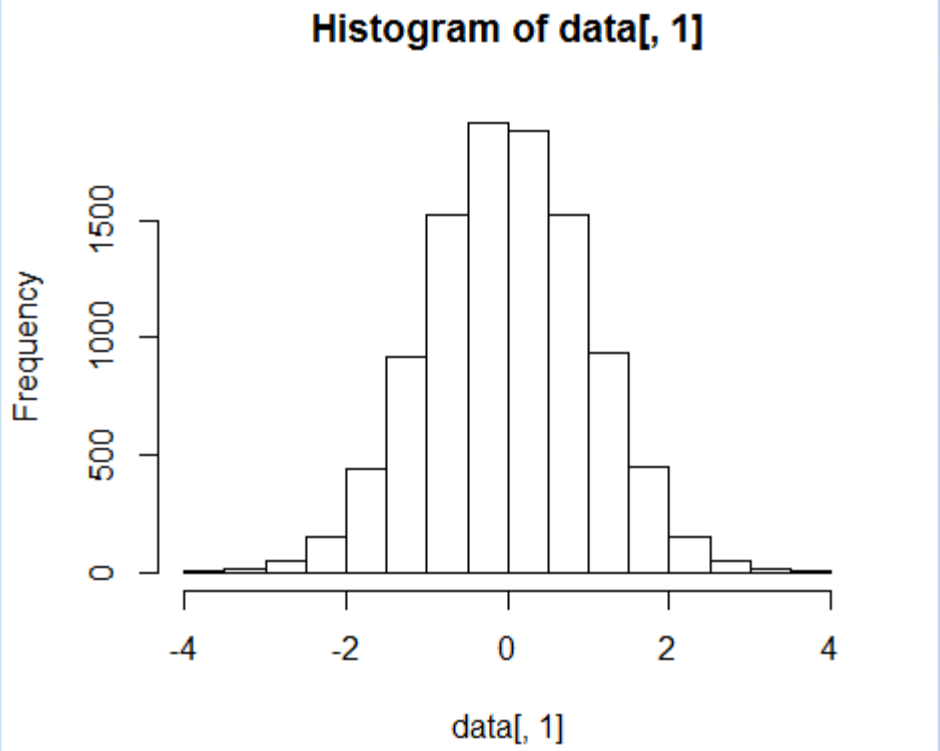
```
> summary(data)
     G1_rep1           G1_rep2           G1_rep3
 Min.   :-3.561788   Min.   :-4.6296   Min.   :-3.97209
 1st Qu.:-0.673850   1st Qu.:-0.6694   1st Qu.:-0.65026
 Median :-0.001270   Median :-0.0038   Median : 0.01834
 Mean   : 0.001152   Mean   :-0.0036   Mean   : 0
 3rd Qu.: 0.679024   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])
>
```

①

②

③

R Graphics: Device 2 (ACTIVE)



ここから先のスライド37までは見るだけでよい。特定の条件を満たす列のみ取り出すやり方

t-test：乱数

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

今ここでやろうとしているのは、①等分散性(var.equal=T)を仮定した②t-test(つまりStudent's t-test)。③p値はここ

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
>
```

②

①

③

t-test：乱数

①このp値情報のみ取り出したい場合は、②のようにすればよい。有意水準 α = 0.05とすると、1行目の遺伝子の*p*値(①0.3030816)は0.05未満ではない。2行目、3行目、…と評価していくわけだが、そもそもどのような手段で$p.valueでよいとわかったのかを次に解説

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
>
```

①

②

Tips：str

str関数を利用すれば、t.test関数に限らず、出力結果（オブジェクト）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。ここでは、①t.test実行結果のhogeオブジェクトに対してstr関数を実行し、抽出可能な情報を眺めている。②例えばp-value情報はここ

R Console

```
> t.test(data[1, data.cl==1], data[1, data
[1] 0.3030816
> hoge <- t.test(data[1, data.cl==1], data[1, data.cl==2], var.equal=T)
> str(hoge)
List of 9
 $ statistic  : Named num 1.18
  ..- attr(*, "names")= chr "t"
 $ parameter  : Named num 4
  ..- attr(*, "names")= chr "df"
 $ p.value    : num 0.303
 $ conf.int   : atomic [1:2] -0.721 1.788
  ..- attr(*, "conf.level")= num 0.95
 $ estimate   : Named num [1:2] 0.269 -0.265
  ..- attr(*, "names")= chr [1:2] "mean of x" "mean of y"
 $ null.value : Named num 0
  ..- attr(*, "names")= chr "difference in means"
 $ alternative: chr "two.sided"
 $ method     : chr " Two Sample t-test"
 $ data.name  : chr "data[1, data.cl == 1] and data[1, data.cl == 2]"
 - attr(*, "class")= chr "htest"
>
```

①

②

t-test：乱数

①まだ例題3の話。②コード下部に移動。$p.valueで情報抽出するテクニックは、③のあたりで利用しています

①

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
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
```

③

②

教科書p108-116

t-test：乱数

①out_fで指定した出力ファイル(hoge3.txt)は、入力ファイル情報の右側に④p.value, q.value, ⑤ranking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
```

②　③　④　⑤　①

t-test：乱数

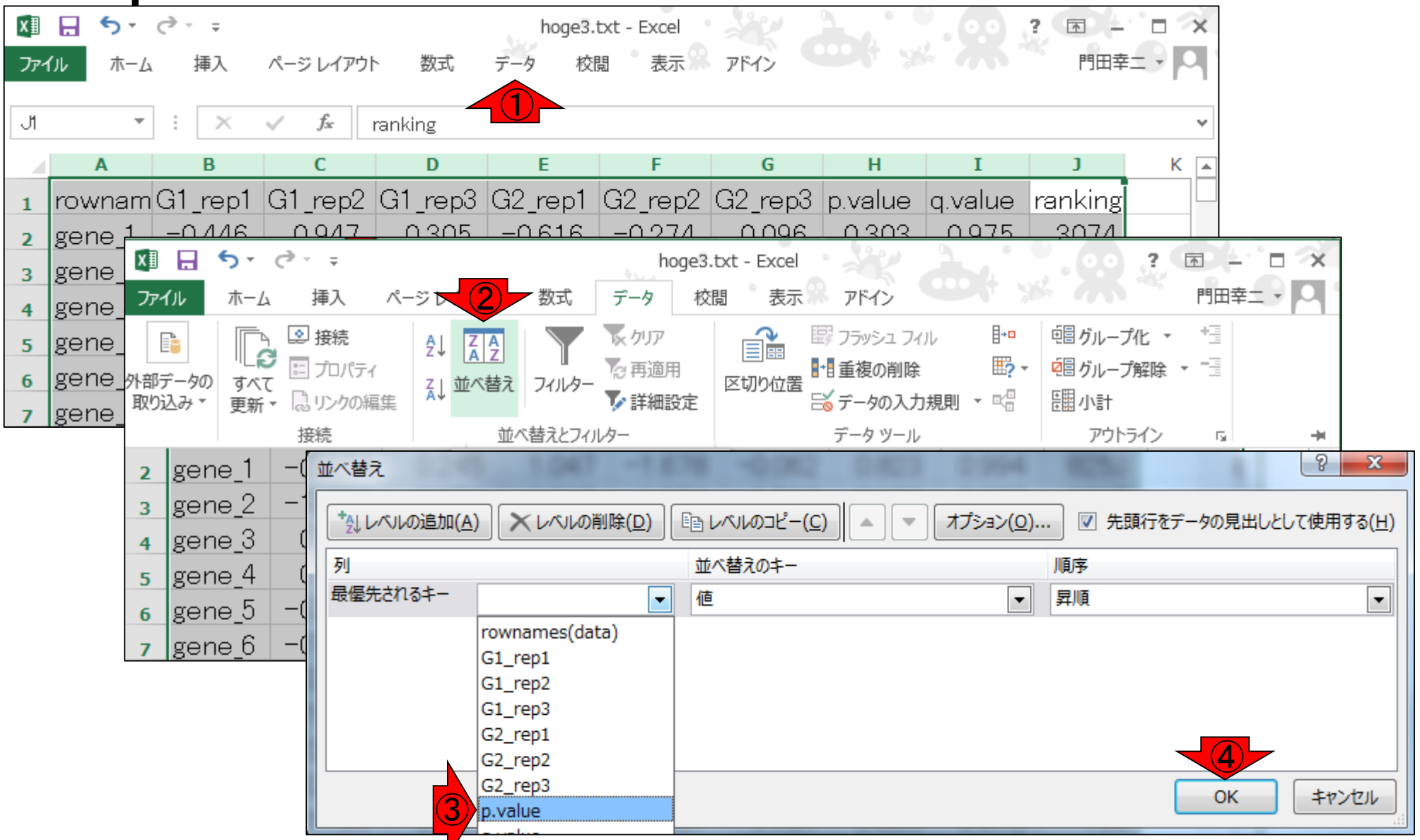
①出力ファイル(hoge3.txt)。④p.value列か⑤ranking列で昇順(ascending order)にソートすると、発現変動順になる

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 0.9750878    3074
gene_2 -1.6776099 -0.0[illegible]
gene_3 -1.2743934  1.8[illegible]
>
```

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

Tips


hoge3.txt - Excel
ファイル ホーム 挿入 ページ レイアウト 数式 データ 校閲 表示 アドイン
門田幸二
①
J1
ranking
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
②
外部データの取り込み
すべて更新
接続
プロパティ
リンクの編集
並べ替え
フィルター
クリア
再適用
詳細設定
区切り位置
フラッシュ フィル
重複の削除
データの入力規則
グループ化
グループ解除
小計
並べ替え
レベルの追加(A)
レベルの削除(D)
レベルのコピー(C)
オプション(O)...
先頭行をデータの見出しとして使用する(H)
列
並べ替えのキー
順序
最優先されるキー
値
昇順
rownames(data)
G1_rep1
G1_rep2
G1_rep3
G2_rep1
G2_rep2
G2_rep3
p.value
③
④
OK
キャンセル

Contents

- 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）

多重比較問題のイントロ

DEGの存在しない、計10,000個の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

教科書p108-116参考

Type-I error（false positive）。一般的な多重比較問題の話を、DEG検出問題に置き換えて説明しているだけ

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。

とばす

- 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。

教科書p108-116 参考

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。ここでは、**このデータセット中に本物のDEGがどれだけあるのか**を正確に見積もりたいと思っていて、p-value情報を頼りに手計算できるが、q-valueという考え方のほうが実用的ですよ、という話。

とばす

多重比較問題：FDRって何?

①*p*–valueの感覚が分かることが最も重要。あとは②実例で*q*–valueの意味を考えればよい

読むだけ

- ***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

①

- ***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

②

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

多重比較問題：FDRって何?

- 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

DEG数に関するよりよい結果を得たい場合には、①***p***-valueではなく②***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。)

読むだけ

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

① ②

DEG検出結果の解釈

これはDEGが存在しないnon-DEGのみからなるデータ。①*p*-value < 0.05を満たす遺伝子数は492個。*q*-valueで考えると、「上位492個をDEGと判定した場合に、その中に占める偽物(non-DEG)の割合は97.51%」

rownames(c	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	p.value		
gene_2313	0.172	0.422	0.234	−1.211	−1.399	−1.564	0.0002		
gene_7754	1.514	1.426	1.814	−0.324	−0.615	−0.275	0.0002		
gene_1175	−1.437	−1.029	−1.326	1.238	1.336	0.780	0.0003		
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

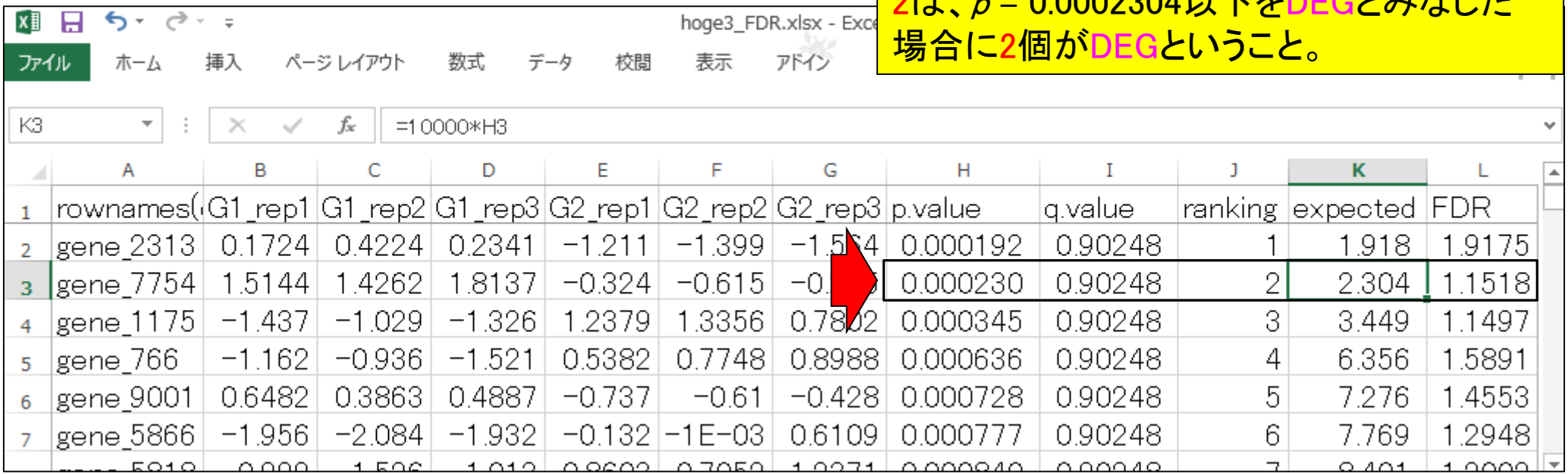
①

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

参考

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

hoge3_FDR.xlsx - Excel

ファイル ホーム 挿入 ページ レイアウト 数式 データ 校閲 表示 アドイン 門田幸二

N1 0004

	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
>
```

• 解析 | 発現変動 | 2群間 | 対応なし | Student's t-test 教科書p115-116 参考

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

①の赤枠部分で指定しているのは
p-valueの閾値（つまり有意水準）

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
```

①

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
> 
```

①

Contents

- 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

①例題4。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。

①

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。**DEGが存在するデータから正しくDEGがあると判定できていることを伝えたい**

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な
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
(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
out_f <- "hoge4.txt"                      #出力ファイル名を指定し
param_G1 <- 3                             #G1群のサンプル数を指定
param_G2 <- 3

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

#等分散性を仮定 (
Students_ttest <
    x.class1 <-
    x.class2 <-
    if((sd(x.cla
```

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

- 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)

②limma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 | 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 2015/02/12)
- 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) ① modified 2015/05/24)
- 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
- 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(la
- 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
- 解析 | 発現変動 | 2群間 | 対応あり | SAM (Tusher

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

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data_rma_2_LIV.txtの場合:

31,099 probesets×8 samples(G1群4サンプル vs. G2群4サンプル)のデータです。

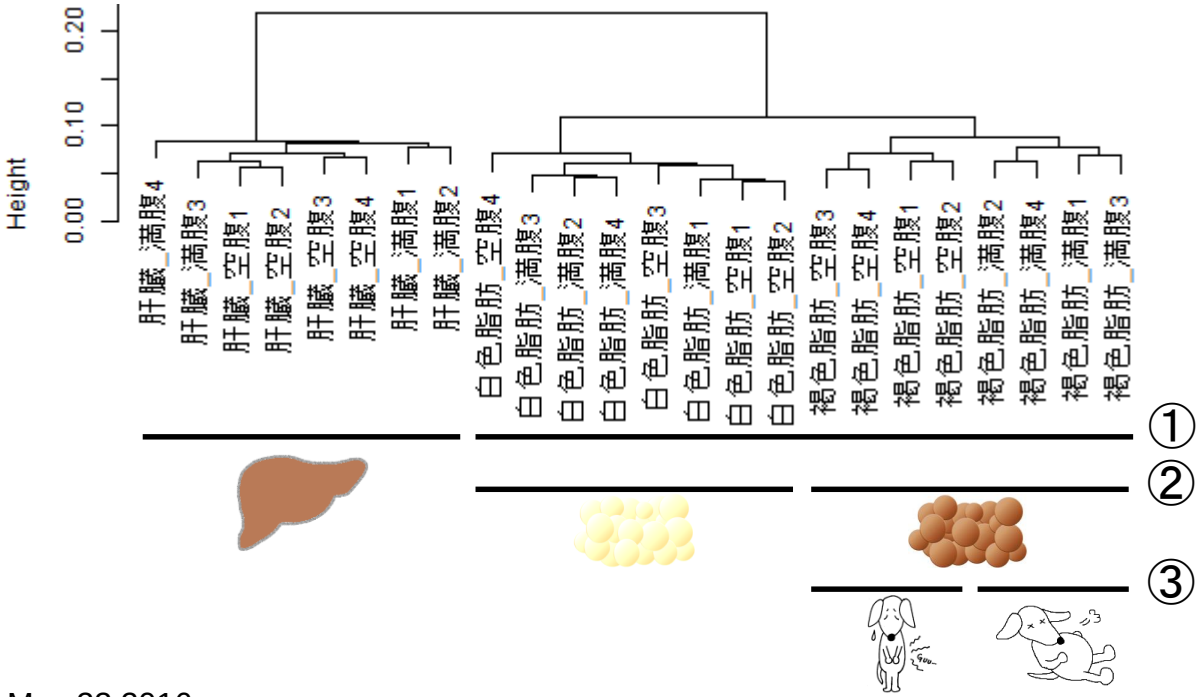
```
in_f <- "data_rma_2_LIV.txt"              #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                      #出力ファイル名を指定してout_fに格納
param_G1 <- 4                             #G1群のサンプル数を指定
param_G2 <- 4                             #G2群のサンプル数を指定

#必要なパッケージのロード
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
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

#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
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
```

② (library(limma) の行)

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
  - Rat 24 samples：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 samples、Liver (肝臓; LIV)8 samples
    - BAT 8 samples：通常(BAT_fed) 4 samples vs. 24時間絶食(BAT_fas) 4 samples
    - WAT 8 samples：通常(WAT_fed) 4 samples vs. 24時間絶食(WAT_fas) 4 samples
    - LIV 8 samples：通常(LIV_fed) 4 samples vs. 24時間絶食(LIV_fas) 4 samples



rcode_clustering_png.txtの実行結果。
①肝臓と脂肪間で大きく2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。

• 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)

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。①解析1用。テンプレートからの変更点および追加点。②Mac userは区切り文字(delimiter)に注意。作業ディレクトリは③ココ

1. サンプルデータ20の data_rma_2_LIV.txt

31,099 probesets×8 samples(G1群4サンプル

```
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)
```

①

```
##########################
###  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
```

②

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

#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
```

R Console

```
> getwd()
[1] "C:/Users/kadota/Desktop/hoge/GSE7623"
> list.files(pattern="EN")
[1] "data_mas_EN.txt" "data_rma_EN.txt"
[3] "data_rob_EN.txt"
>
```

③

rcode_limma_basic.txt

入力ファイル（data_mas_EN.txt）読み込み後の①dataオブジェクトは24サンプルからなる。一緒に黒枠内をコピペしてついてきてもよいが、眺めるだけでもよい

```
############################
###  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

#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="
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1

#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
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
design <- model.matrix(~data.cl)             #デザイン行列を作
fit <- lmFit(data, design)                   #モデル構築(ばら
out <- eBayes(fit)                           #検定(経験ベイズ
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でランキ
sum(q.value < 0.05)                          #FDR < 0.05を満た

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

①

R Console

```
> ############################
> ###  解析1 (Analysis1)  ###
> ############################
> in_f <- "data_mas_EN.txt"                    #入力ファ$
> out_f <- "hoge1.txt"                         #出力ファ$
> param_G1 <- 2                                #G1群の$
> param_G2 <- 2                                #G2群の$
>
> #必要なパッケージをロード
> library(limma)                               #パッケ$
>
> #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.name$
> data.cl <- c(rep(1, param_G1), rep(2, param_G2$
> 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

①サブセット抽出後のdataオブジェクトを確認。②posi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="¥
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1,

#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
data <- data[,posi]                    #サブセットを抽出

#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(ばら
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
ranking <- rank(p.value)               #p.valueでランキン
sum(q.value < 0.05)                    #FDR < 0.05を満た

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

①

②

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
> |
```

発現変動解析(limma)

①data.clで指定している情報は、クラスラベル情報（class label；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
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

①

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
> |
```

①rcode_limma_basic.txtを一通りコピペ実行した後、②様々なFDR閾値を満たす遺伝子数を調査した結果。③FDR 65%（65%の偽物混入を許容すれば、という意味）というカナリ緩めの閾値のところで167遺伝子が得られる。これは167×0.65 = 108.55個が偽物、残りの167×(1 − 0.65) = 58.45個が本物のDEGだということ

rcode_limma_basic.tx

①

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

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

#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="
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1

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

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

③

R Console

```
[1] 0
>
> #ファイルに保存
> tmp <- cbind(rownames(data), data, p.value, q.$
> write.table(tmp, out_f, sep="\t", append=F, qu$
> sum(q.value < 0.05)
[1] 0
> sum(q.value < 0.10)
[1] 0
> sum(q.value < 0.30)
[1] 0
> sum(q.value < 0.50)
[1] 0
> sum(q.value < 0.70)
[1] 330
> sum(q.value < 0.60)
[1] 0
> sum(q.value < 0.65)
[1] 167
>
```

②

①rcode_limma_basic.txtで動作確認をしてから、②param_posiのように変更予定箇所を上のほうに移動して、③一気に解析2-4用のコードを実行するためのファイルを作成する（のが門田流）

rcode_limma_basic.txt

```
############################
###  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"
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
```

発現変動解析（教科書p167-）

①rcode_limma_all.txt。②解析2用のコード実行部分をコピペ

	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

①rcode_limma_all.txt（の一部）

②

```
#############################↓
###  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（教科書p167-）

通常（私）は、①0.05～0.50あたりのFDR閾値を概観。この場合はDEGがあると判断します。論文で議論するのは、FDR = 0.05や0.10が一般的

	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"
out_f <- "hoge2.txt"
param_G1 <- 2
param_G2 <- 2
param_posi <- c

#必要なパッケー
library(limma)

#入力ファイルの
data <- read.tab
data.cl <- c(re
data <- data[,p
colnames(data)

#本番
design <- model
fit <- lmFit(da
out <- eBayes(f
p.value <- out$
q.value <- p.ad
ranking <- rank
sum(q.value < 0
sum(q.value < 0
sum(q.value < 0
sum(q.value < 0

#ファイルに保存
tmp <- cbind(ro
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定し
```

```
#入力ファイル名を指定してin_fに格納
#出力ファイル名を指定してout_fに格納
#G1群のサンプル数を指定
#G2群のサンプル数を指定
```

R Console

```
> #本番
> design <- model.matrix(~data.cl)          #デザイ$
> fit <- lmFit(data, design)                #モデル$
> out <- eBayes(fit)                        #検定($
> p.value <- out$p.value[,ncol(design)]     #p値をp$
> q.value <- p.adjust(p.value, method="BH") #q値$
> ranking <- rank(p.value)                  #p.valu$
> sum(q.value < 0.05)                       #FDR < $
[1] 0
> sum(q.value < 0.10)                       #FDR < $
[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, q.$
> write.table(tmp, out_f, sep="\t", append=F, qu$
>
```

①

発現変動解析

教科書p167

例えば、①q < 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

	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"
out_f <- "hoge2.txt"
param_G1 <- 2
param_G2 <- 2
param_posi <- c
↓
#必要なパッケー
library(limma)
↓
#入力ファイルの
data <- read.tab
data.cl <- c(re
data <- data[,pa
colnames(data)
↓
#本番
design <- model
fit <- lmFit(da
out <- eBayes(f
p.value <- out$
q.value <- p.ad
ranking <- rank
sum(q.value < 0
sum(q.value < 0
sum(q.value < 0
sum(q.value < 0
↓
#ファイルに保存
tmp <- cbind(ro
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身を指定し
```

```
#入力ファイル名を指定してin_fに格納
#出力ファイル名を指定してout_fに格納
#G1群のサンプル数を指定
#G2群のサンプル数を指定
```

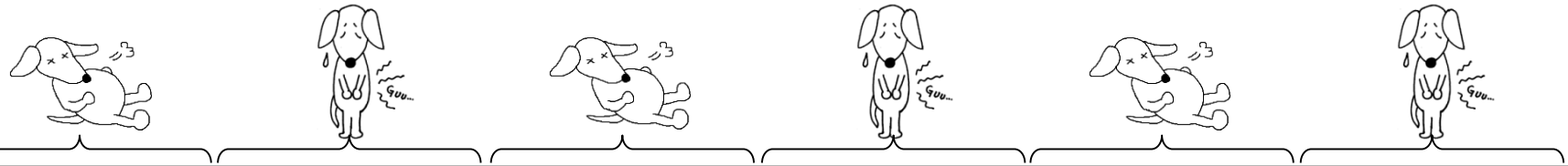
R Console

```
> #本番
> design <- model.matrix(~data.cl)          #デザイ$
> fit <- lmFit(data, design)                #モデル$
> out <- eBayes(fit)                        #検定($
> p.value <- out$p.value[,ncol(design)]     #p値をp$
> q.value <- p.adjust(p.value, method="BH") #q値$
> ranking <- rank(p.value)                  #p.valu$
> sum(q.value < 0.05)                       #FDR < $
[1] 0
> sum(q.value < 0.10)                       #FDR < $
[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, q.$
> write.table(tmp, out_f, sep="\t", append=F, qu$
>
```

①

発現変動解析(limma)

解析3のコードをコピペした結果。①DEGがあると判断します



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2
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3の予想：DEGあり

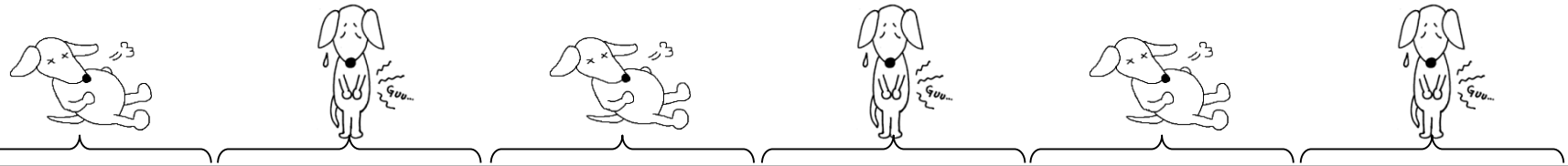
R Console

```
> colnames(data)                                    #サブセ$
[1] "BAT_fed1" "BAT_fed2" "WAT_fed1" "WAT_fed2"
>
> #本番
> design <- model.matrix(~data.cl)                  #デザイ$
> fit <- lmFit(data, design)                        #モデル$
> out <- eBayes(fit)                                #検定($
> p.value <- out$p.value[,ncol(design)]             #p値をp$
> q.value <- p.adjust(p.value, method="BH")#q値$
> ranking <- rank(p.value)                          #p.valu$
> 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
>
```

①

発現変動解析(limma)

解析4のコードをコピペした結果。①DEGがあると判断します



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2
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4の予想：DEGあり

R Console

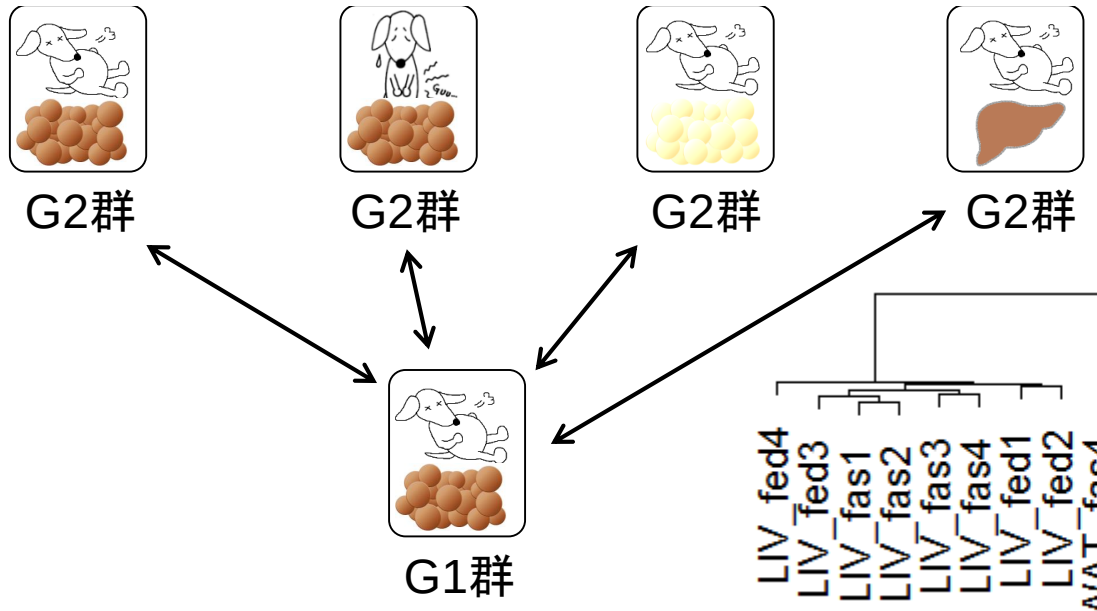
```
> colnames(data)                                  #サブセ$
[1] "BAT_fed1" "BAT_fed2" "LIV_fed1" "LIV_fed2"
>
> #本番
> design <- model.matrix(~data.cl)                #デザイ$
> fit <- lmFit(data, design)                      #モデル$
> out <- eBayes(fit)                              #検定($
> p.value <- out$p.value[,ncol(design)]           #p値をp$
> q.value <- p.adjust(p.value, method="BH")#q値$
> ranking <- rank(p.value)                        #p.valu$
> 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
>
```

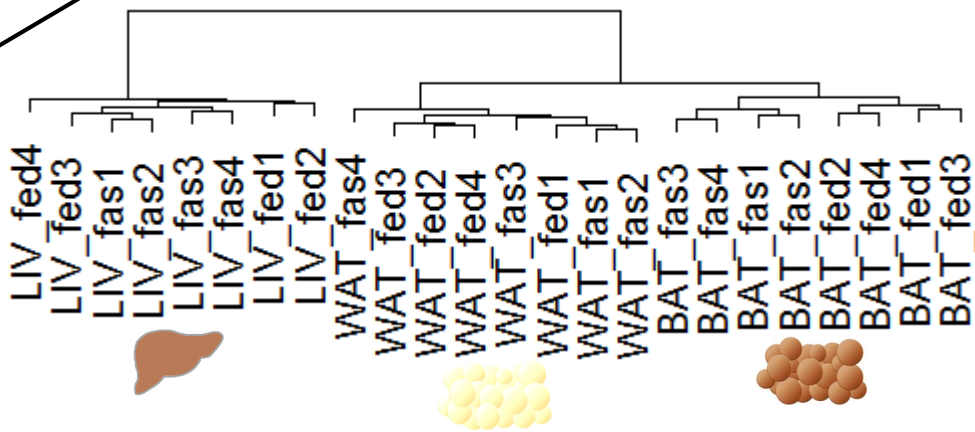
①

解析結果まとめ(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

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

- 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

①例題4、②例題5、③例題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

①

4. サンプルデータ23のsample2 txtの場合:

最初の3サンプルがG1群、残
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.equa
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
    }
```

②

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 <- 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
```

③

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
```

Tips：関数の利用法

①例題5。②で提供している様々な関数群を、③source関数で読み込んで利用可能

①

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")
```

②

③

```
#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 <-
```

②

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
    }
}
```

Tips：関数の利用法

①例題6。ネット接続環境でなくても、一旦②R_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、③のような書き方で読み込んで、④Students_ttest関数を利用可能

①

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
```

③

②

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

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

④

```
################################
###  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(①FCROSなど)も多数あり。次に紹介するのは、limmaと似た統計的手法の②IBMT法

- 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 20
- 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) (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 2015/02/12)
- 解析 | 発現変動 | 2群間 | 対応なし | empirical Bayes (Smyth_2004) (last modified 2015/05/24) NEW
- 解析 | 発現変動 | 2群間 | 対応なし | samroc (Broberg
- 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 | 一元配置分散分
- 解析 | 発現変動 | 3群間 | 対応なし | Kruskal-Wallis(

②

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.

①

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.

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

多数の方法があります

IBMT法は、①limma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②関数部分のみ提供しているという解釈でもよい。③sourceで読み込んでいるおかげで、④IBMTという関数を利用可能

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

eh3.uc.edu

②

```
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
limma, or at least have attributes named sigma, A
stdev.unscaled. ## "testcol" is an integer or vec
mdata$coefficients for which the function is to
augmented form of "mdata" (the input), with th
t-value for IBMT ## IBMT.p - P-value for IBMT
for IBMT ## IBMT.priorvar- prior variance for IBM
for IBMT ## ## Example Function Call: ## IBMT.r
further help on implementing function, contact sa
##############################################
library("stats") library("limma") logVAR<-log(mda
numgenes<-length(logVAR[df>0]) df[df==0]<-NA
egpred<-loessFit(eg,mdata$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
```

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
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$Amean<-rowMeans(data)                 #おまじない
fit <- IBMT(fit,2)                        #IBMTプログラムの実行
p.valu  fit$IBMT.p                        #p値をp.valuelに格納
```

①

③

②

④

参考

発現変動解析(IBMT)

IBMT法はlimmaと似ているので、全体的な傾向は変わりません。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

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						

参考

rcode_ibmt_basic.txt。①解析1用。テンプレートからの変更点および追加点。②Macのヒトは区切り文字に注意。コピペ実行はやらなくてよい

発現変動解析(IBMT)

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
```

①

```
#############################
###  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
```

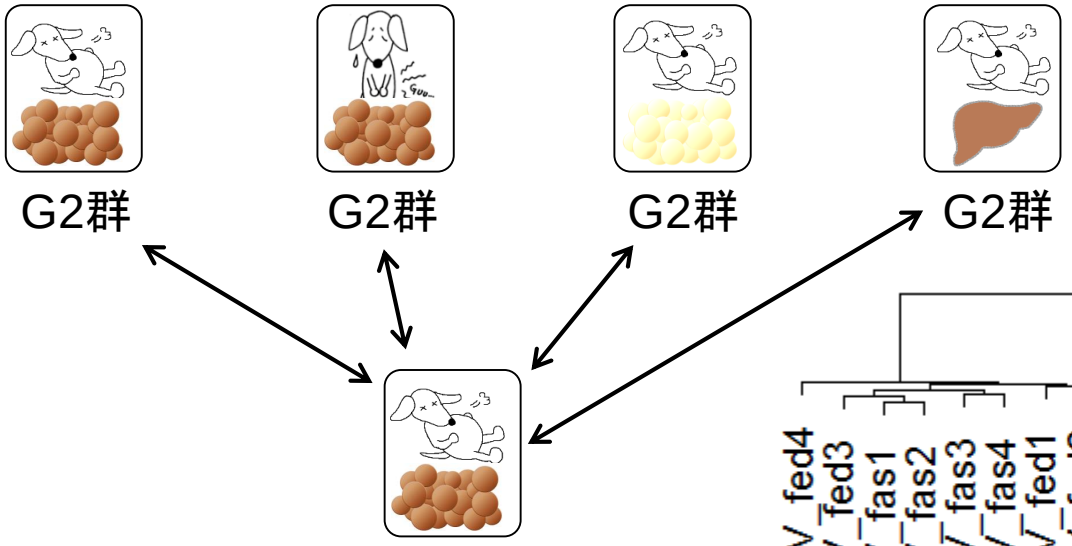
②

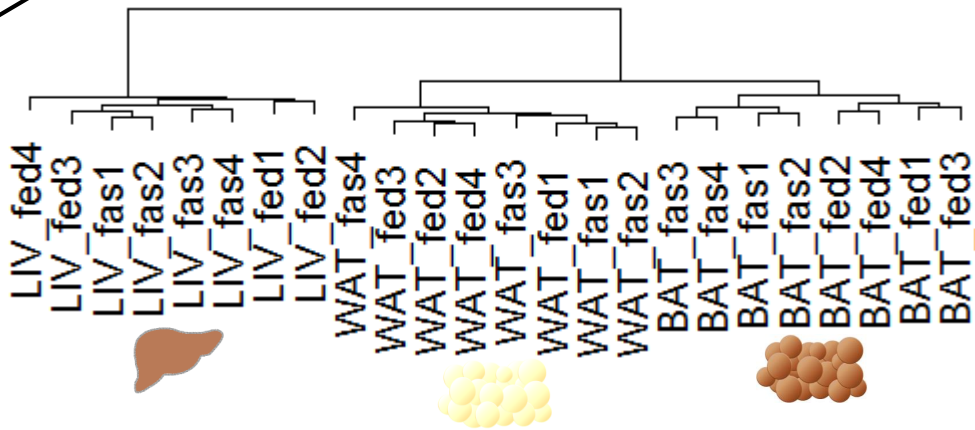
```
#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#in_fで指定
data.cl <- c(rep(1, param_G1), rep(2, param_G2))#G1群を1、G2群を2としたベクトルdat

#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
```

解析結果まとめ(IBMT)

①rcode_ibmt_all.txtのコピペ実行結果をまとめたもの。入力は、MAS5前処理法を実行して得られたデータ(MAS5データ)。傾向としてはlimmaと同じことがわかる

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





解析結果(IBMT)

①rcode_ibmt_all.txtの入力ファイル名部分を変えて、RMAとRobLoxBioCデータについて実行した結果。同じDEG検出法でも、前処理法（MAS5, RMA, RobLoxBioC）次第で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

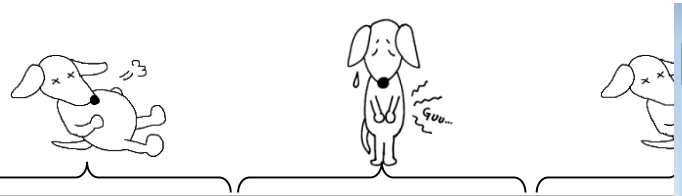
遺伝子数	解析1 G1群：BAT_fed G2群：BAT_fed	解析2 G1群：BAT_fed G2群：BAT_fas	解析3 G1群：BAT_fed G2群：WAT_fed	解析4 G1群：BAT_fed G2群：LIV_fed
FDR < 0.05	0	1832	1025	8438
FDR < 0.10	0	3654	3234	10869
FDR < 0.30	0	9618	11097	16288
FDR < 0.50	0	14731	16576	20145

Contents

- 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）

M-A plot

limma解析結果（①解析4の、②FDRが0.05を満たす2,892遺伝子）のM-A plotを描画する。M-A plotは、2群間比較時に大抵デフォルトとして作成される図



	BAT_fed1	BAT_fed2	BAT_fed3	BAT_fed4	BAT_fas1	BAT_fas2	BAT_fas3	BAT_fas4	WAT_fed1	WAT_fed2
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								

R Console

```
> colnames(data)                                  #サブセ$
[1] "BAT_fed1" "BAT_fed2" "LIV_fed1" "LIV_fed2"
>
> #本番
> design <- model.matrix(~data.cl)                #デザイ$
> fit <- lmFit(data, design)                      #モデル$
> out <- eBayes(fit)                              #検定($
> p.value <- out$p.value[,ncol(design)]           #p値をp$
> q.value <- p.adjust(p.value, method="BH")#q値$
> ranking <- rank(p.value)                        #p.valu$
> 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
>
```

M-A plot

rcode_limma_MAplot_basic.txtは、②例題7をベースに作成しています。最終形は③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」

1. サンプルデ
場合:

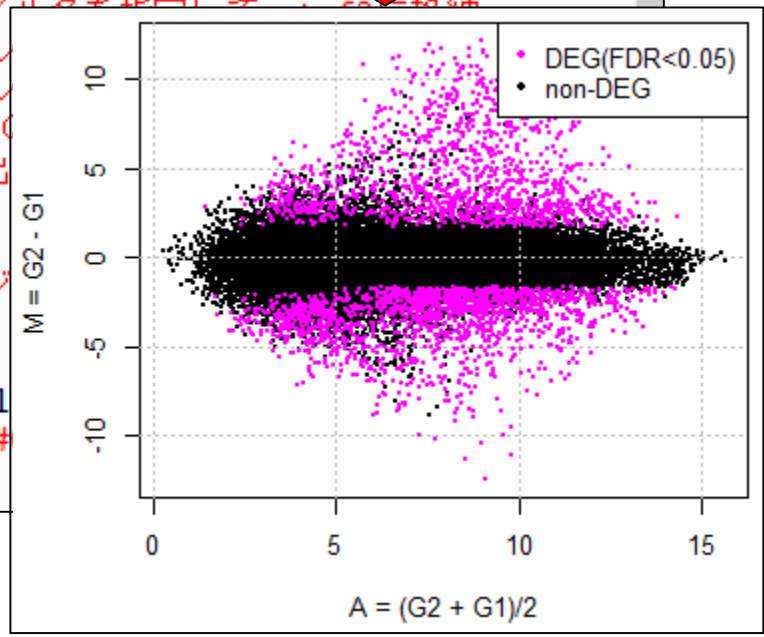
```
in_f <-
out_f <-
param_G1
param_G2

#必要なパ
library(l

#入力ファ
data <- r
data.cl <

#本番
#design <
design <-
```

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"                 #出力ファイル名を指定してout_f2に格納
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))#
```

③



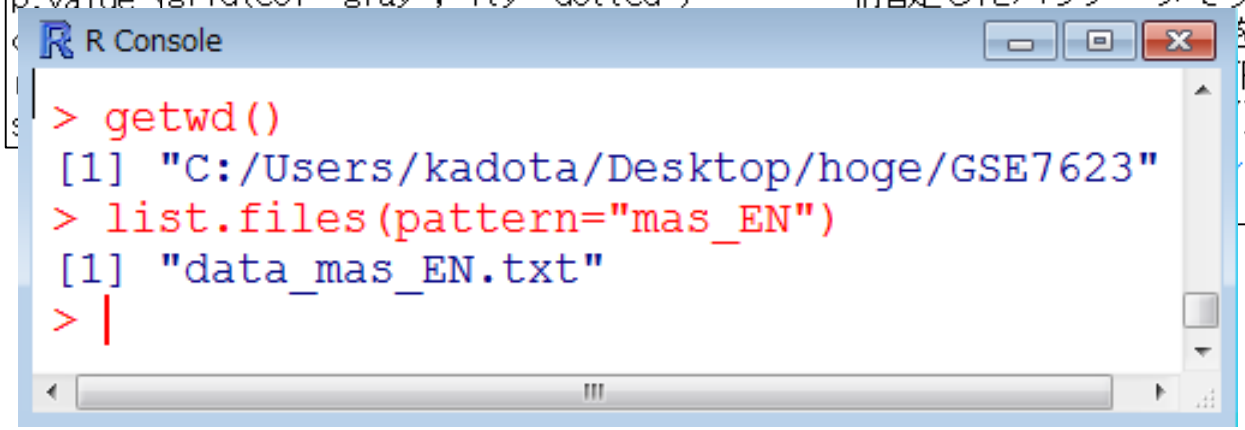
①M-A plotの説明。②横軸のAは平均発現レベル、③縦軸のMは$\log_2$(G2/G1)に相当

```
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定して
out_f1 <- "hoge1.txt"                    #出力ファイル名を指定して
out_f2 <- "hoge1.png"                    #出力ファイル名を指定して
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 <-
data.cl
data <-
colnames

#本番(DE
design <-
fit <- l
out <- e
p.value
```

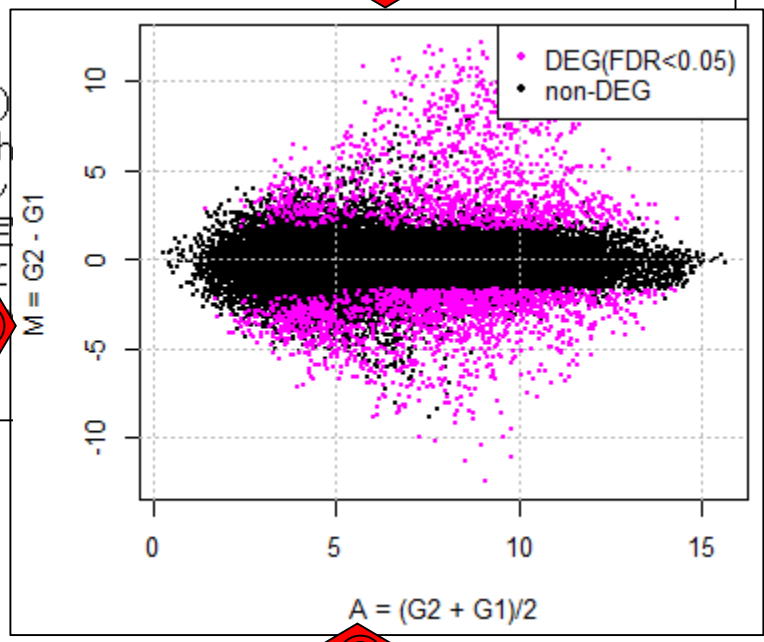
①

```
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
```

を判定
RUE と
),



③



②

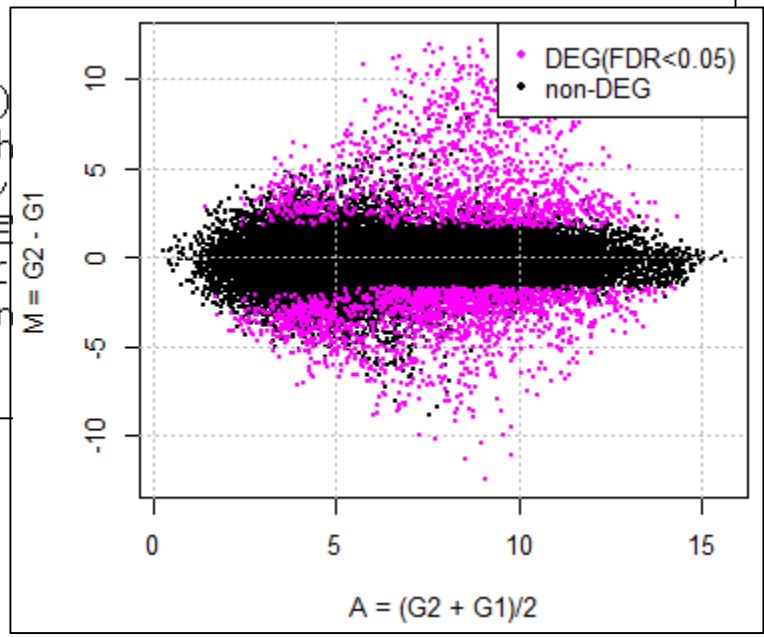
```
in_f <- "data_mas_EN.txt"                #入力ファイル名を指定
out_f1 <- "hoge1.txt"                    #出力ファイル名を指定
out_f2 <- "hoge1.png"                    #出力ファイル名を指定
param_G1 <- 2                            #G1群のサンプル数を指
param_G2 <- 2                            #G2群のサンプル数を指
param_posi <- c(1,2,17,18)               #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
data <-
data.cl
data <-
colnames

#本番(DE
design <-
fit <- l
out <- e
p.value
```

M値とA値の元となる情報は、mean_G1とmean_G2。これらは単純に群ごとの(対数変換後の)シグナル強度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
```

①

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



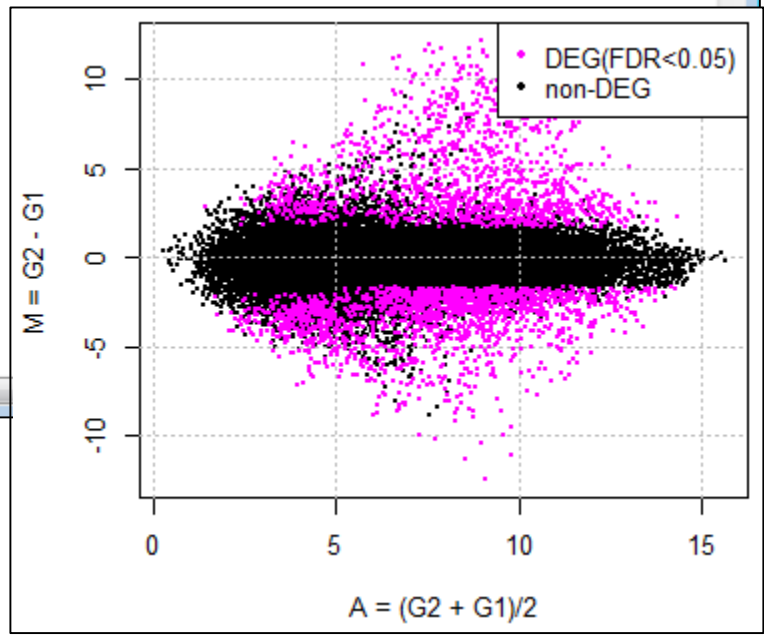
参考

計算例

```
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
```

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

参考

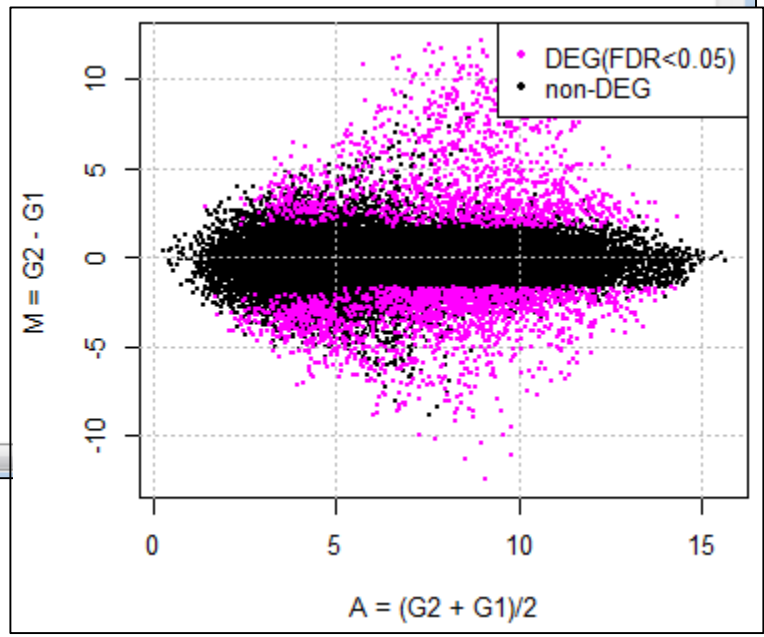
①(mean_G2 − mean_G1)の計算結果を縦軸のMとして計算できるのは、対数変換後のデータだから

```
mean_G1 <- apply(as.matrix(data[,data.cl==1]), 1, mean)#遺伝子ごとにG1群の平均を計
mean_G2 <- apply(as.matrix(data[,data.cl==2]), 1, mean)#遺伝子ごとにG2群の平均を計
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
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
>
```

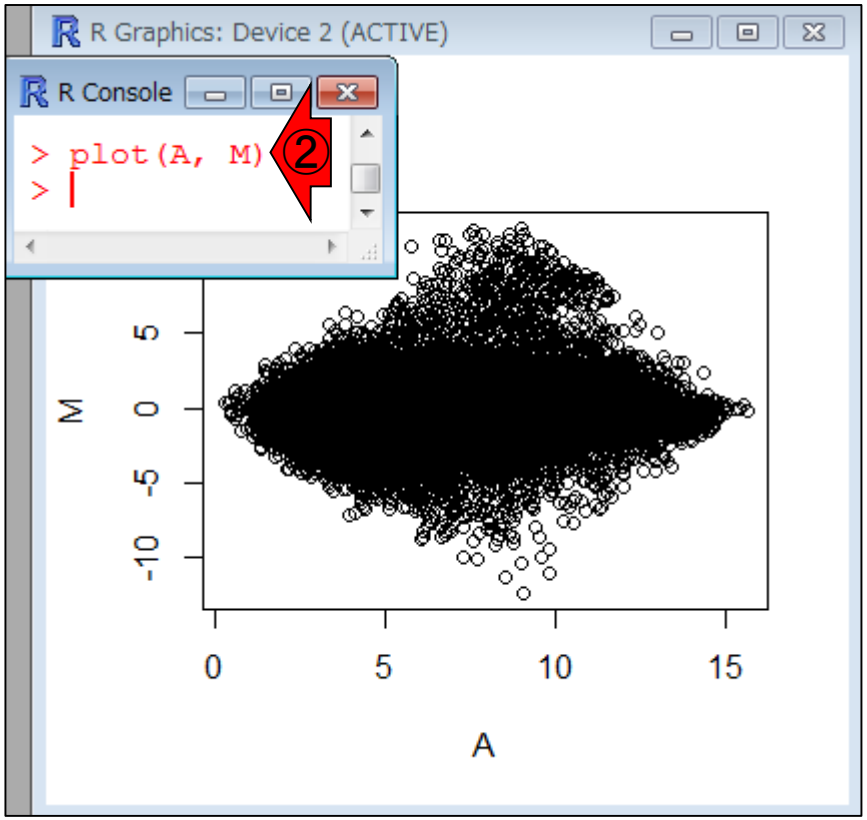
①

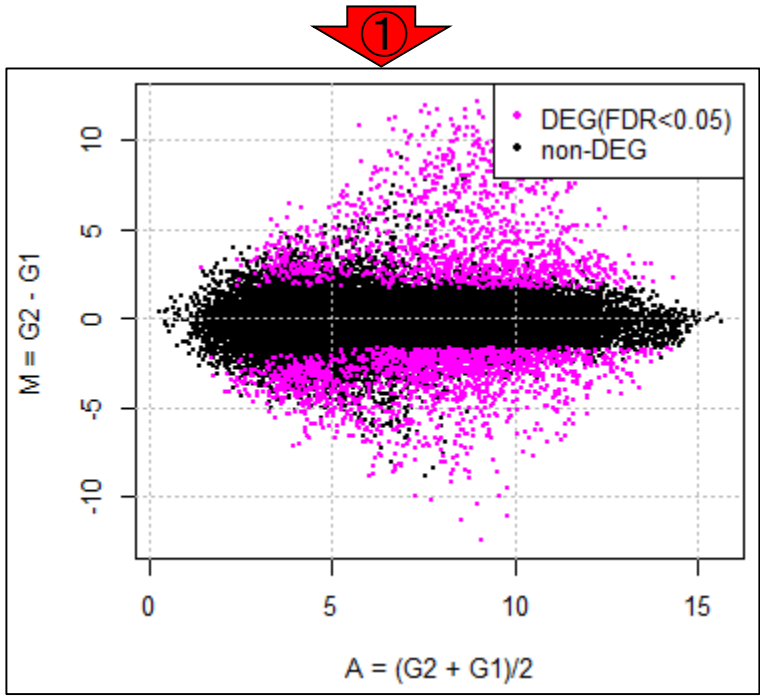


横軸のAは平均発現レベル、
縦軸のMは$\log_2$(G2/G1)に相当

最終的には①のような書き方になっているが、plotの基本形は②plot(A, M)

```
#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↓
```

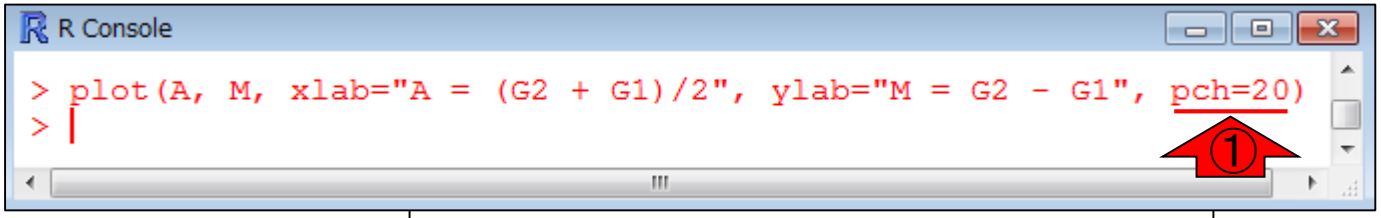


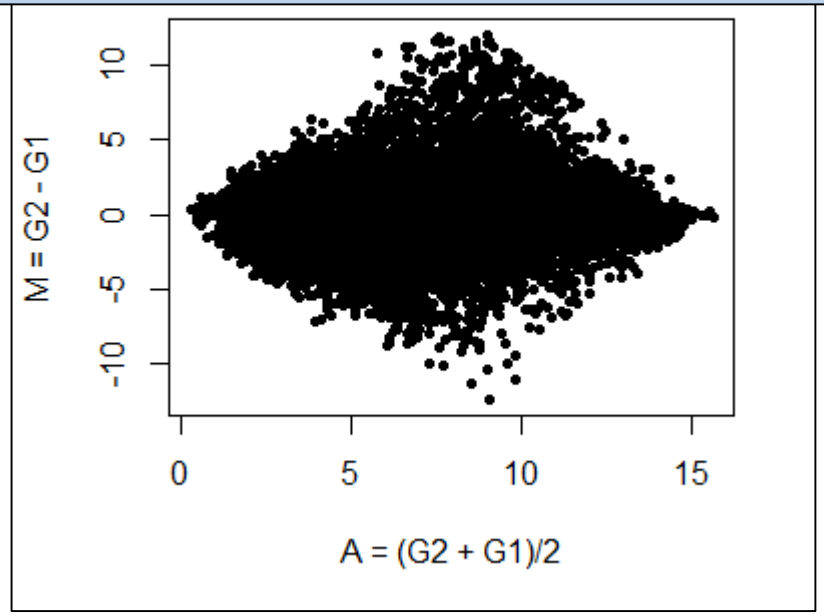


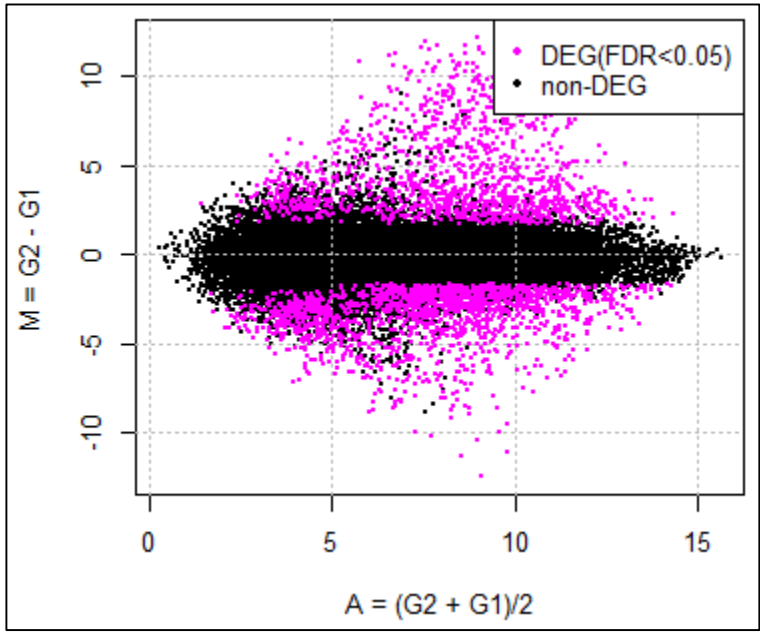
黒丸の塗りつぶしにすべく、①pch=20オプションを追加

```
#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↓
```

①







プロットの大きさをデフォルトの10%にすべく、cex=.1を追加

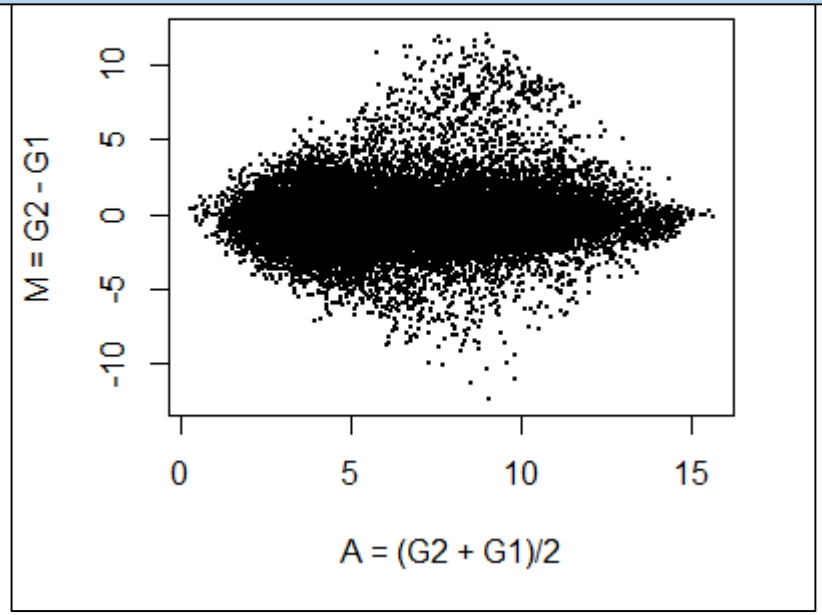
```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パラメータを指定↓
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)#M-A plotを描画↓
grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを表示↓
obj <- as.logical(q.value < param_FDR) #条件を満たす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↓
```

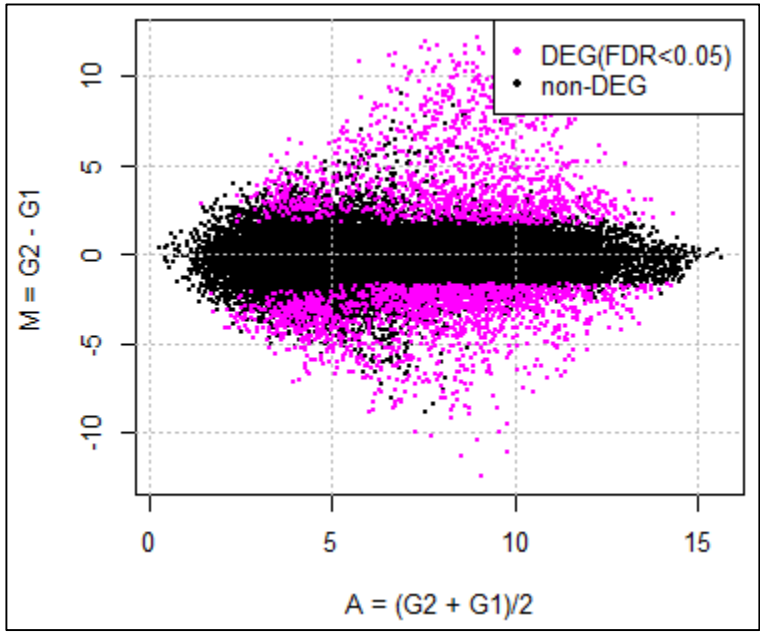
①

R Console

```
> plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1, pch=20)
>
```

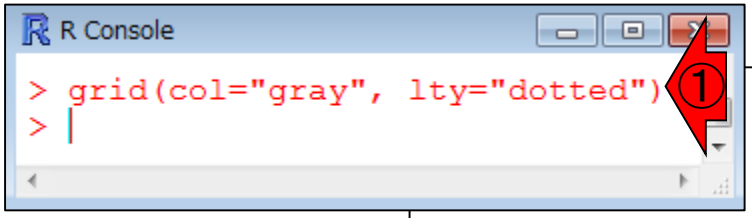
①

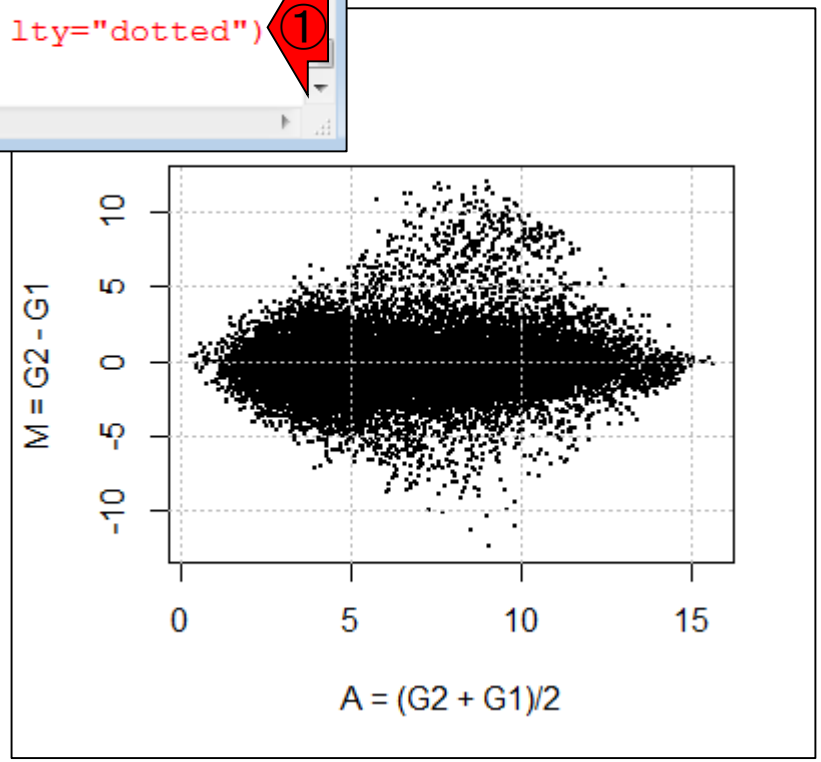


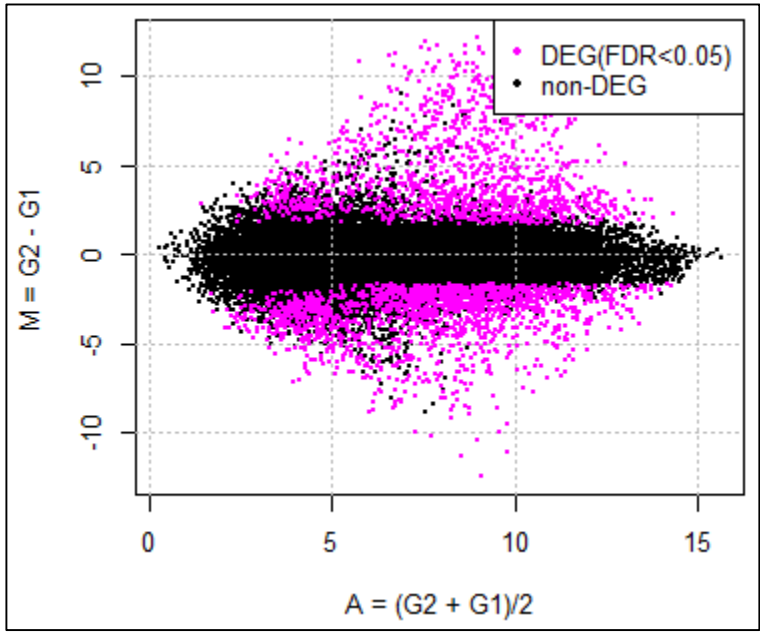


①グリッド線を追加

```
#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↓
```

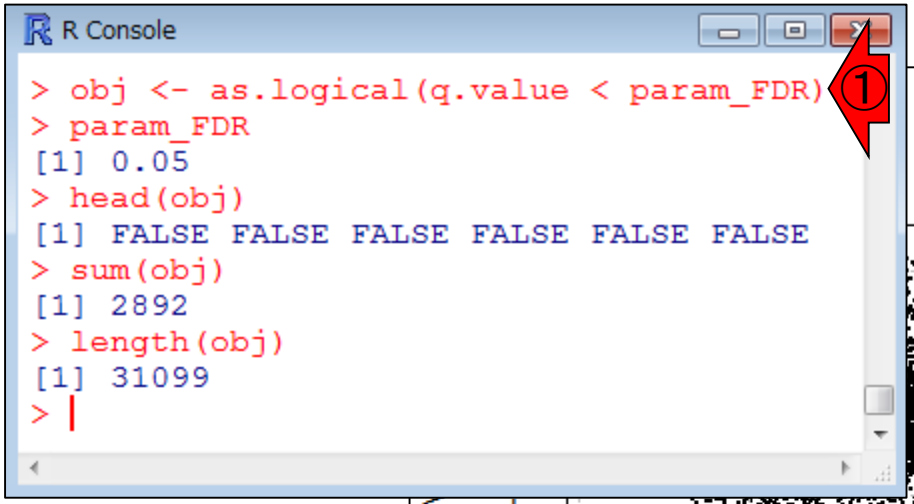


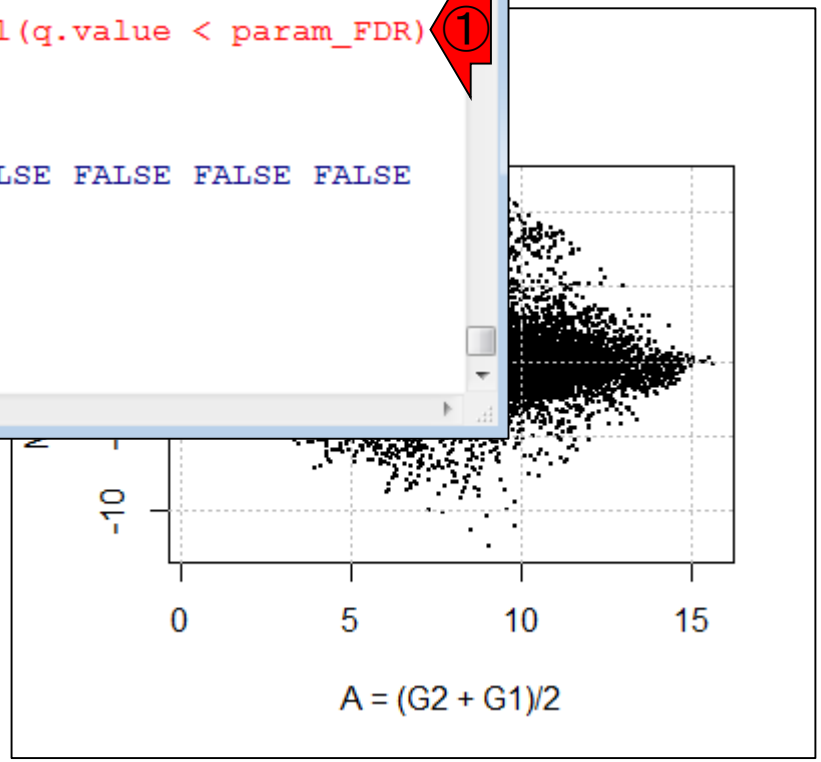


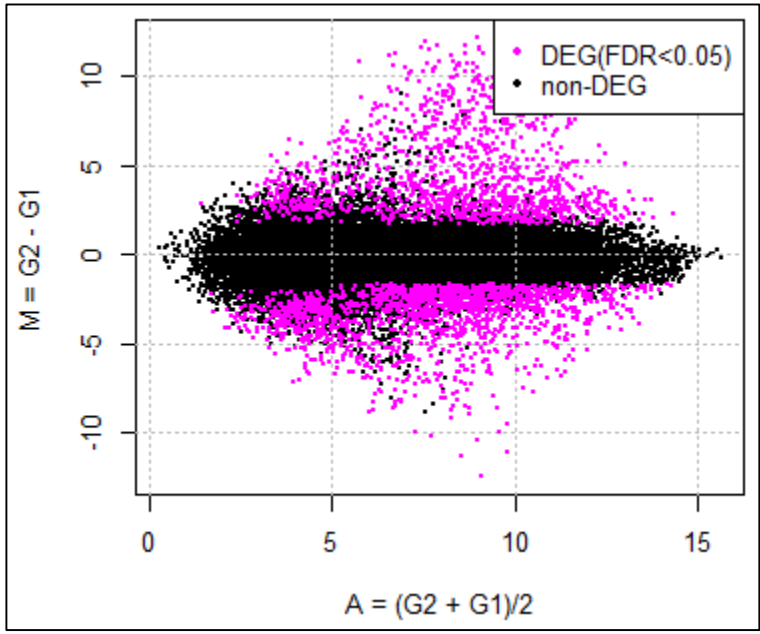


①指定したFDR閾値を満たすDEGの位置情報を、論理値（TRUE or FALSE）ベクトルobjとして取得

```
#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)  ①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↓
```

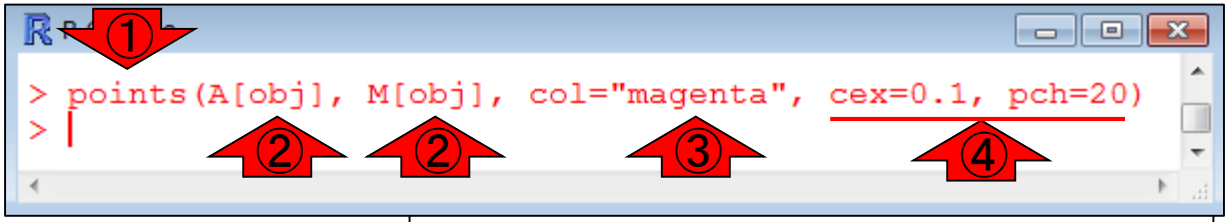


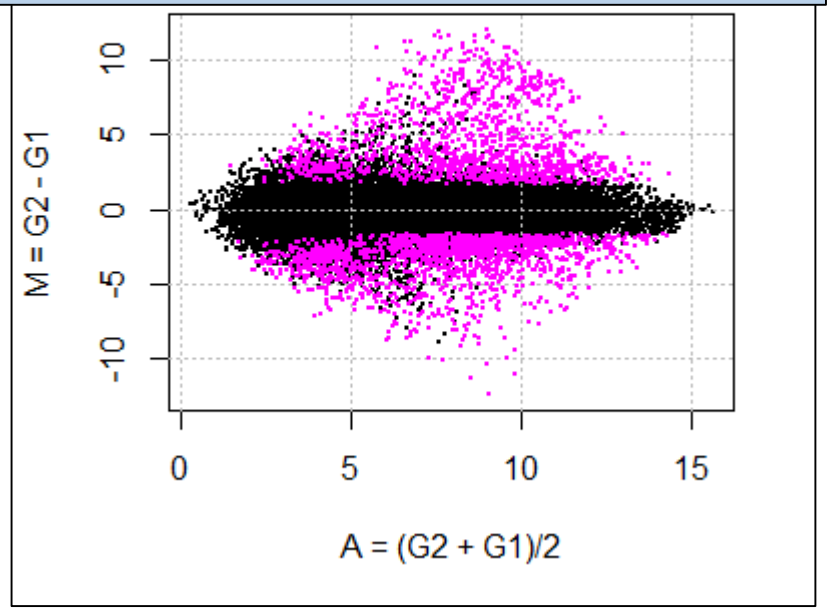


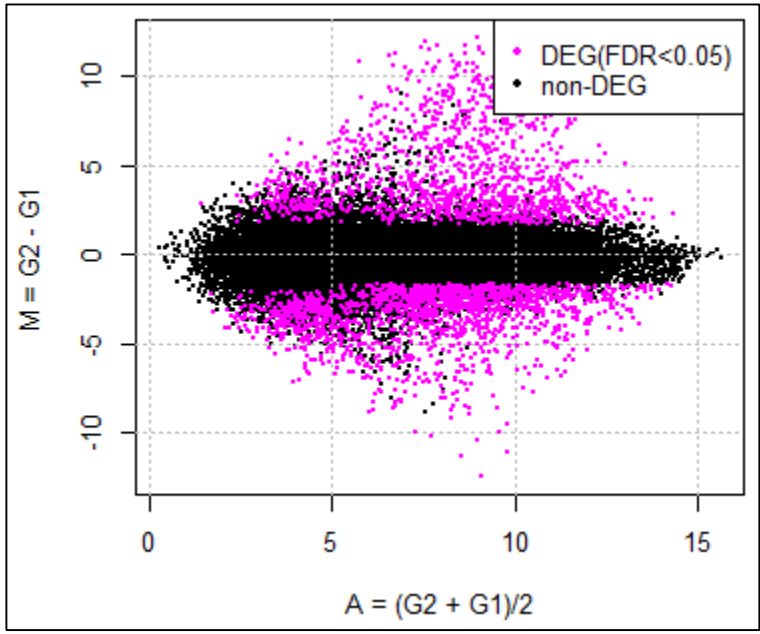


①points関数で、②objがTRUEの場所を③magenta色で描画。④cexとpchオプションの値を同じにすることで色だけを変更していることに相当する

```
#ファイルに保存(M-A plot)↓
png(out_f2, pointsize=13, width=param_fig[1], height=pa
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↓
①
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↓
```







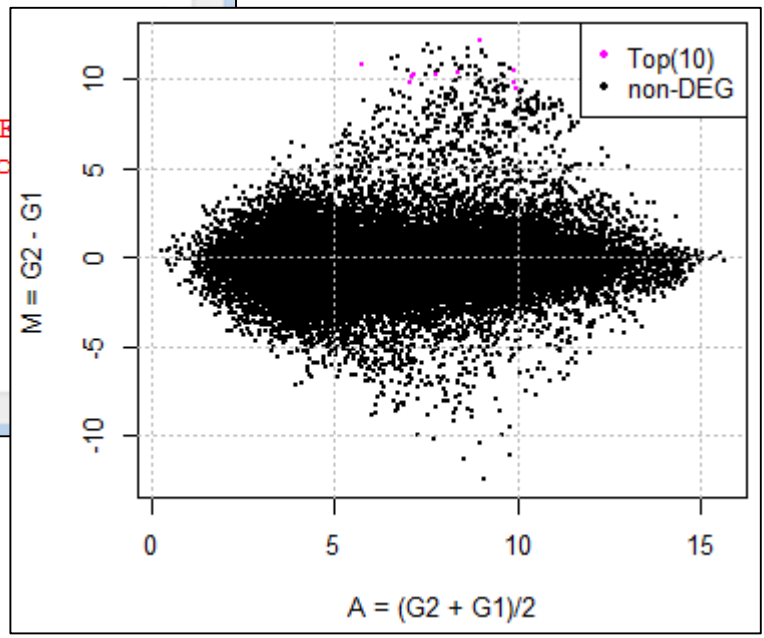
参考

①rcode_limma_MAplot_basic.txtの下のほうのコード。上位x個のみ色を変えることも簡単にできます。これは、①Top10のみマゼンタ色にするやり方

```
##################################↓
#ファイルに保存(M-A plot)1 ①
##################################↓
param_TOP <- 10 ②↓
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
>
```



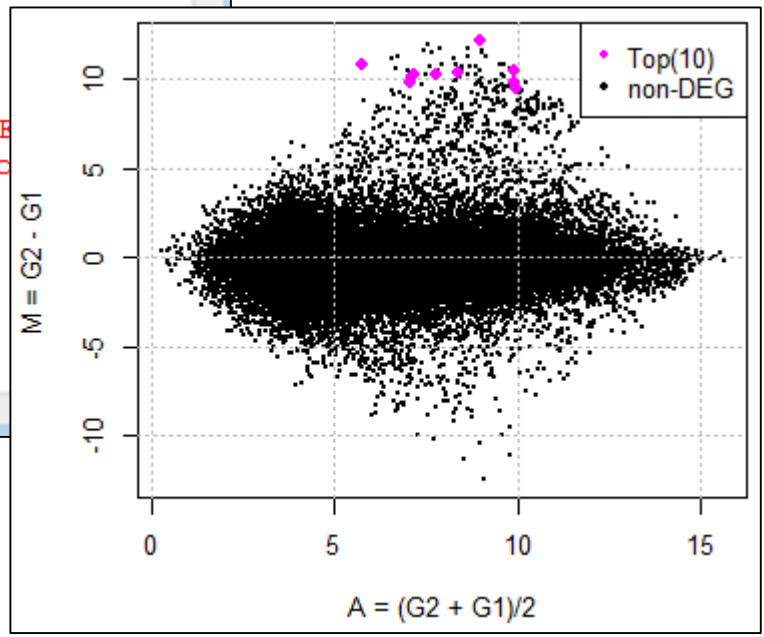
参考

①rcode_limma_MAplot_basic.txtの下のほうのコード。cexのところの値を大きくして、②通常の1.5倍の大きさにしたい場合

```
##############################↓
#ファイルに保存(M-A plot)2 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②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

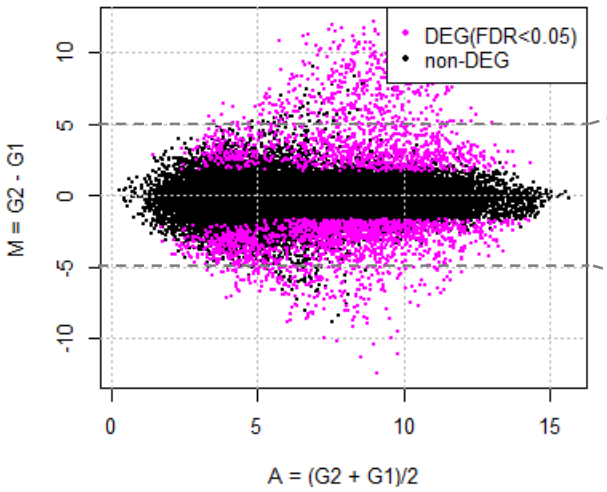
```
> param_TOP <- 10
> png(out_f2, pointsize=13, width=param_fig[1], height=param_fig[2])#出$
> plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1", cex=.1,
> grid(col="gray", lty="dotted")          #指定したパラメータでグリッドを$
> obj <- as.logical(ranking <= param_TOP)#条件を満たすかどうかを判定した$
> points(A[obj], M[obj], 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
>
```

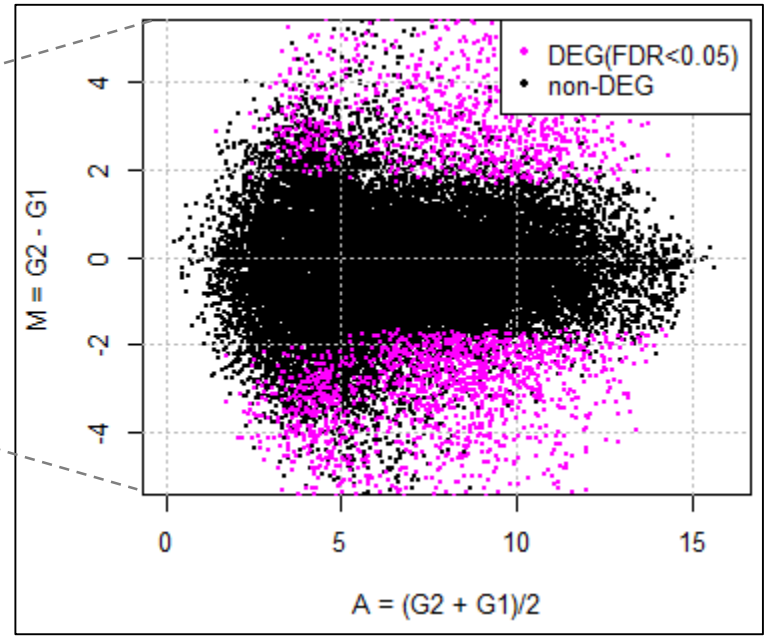


参考

①rcode_limma_MAplot_basic.txtの下のほうのコード。表示範囲を自在に変更可能

```
#############################↓
#ファイルに保存(M-A plot)3 ①
#############################↓
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↓
↓
↓
```

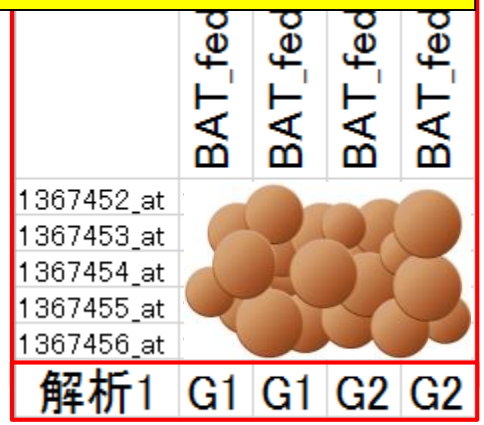


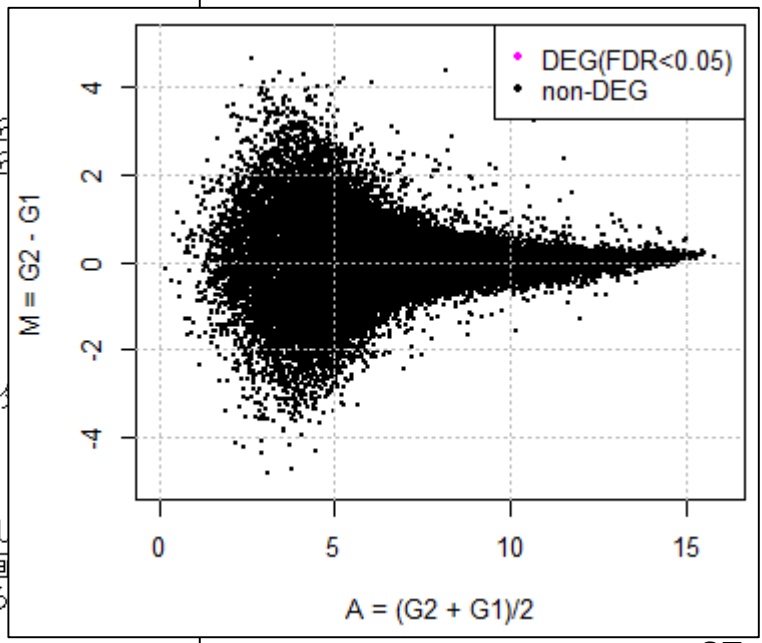


```
in_f <- "data_mas_EN.txt"                  #入力ファイル名を指定してin_fに格納↓
out_f1 <- "hoge1.txt"                      #出力ファイル名を指定してout_f1に格納↓
out_f2 <- "hoge1.png"                      #出力ファイル名を指定してout_f2に格納↓
param_G1 <- 2                              #G1群のサンプル数を指定↓
param_G2 <- 2                              #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↓
```

①

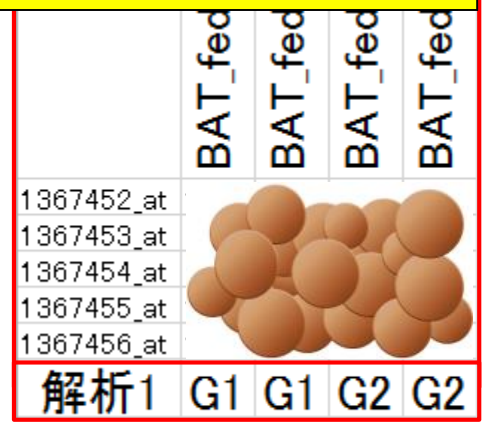
rcode_limma_MAplot_basic2.txt。①デフォルトはc(1,2,3,4)でBAT_fed内のサンプル。例えば、これをc(5,6,7,8)に変更すればBAT_fas内（同一群内）のばらつきの程度を調べることに相当





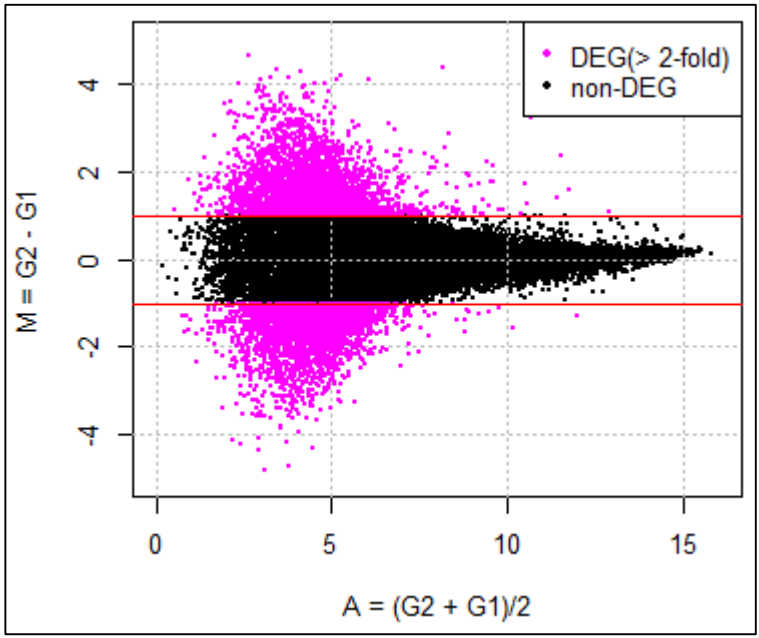
rcode_limma_MAplot_basic2.txtの下の方。2倍以上発現変動しているものをmagenta色で表示。これはBAT_fed群内のバラツキを眺めていることに相当するので(DEGのないデータ)、Fold-changeによるDEG検出の危険性がよくわかります

```
##################################################################
###   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]
plot(A, M, xlab="A = (G2 + G1)/2", ylab="M = G2 - G1",     #M-A pl
     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o
points(A[obj], M[obj], col="magenta", cex=0.1, pch=20)#objがTRUEとなる要素のみ指定
legend("topright", c(paste("DEG(> ", param_FC, "-fold)", sep=""), "non-DEG"),#凡例
       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
```



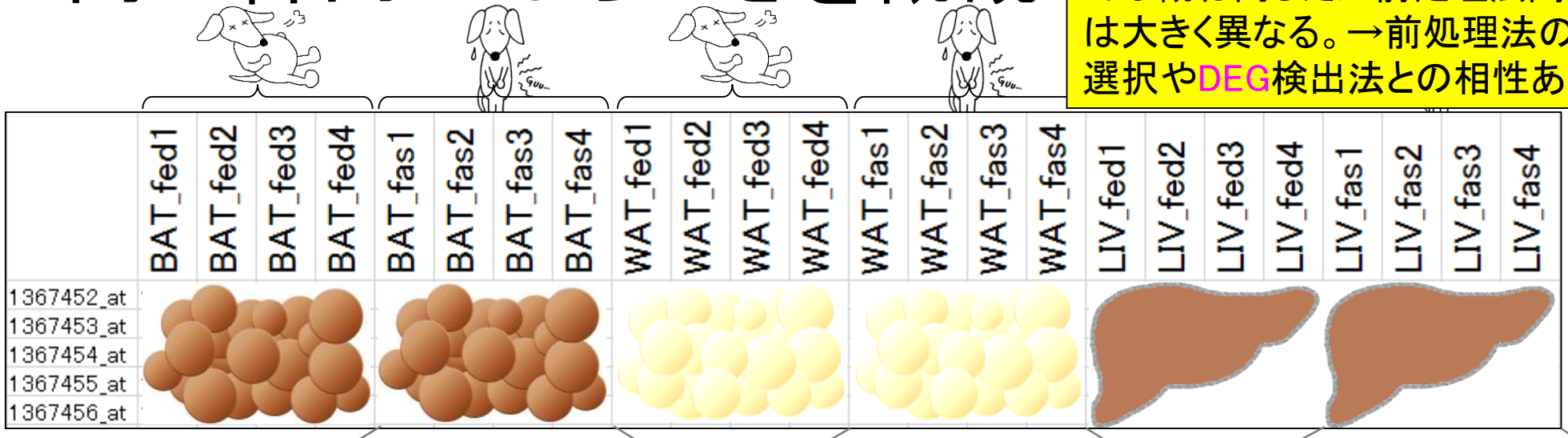
R Console

```
> 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
>
```

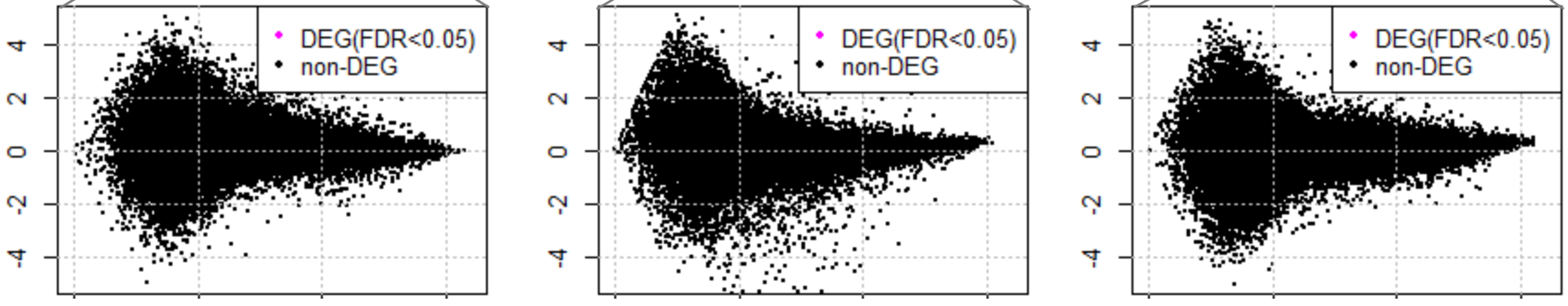


同一群内のばらつきを概観

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