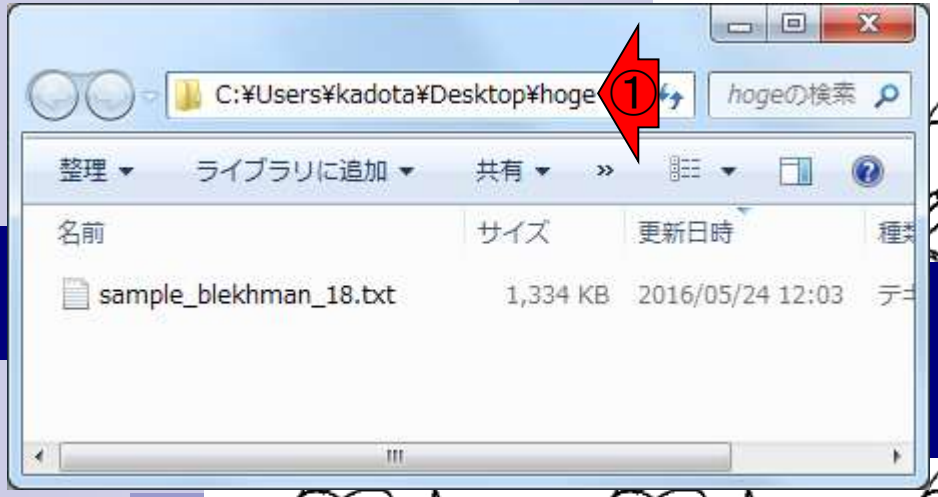
2016.07.13版

①デスクトップ上にhogeフォルダがあり、そこに解析用入力ファイル(sample_blekhman_18.txt)が存在するという前提で行います



第1部：統計解析
～トランスクリプトーム解析2～

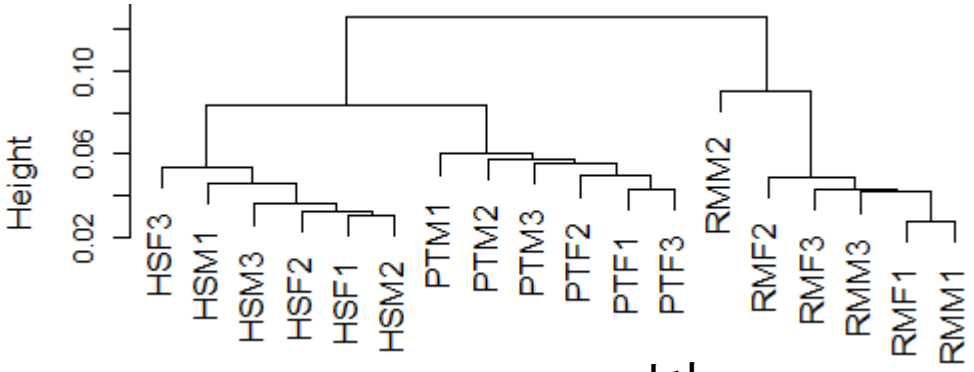
東京大学・大学院農学生命科学研究科
アグリバイオインフォマティクス教育研究プログラム
門田幸二（かどた　こうじ）
kadota@iu.a.u-tokyo.ac.jp
http://www.iu.a.u-tokyo.ac.jp/~kadota/

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

3群間比較用データ

このデータは、3種類の生物種間比較：ヒト(*Homo sapiens*; HS)、チンパンジー(*Pan troglodytes*; PT)、アカゲザル(*Rhesus macaque*; RM)。**どこかの群間で発現変動**している遺伝子を検出するやり方を示す



ヒト (*Homo sapiens*; HS)　チンパンジー (*Pan troglodytes*; PT)　アカゲザル (*Rhesus macaque*; RM)

メス(Female)　オス(Male)　メス　オス　メス　オス

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

Blekhman et al., *Genome Res.*, **20**: 180–189, 2010

性能評価論文

3群間比較用に特化した手法選択のガイドライン。①反復ありデータの場合は(内部的にedgeRの関数を用いた)TCC、②反復なしの場合は(内部的にDESeq2を用いた)TCCがおススメ

BMC Bioinformatics. 2015 Nov 4;16:361. doi: 10.1186/s12859-015-0794-7.

Evaluation of methods for differential expression analysis on multi-group RNA-seq count data.

Tang M[1], Sun J[2], Shimizu K[3], Kadota K[4].

Author information

Abstract

BACKGROUND: RNA-seq is a powerful tool for measuring transcriptomes, especially for identifying differentially expressed genes or transcripts (DEGs) between sample groups. A number of methods have been developed for this task, and several evaluation studies have also been reported. However, those evaluations so far have been restricted to two-group comparisons. Accumulations of comparative studies for multi-group data are also desired.

METHODS: We compare 12 pipelines available in nine R packages for detecting differential expressions (DE) from multi-group RNA-seq count data, focusing on three-group data with or without replicates. We evaluate those pipelines on the basis of both simulation data and real count data.

RESULTS: As a result, the pipelines in the TCC package performed comparably to or better than other pipelines under various simulation scenarios. TCC implements a multi-step normalization strategy (called DEGES) that internally uses functions provided by other representative packages (edgeR, DESeq2, and so on). We found considerably different numbers of identified DEGs (18.5 ~ 45.7% of all genes) among the pipelines for the same real dataset but similar distributions of the classified expression patterns. We also found that DE results can roughly be estimated by the hierarchical dendrogram of sample clustering for the raw count data.

CONCLUSION: We confirmed the DEGES-based pipelines implemented in TCC performed well in a three-group comparison as well as a two-group comparison. We recommend using the DEGES-based pipeline that internally uses edgeR (here called the EEE-E pipeline) for count data with replicates (especially for small sample sizes). For data without replicates, the DEGES-based pipeline with DESeq2 (called SSS-S) can be recommended.

①

②

Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

参考

発現変動解析（サンプル間比較）時に重要となる、sequence depth周辺の正規化法の進展。2013.03.07のHPCIセミナーのスライド

データ正規化周辺

- RPM (Mortazavi *et al.*, *Nat. Methods*, **5**: 621–628, 2008)
  - RPKM(Reads per kilobase of exon per million mapped reads)の長さ補正を行わないバージョン
  - **R**eads **P**er **M**illion mapped readsの略。
- TMM正規化 (Robinson and Oshlack, *Genome Biol.*, **11**: R25, 2010)
  - **T**rimmed **M**ean of **M** valuesの略。edgeRパッケージに実装されている。
  - 発現変動遺伝子（DEG）のデータ正規化時の悪影響を排除すべく、M-A plot上で周縁部にあるデータを使わずに（トリムして）正規化係数を決定する方法。
- TbT正規化 (Kadota *et al.*, *Algorithms Mol. Biol.*, **7**: 5, 2012)
  - **TMM法の改良版**で、TMM-baySeq-TMMという3ステップで正規化を行う方法。
  - 1st stepで得られたTMM正規化係数を用いて、2nd step (baySeq)でDEG同定を行い、3rd step (TMM)ではDEGを排除した残りのデータでTMM正規化。DEGの影響を排除しつつもできるだけ多くのnon-DEGデータを用いて頑健に正規化係数を決めるという思想（DEG elimination strategy提唱論文）。
- iDEGES正規化（Sun *et al.*, *BMC Bioinformatics*, **14**: 219, 2013）
  - **TCC**パッケージの原著論文
  - DEG elimination strategy (DEGES) を一般化し、より高速且つ頑健にしたもの。TbTは「複製あり」のデータのみにしか対応していなかったが、「複製なし」データにも対応。
  - iDEGES/edgeR正規化法：「複製あり」データ正規化用。TMM-(edgeR-TMM)$_n$パイプライン
  - iDEGES/DESeq正規化法：「複製なし」データ正規化用。DESeq-(DESeq-DESeq)$_n$パイプライン

TbT正規化法の説明。2014.07.22
のイルミナウェビナー時のスライド

TbT正規化法

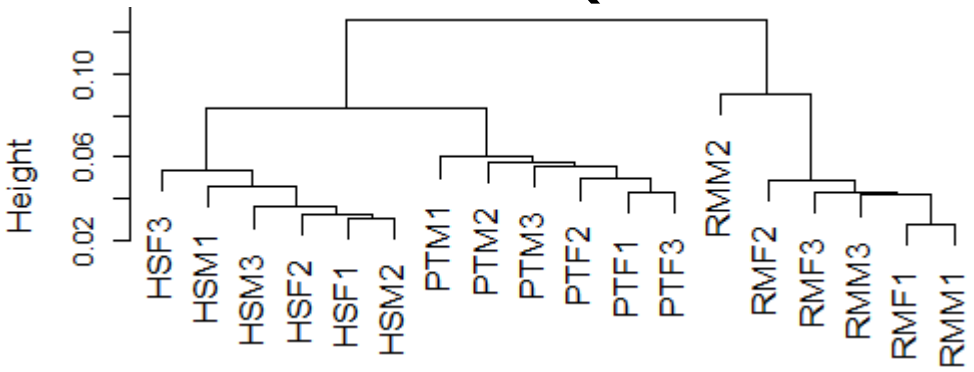
- TCCパッケージに実装している基本コンセプトの原著論文
- 本来の目的である発現変動遺伝子(DEG)自体がデータ正規化時に悪影響を与えるのでDEG候補を除去して正規化を行うほうがよいこと(DEG Elimination Strategy)を提唱した論文。既存の正規化法は、比較するグループ間でDEG数に偏りがない(unbiased DE)場合にはうまく正規化できるが、偏りがある場合(biased DE)にはうまく正規化できないことを示した。
- TbT法の実体は、①edgeRパッケージ中のTMM正規化法実行、②baySeqパッケージ中のDEG検出法実行、および③DEG候補を除去した残りのnon-DEG候補のみを用いたTMM正規化法実行、の3ステップを基本とするTMM-baySeq-TMMパイプライン。出力は正規化後の結果(正確には正規化係数)なので、TbT正規化後に任意のDEG検出法を適用することで一連の発現変動解析が終了することになる。例えばTbT正規化法実行後にedgeR中のDEG検出法を適用する一連の手順はTMM-baySeq-TMM-edgeRに相当し、原著論文中ではedgeR/TbTと略記している。論文中ではTbTにした理由を論理的に書いたが、本音は”ToT”に近いものということでTMMとbaySeqを採用。
- 提案したマルチステップの正規化パイプラインは、第2および第3ステップを繰り返して実行することでより頑健な正規化を実現可能であることも示している。これが図3で説明しているiterative TbT approachに相当するものであり、TMM-(baySeq-TMM)nとも表現できる。例えばiterative TbT正規化法実行後にedgeR中のDEG検出法を適用する一連の手順はTMM-(baySeq-TMM)n-edgeRに相当する。n = 0の場合はTMM-edgeRとなり、これはedgeRパッケージ中のオリジナルの手順と同じである。

TbT正規化法の説明。2014.07.22のイルミナウェビナー時のスライド

TCC

- TbT論文の考え方を一般化し、Rパッケージとしてまとめたという論文
- TbTはDEG Elimination Strategy (DEGES；でげす)に基づく一つの正規化パイプラインにすぎないこと、第2ステップのbaySeqによるDEG同定ステップが律速であり高速化が課題であったこと、そして各ステップにおいて他の方法が原理的に適用可能であることなどを述べている。
- 第2ステップのDEG同定法をedgeR中のものに置き換えると、TMM-edgeR-TMMという正規化パイプラインになる。これは、全てedgeRパッケージ中の関数のみで成立するため、DEGES/edgeRと略記している。また、DEGES正規化後にedgeR中のDEG同定法を適用する一連の解析手順は「DEGES/edgeR-edgeR」または「TMM-edgeR-TMM-edgeR」と表記できる。これは実質的にedgeRパッケージ中のオリジナルの解析手順を2回繰り返して行っていることと同義である(ただし、第3ステップのTMMは検出されたDEG候補以外のデータで実行される)。それが、実質的に「TCCは例えばiterative edgeRという理解でよい」と主張する根拠である。
- TbT論文中で言及したiterative TbTに相当するものは、この論文中ではiterative DEGES (略してiDEGES)と称している。例えば、「iDEGES/edgeR-edgeR」はTMM-(edgeR-TMM)n-edgeRに相当する。n=1は「DEGES/edgeR-edgeR」に相当する。nが2以上の場合がiDEGESに相当するが、nの数を増やしてもその分計算コストがかかる一方で、実質的にn=3程度で頭打ちになることを論文中で示している。それゆえ、iterative DEGESのデフォルトはn=3としている。compcodeR (Soneson, C., Bioinformatics, 2014)中でもデフォルトはそうなっている。

3群間比較(反復あり)



sample_blekhman_18.txtを入力として、ヒト(HS)6サンプル、チンパンジー(PT)6サンプル、アカゲザル(RM)6サンプルの3群間比較を行います。**どこかの群間で発現変動**している遺伝子を検出するやり方です。各群のサンプルは全て別個体です。例えばヒトの場合は6人分のデータ(6 biological replicates)であり、1人のサンプルを6個に分割したデータ(6 technical replicates)ではありません

ヒト (*Homo sapiens*; HS) | チンパンジー (*Pan troglodytes*; PT) | アカゲザル (*Rhesus macaque*; RM)

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

3群間比較(反復あり)

②例題7。③入力はsample_blekhman_18.txt、出力はhoge7.txtのみ。M-A plotはない

- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | DESeq(Anders_2010) (last modified 2014/03/14)
- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | edgeR(Robinson_2010) (last modified 2014/01/07)
- 解析 | 発現変動 | 3群間 | 対応なし | について (last modified 2015/11/05)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq2(Love_201 (last modified 2015/02/04)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun_2013) (①) modified 2015/11/05)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | EBSeq(Leng_2013 (last modified 2016/02/16) NEW
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | SAMseq(Li_2013) (last modified 2015/02/10)

解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun_2013)

TCCを用いたやり方を示します。内部的にiDEGES/edgeR(Sun_2013)正規化を実行したのち、edgeRパッケージ中のGLM LRT法で発現変動遺伝子(Differentially expressed Genes; DEGs)検出を行っています。TCC原著論文中のiDEGES/edgeR-edgeRという解析パイプラインに相当します。TCC原著論文(Sun et al., BMC Bioinformatics, 2013)では 3群間複製ありデータ用の推奨パイプラインを示していませんでしたが、多群間比較用の推奨ガイドライン提唱論文 (Tang et al., BMC Bioinformatics, 2015) で推奨しているパイプライン"EEE-E"が"iDEGES/edgeR-edgeR"と同じものです。この2つの論文を引用し、安心してご利用ください。尚、ここでやっていることはANOVAのような「どこかの群間で発現に差がある遺伝子を検出」です。

「ファイル」-「ディ

1. サンプルデータ15

シミュレーションデー
DEG (gene_1～gen
gene_3000がG3群で

```
in_f <- "data_
out_f <- "hoge
param_G1 <- 3
param_G2 <- 3
param_G3 <- 3
param_FDR <- 0

#必要なパッケー
```

(②) 7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f <- "hoge7.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を指定

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

#入力ファイルの読み込み
, quote="")#in_fで指定した
```

③

Sun et al., *BMC Bioinformatics*, **14**: 219, 2013
Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

3群間比較(反復あり)

①ここで各群のサンプル数(列数)を指定。
②FDR閾値を指定するところだが、出力ファイルの読み取り方が分かっていれば気にしなくてもよい。コピペ。約2分

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f <- "hoge7.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を指定

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定した

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnorma
```

①

②

R Console

```
> getwd()
[1] "C:/Users/kadota/Desktop/hoge"
> list.files(pattern="blekh")
[1] "sample_blekhman_18.txt"
>
```

Sun et al., *BMC Bioinformatics*, **14**: 219, 2013
Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

3群間比較(反復あり)

左上はコード下部。①赤枠部分は、4種類のFDR閾値を満たす遺伝子数を表示している。例えば②は5%の偽物混入を許容すると7,236個の遺伝子が得られるということ。7,236個のうちの5%分が偽物で、残りの95%分が本物(のDEG)と判断する

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2(
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger",
result <- getResult(tcc, sort=FALSE)    #p値

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized
tmp <- tmp[order(tmp$rank),]              #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value               #q-va
sum(q.value < 0.05)                       #FDR
sum(q.value < 0.10)                       #FDR
sum(q.value < 0.20)                       #FDR
sum(q.value < 0.30)                       #FDR
```

①

R Console

```
>
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 7236
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 8485
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 10068
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 11467
>
```

②

①

2016年3月の結果

2016年3月3-4日のHPCI講習会時の結果。パッケージのバージョンが異なると結果も異なるという一例です。この当時のバージョンは忘れました

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、
tcc <- new("TCC", data, data.cl)       #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1,
normalized <- getNormalizedData(tcc)   #正規

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger",
result <- getResult(tcc, sort=FALSE)   #p値

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]           #発現
write.table(tmp, out_f, sep="\t", append=F,
sum(tcc$stat$q.value < 0.05)           #FDR
sum(tcc$stat$q.value < 0.10)           #FDR
sum(tcc$stat$q.value < 0.20)           #FDR
sum(tcc$stat$q.value < 0.30)           #FDR
```

① ②

R Console

```
> tcc <- estimateDE(tcc, test.method="edger", FDR=param$
TCC::INFO: Identifying DE genes using edger ...
TCC::INFO: Done.
> result <- getResult(tcc, sort=FALSE)    #p値などの結果$
>
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]             #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
> sum(tcc$stat$q.value < 0.05)             #FDR < 0.05を$
[1] 7247
> sum(tcc$stat$q.value < 0.10)             #FDR < 0.10を$
[1] 8487
> sum(tcc$stat$q.value < 0.20)             #FDR < 0.20を$
[1] 10063
> sum(tcc$stat$q.value < 0.30)             #FDR < 0.30を$
[1] 11469
>
```

① ②

DEG数の見積もり

FDR閾値が比較的緩めのところを眺め、①20,689 genes中8,000個程度がどこかの群間で発現変動している本物のDEGと判断する。②解析に用いたパッケージのバージョン情報を調べるやり方。sessionInfo()でもよい

• DEG数の見積もり

```
7236*(1 - 0.05)
8485*(1 - 0.10)
10068*(1 - 0.20)
11467*(1 - 0.30)
packageVersion("TCC")                    #パッケージのバージョン情報を表示
```

R Console

```
> sum(q.value < 0.05)                    #FDR = 0.05 (q$
[1] 7236
> sum(q.value < 0.10)                    #FDR = 0.10 (q$
[1] 8485
> sum(q.value < 0.20)                    #FDR = 0.20 (q$
[1] 10068
> sum(q.value < 0.30)                    #FDR = 0.30 (q$
[1] 11467
> 7236*(1 - 0.05)
[1] 6874.2
> 8485*(1 - 0.10)
[1] 7636.5
> 10068*(1 - 0.20)
[1] 8054.4
> 11467*(1 - 0.30)
[1] 8026.9
> packageVersion("TCC")                  #パッケージの$
[1] '1.12.0'
>
```

①

②

出力ファイル解説

①出力ファイル(hoge7.txt)の中身を解説。②正規化後のデータ、③統計解析結果(p値、q値、順位情報など)。④発現変動順にソートされた結果

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f <- "hoge7.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を指定
```

①

④

rownames(to	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	4.7E-113	1.6E-109	6	1
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.5E-106	7.4E-103	7	1
ENSG00000220191	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.2E-105	5.7E-102	8	1
ENSG00000208978	98	173	118	192	66	142	10	19	6	16	3	1	###	###	###	###	###	###	ENS	NA	NA	1.9E-100	4.31E-97	9	1
ENSG00000208070	0	0	0	0	0	0	0	0	0	0	0	0	190	186	98	225	205	113	ENS	NA	NA	1.75E-98	3.63E-95	10	1

②

③

コードの解説

正規化後のデータを取り出す部分の説明。①データ正規化（正確には正規化係数を得た）後のtccオブジェクトを入力として、②getNormalizedData関数を用いて正規化後のデータを取得した結果をnormalizedオブジェクトに格納し、③出力の一部として組み込んでいる

7. サンブルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genor
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(R

①

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
```

②

③

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1

③

コードの解説

（このウェブページではお約束的にそうしているので）
①1番左側を行名としているが、実は②のresultオブジェクト内の③1番左側が同じ情報なのでなくてもよい

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
```

①

②

③

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E−148	4.6E−144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E−141	1.9E−137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E−131	3.4E−127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E−125	3.6E−121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E−115	2.7E−111	5	1

①

②

コードの解説

①rank（順位）情報は、②p値をベースに計算している。p値が低いほど発現変動の度合いが高いことを意味する。実際、③1位の発現パターンはチンパンジー(PT)群で4桁以上のカウント数で、それ以外の群では1桁となっており妥当

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 201
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
```

②　①

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	ene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	③	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	EN	NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1

ANOVA的な解析…

①2位はアカゲザル(RM)群で高発現のパターン、②3位はヒト(HS)群で高発現のパターン。(どの群間で違いがあるかは問わずに)どこかの群間で発現に差があるものを検出する枠組み(ANOVA的な解析;門田スラング)なので、このような結果になる

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
```

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	E	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551		NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	E	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1

コードの解説

①の段階では、まだ発現変動順にはソートされておらず、cbind関数を用いて列方向で結合した(column bind)結果をtmpオブジェクトに格納しているだけ。実際、②resultオブジェクトは入力ファイルの遺伝子名の並びになっている

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_F
result <- getResult(tcc, sort=FALSE)    #p値などの結果を

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)
tmp <- tmp[order(tmp$rank),]            #発現変動順にソー
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.n

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value             #q-valueをq.valueに
sum(q.value < 0.05)                     #FDR = 0.05 (q-valu
sum(q.value < 0.10)                     #FDR = 0.10 (q-valu
sum(q.value < 0.20)                     #FDR = 0.20 (q-valu
sum(q.value < 0.30)                     #FDR = 0.30 (q-valu
```

②

①

R Console

```
> head(result)
          gene_id a.value m.value      p.value
1 ENSG00000000003      NA      NA 0.006423850
2 ENSG00000000005      NA      NA 0.101788674
3 ENSG00000000419      NA      NA 0.907001891
4 ENSG00000000457      NA      NA 0.001335635
5 ENSG00000000460      NA      NA 0.002865749
6 ENSG00000000938      NA      NA 0.075091281
      q.value  rank estimatedDEG
1 0.021627831  6145            1
2 0.207498855 10149            0
3 1.000000000 17496            0
4 0.005611892  4924            1
5 0.010860870  5459            1
6 0.161863255  9598            0
>
```

コードの解説

（赤字のコメント部分をよく見ればわかるがw）①
発現変動順にソートしている部分がココ。
tmp[order(x),]は、xの並びで行をソートするお約束的な書き方。発現変動順にソートしたくない場合は、行頭に#をつけてコメントアウトすればよい

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$cou  , normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]                #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f, sep="  , append=F, quote=F, row.names=F)#tmpの中身を指定した

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value                 #q-valueをq.valueに格納
sum(q.value < 0.05)                         #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数を
sum(q.value < 0.10)                         #FDR = 0.10 (q-value < 0.10)を満たす遺伝子数を
sum(q.value < 0.20)                         #FDR = 0.20 (q-value < 0.20)を満たす遺伝子数を
sum(q.value < 0.30)                         #FDR = 0.30 (q-value < 0.30)を満たす遺伝子数を
```

コードの解説

①の部分は、出力ファイル中の②q.value列を見ているのと同じ

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です。

```
#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
sum(q.value < 0.05)                      #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数を表
sum(q.value < 0.10)                      #FDR = 0.10 (q-value < 0.10)を満たす遺伝子数を表
sum(q.value < 0.20)                      #FDR = 0.20 (q-value < 0.20)を満たす遺伝子数を表
sum(q.value < 0.30)                      #FDR = 0.30 (q-value < 0.30)を満たす遺伝子数を表
```

①

②

																		RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E−148	4.6E−144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E−141	1.9E−137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E−131	3.4E−127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E−125	3.6E−121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E−115	2.7E−111	5	1

コードの解説

例えば①5% FDR thresholdというのは、②q-value(q値)が0.05未満(or以下)という条件判定を行っていることと同義。これは有意水準5%というのが、実際の手順としてp-value(p値) < 0.05という条件判定を行っていることを思い出せば納得できる

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生

```
#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtccに
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結果
tmp <- tmp[order(tmp$rank),]            #発現変動順にソートした結果を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
sum(tcc$stat$q.value < 0.05)            #FDR < 0.05を満たす遺伝子数を表示
sum(tcc$stat$q.value < 0.10)            #FDR < 0.10を満たす遺伝子数を表示
sum(tcc$stat$q.value < 0.20)            #FDR < 0.20を満たす遺伝子数を表示
sum(tcc$stat$q.value < 0.30)            #FDR < 0.30を満たす遺伝子数を表示
```

①

②

																		RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000139826	71	98	37	60	65	60	76	78	104	78	73	58	113	131	85	63	123	138	ENS	NA	NA	0.0174	0.049834	7234	1
ENSG00000006576	43	33	37	52	33	33	15	41	24	20	19	9	64	54	48	33	18	29	ENS	NA	NA	0.0175	0.04991	7235	1
ENSG00000116117	31	19	43	49	31	37	44	39	28	50	59	46	24	34	27	23	20	20	ENS	NA	NA	0.0175	0.049973	7236	1
ENSG00000155465	80	51	58	83	51	60	146	161	106	99	48	110	110	48	64	110	47	41	ENS	NA	NA	0.0175	0.050012	7237	0
ENSG00000115694	269	229	225	199	306	334	487	271	422	420	348	595	335	249	410	343	420	471	ENS	NA	NA	0.0175	0.050013	7238	0
ENSG00000151792	115	142	66	109	86	118	56	82	101	31	43	38	128	59	79	62	79	76	ENS	NA	NA	0.0175	0.050013	7239	0

コードの解説

①1番右側のestimatedDEGという名前の列がq-value（q値）< 0.05のところで1から0に切り替わっている。これは、②param_FDRのところで0.05を指定していたから。②を気にしなくてよいと最初に解説したのは、③q.value列の取り扱い方法を理解していればそれで充分だから

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_f
out_f <- "hoge7.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 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を指定

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定した

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納
```

ENSG00000139826	71	98	37	60	65	60	76	78	104	78	73	58	113	131	85	63	123	138	ENS	NA	NA	0.0174	0.049834	7234	1
ENSG00000006576	43	33	37	52	33	33	15	41	24	20	19	9	64	54	48	33	18	29	ENS	NA	NA	0.0175	0.04991	7235	1
ENSG00000116117	31	19	43	49	31	37	44	39	28	50	59	46	24	34	27	23	20	20	ENS	NA	NA	0.0175	0.049973	7236	1
ENSG00000155465	80	51	58	83	51	60	146	161	106	99	48	110	110	48	64	110	47	41	ENS	NA	NA	0.0175	0.050012	7237	0
ENSG00000115694	269	229	225	199	306	334	487	271	422	420	348	595	335	249	410	343	420	471	ENS	NA	NA	0.0175	0.050013	7238	0
ENSG00000151792	115	142	66	109	86	118	56	82	101	31	43	38	128	59	79	62	79	76	ENS	NA	NA	0.0175	0.050013	7239	0

パターン分類…

（どの群間で違いがあるかは問わずに）どこかの群間で発現に差があるものを検出する枠組み（ANOVA的な解析）なのでこのような結果になる、のはしょうがいないとして…。G1(HS)群で高発現のものがx個、G2(PT)群で高発現のものがy個、みたいなものが欲しい！ので順を追って説明

7. サンブルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 20
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してi
out_f <- "hoge7.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を指定
```

rownames(to	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	4.7E-113	1.6E-109	6	1
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.5E-106	7.4E-103	7	1
ENSG00000220191	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.2E-105	5.7E-102	8	1
ENSG00000208978	98	173	118	192	66	142	10	19	6	16	3	1	###	###	###	###	###	###	ENS	NA	NA	1.9E-100	4.31E-97	9	1
ENSG00000208070	0	0	0	0	0	0	0	0	0	0	0	0	190	186	98	225	205	113	ENS	NA	NA	1.75E-98	3.63E-95	10	1

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

• 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(

post-hoc test…

一般的には、(私はやりませんが)ANOVA後にどこの群間で差があるかを調べるために*post-hoc* testを行うようです。ここでは、RNA-seq発現解析用Rパッケージ中のマニュアルによく記載されている「デザイン行列」や「コントラスト」の概念説明を兼ねて、「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」の3つの*post-hoc* testを行います。
マイクロアレイだが、教科書p173-182にも解説あり

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genc
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(R

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を
out_f <- "hoge7.txt"                      #出力ファイル名を
param_G1 <- 6                             #G1群のサンプル数
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を指定
```

rownames(to	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.2E-148	4.6E-144	1	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	1.8E-141	1.9E-137	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.9E-131	3.4E-127	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	6.9E-125	3.6E-121	4	1
ENSG00000134201	16	7	18	3	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	6.4E-115	2.7E-111	5	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	4.7E-113	1.6E-109	6	1
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.5E-106	7.4E-103	7	1
ENSG00000220191	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.2E-105	5.7E-102	8	1
ENSG00000208978	98	173	118	192	66	142	10	19	6	16	3	1	###	###	###	###	###	###	ENS	NA	NA	1.9E-100	4.31E-97	9	1
ENSG00000208070	0	0	0	0	0	0	0	0	0	0	0	0	190	186	98	225	205	113	ENS	NA	NA	1.75E-98	3.63E-95	10	1

• 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun 2013)

3群間比較(反復あり)

①の②例題7は、(どの群間で違いがあるかは問わずに)どこかの群間で発現に差があるものを検出する枠組み(ANOVA的な解析)でした。このコードはデザイン行列などの概念を一切気にすることなく解析結果を得られるように記述していますが、それを明示させたものが、③「基礎」→「応用」になっている項目です。つまり…

7. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

5.と基本的に同じで、入力ファイルが違うだけです。Blekhman et al., Genome Res., 2010の 20,6
samplesのカウントデータです。ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格
out_f <- "hoge7.txt"                      #出力ファイル名を指定してout_fに格
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を指

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定した

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をtccに
result <- getResult(tcc, sort=FALSE)      #p値などの結果をした結果をresultに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出結
tmp <- tmp[order(tmp$rank),]              #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q_value <- tcc$stat$q_value               #q-valueをq_valueに格納
```

• 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013)

オプション明示(ANOVA)

①ココです。①の例題1をベースに作成したコードで、sample_blekhman_18.txtを入力ファイルとして解析します。**実際にはここをクリックしない!**

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～
(last modified 2016/05/24, since 2011)

What's new?

- このウェブページはインストール | についての推奨手順 (Windows2015.04.04版とMacintosh2015.04.03版)に従って
基本的
的にま
- 平成2

- 解析 | 発現変動 | 2群間 | 対応あり | について (last modified 2015/11/10)
- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | TCC中のDEGES/edgeR-edgeR(Sun_2013) (last modified 2014/03/13)
- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | TCC中のDEGES/DESeq-DESeq(Sun_2013) (last modified 2014/03/13)
- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | DESeq(Anders_2010) (last modified 2014/03/14)
- 解析 | 発現変動 | 2群間 | 対応あり | 複製なし | edgeR(Robinson_2010) (last modified 2014/01/07)
- 解析 | 発現変動 | 3群間 | 対応なし | について (last modified 2015/11/05)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq2(Love_2014) (last modified 2015/02/04)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun_2013) (last modified 2016/05/24)推奨 NEW
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | EBSeq(Leng_2013) (last modified 2016/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | SAMseq(Li_2013) (last modified 2015/02/10)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq(Anders_2010) (last modified 2014/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | baySeq(Hardcastle_2010) (last modified 2016/03/07)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | edgeR(Robinson_2010) (last modified 2015/02/03)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) (last modified 2016/02/17)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+baySeq(Hardcastle_2010) (last modified 2016/03/13)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+EBSeq(Leng_2013) (last modified 2016/03/13)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製なし | TCC(Sun_2013) (last modified 2015/11/05)推奨
- 解析 | 発現変動 | 5群間 | 対応なし | 複製あり | TCC(Sun_2013) (last modified 2015/11/05)推奨
- 解析 | 発現変動 | 時系列 | について (last modified 2015/11/11)
- 解析 | 発現変動 | 時系列 | maSigPro(Nueda_2014) (last modified 2015/08/16)
- 解析 | 発現変動 | 時系列 | Bayesian model-based clustering(Nascimento_2012) (last modified 2012/09/10)
- 解析 | 発現変動 | exon/isoform | について (last modified 2015/11/10)

オプション明示(ANOVA

①実際にコピペするのはココのコード。②赤下線が新たにオプションを明示した部分。まずはコピペ実行し、「以前と同じ結果が得られ、この書き方で間違ってない」ことを確認

- オプション明示(ANOVA)①
  「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題1をベースに作成。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_coef <- c(2, 3)                   #デザイン行列中の除きたい列を指定(reduced mod

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignに格納
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR,#DEG検出を実行した結果をt
                  design=design, coef=param_coef)#DEG検出を実行した結果をtccに格納
result <- getResult(tcc, sort=FALSE)    #p値などの計算をした結果をresultに格納
design                                  #デザイン行列の情報を表示

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出
tmp <- tmp[order(tmp$rank),]            #発現変動順にソートした結果を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
```

オプション明示(ANOVA

①確かに以前と同じ結果が得られている。②の部分がクラスラベル情報data.clを基に作成した「デザイン行列」と呼ばれるものです

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題1をベースに作成。

```
                        iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignに格納
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR,#DEG検出を実行した結果をt
                  design=design, coef=param_coef)#DEG検出を実行した結果をtccに格納
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納
design                                  #デザイン行列の情報を表示

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]            #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value             #q-v
sum(q.value < 0.05)                     #FDR
sum(q.value < 0.10)                     #FDR
sum(q.value < 0.20)                     #FDR
sum(q.value < 0.30)                     #FDR
```

②

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 7236
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 8485
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 10068
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 11467
>
```

①

デザイン行列

①の部分のみ要素に分解して説明。②data.clは、入力データの各列（サンプル）がどの群由来かを示す情報。③as.factorは「因子として」取り扱うという宣言。見た目がちょっと変わる、程度の認識でよい。factor analysis (因子分析)のfactorと同じ意味です。「Levelsって何?」とよく聞かれますが、これは群の種類を表し「1と2と3というものがあります」という意味。④ベクトル中のユニークな要素を表示しているものという認識でよい

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の

```
                        iteration=3, FDR=0.1, floorPDEG=
normalized <- getNormalizedData(tcc)    #正規化後のデータ

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))#デザイン行
tcc <- estimateDE(tcc, test.method="edger", FDR=param_
                  design=design, coef=param_coef)#DEG検出を実行した結果をtccに格納
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納
design                                   #デザイン行列の情報を表示

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normal
tmp <- tmp[order(tmp$rank),]             #
write.table(tmp, out_f, sep="\t", append=

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-v
sum(q.value < 0.05)                      #
sum(q.value < 0.10)                      #
sum(q.value < 0.20)                      #FD
sum(q.value < 0.30)                      #FDR
```

①

②

③

④

R Console

```
> data.cl
 [1] 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 3
> as.factor(data.cl)
 [1] 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 3
Levels: 1 2 3
> unique(data.cl)
[1] 1 2 3
>
```

デザイン行列

①作成したデザイン行列のdesignオブジェクトを表示

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TC

```
                        iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規

#本番(①検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, coef=param
result <- getResult(tcc, sort=FALSE)    #p値
design                                   #デ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]             #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-v
sum(q.value < 0.05)                      #FDR
sum(q.value < 0.10)                      #FDR
sum(q.value < 0.20)                      #FDR
sum(q.value < 0.30)                      #FDR
```

①

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

デザイン行列

デザイン行列作成時の①の部分の記述内容が、
②designオブジェクトの③の部分に利用されている

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TC

```
                          iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, param
result <- getResult(tcc, sort=FALSE)    #p値
design                                   #デ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]             #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-v
sum(q.value < 0.05)                      #FDR
sum(q.value < 0.10)                      #FDR
sum(q.value < 0.20)                      #FDR
sum(q.value < 0.30)                      #FDR
```

① ② ③

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

デザイン行列

①は、発現変動解析結果でどこを比較するかを指し示す情報としても利用されるため、②の指定前に分かり易い名前に変更したり、③design作成後に変更したり…するので混乱しがちな部分

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TC

```
                        iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, param
result <- getResult(tcc, sort=FALSE)    #p値
design                                   #デザ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]             #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-v
sum(q.value < 0.05)                      #FDR
sum(q.value < 0.10)                      #FDR
sum(q.value < 0.20)                      #FDR
sum(q.value < 0.30)                      #FDR
```

②

③

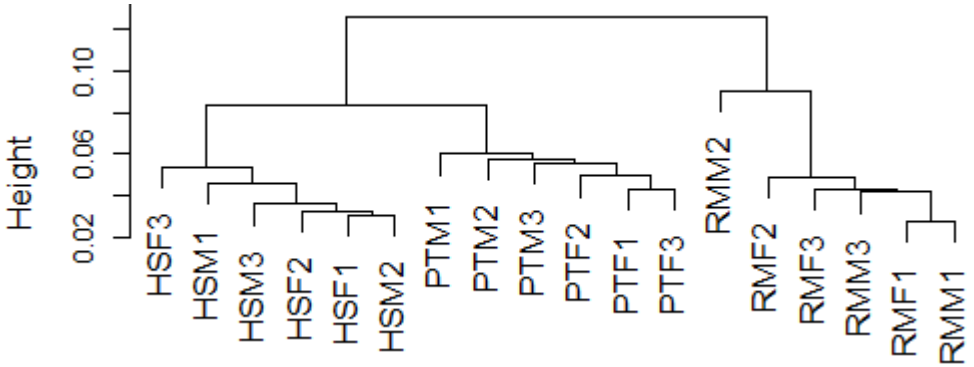
```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

①

①

おさらい

今は、sample_blekhman_18.txtを入力として、ヒト(HS)6サンプル、チンパンジー(PT)6サンプル、アカゲザル(RM)6サンプルの3群間比較を行っている
ANOVA的な解析を行い、**どこかの群間で発現変動**している遺伝子を検出しようとしている。**帰無仮説は「どの群間でも差がない(G1 = G2 = G3)」**である



ヒト (*Homo sapiens*; HS)　チンパンジー (*Pan troglodytes*; PT)　アカゲザル (*Rhesus macaque*; RM)

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

帰無仮説の設定

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を
param_G2 <- 6                             #G2群のサンプル数を
param_G3 <- 6                             #G3群のサンプル数を
param_FDR <- 0.05                         #DEG検出時のfalse
param_coef <- c(2, 3)                     #デザイン行列中の除

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3,
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェ

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))#デザイン行列を作成した結果をdesignに格納
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR,#DEG検出を実行した結果を
                  design=design, coef=param_coef)#DEG検出を実行した結果をtccに格納
result <- getResult(tcc, sort=FALSE)    #順などの結果をした結果をresultに格納
design                                  #ン行列の情報を表示

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検
tmp <- tmp[order(tmp$rank),]            #発現変動順にソートした結果を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
```

①

②

①は、どういう検定を行いたいかを指定するところという理解でよい。②ここで利用。正確には、一般化線形モデル(Generalized Linear Model; GLM)を用いた解析時に指定するreduced modelに相当する。尚、以前作成したデザイン行列(designオブジェクト)がfull modelと呼ばれるものに相当する。reduced modelは、full modelの一部を指定する(だからreduced)。①では、full modelから除きたいものを指定する。そしてそれは実質的に検定したい事柄(帰無仮説)を指定していることに相当する。が、そんなことを真面目に説明してもついてこられないヒトがほとんどだし、そんなことはどうでもいいから正しい手順と結果の解釈を教えて欲しいというのが過去の受講生のコメントであり、私を含むエンドユーザはそれでいいと思っています

帰無仮説の設定

①の2と3からなる数値ベクトルは、②行列designから取り除きたい列を指定することに相当します。この場合は③の2列分の除去に相当

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TC

```
in_f <- "sample_blekhman_18.txt"          #入力
out_f <- "hoge1.txt"                      #出力
param_G1 <- 6                             #G1群
param_G2 <- 6                             #G2群
param_G3 <- 6                             #G3群
param_FDR <- 0.05                         #DEG
param_coef <- c(2, 3)                     #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)          #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)      #正規

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, coef=param
result <- getResult(tcc, sort=FALSE)      #p値
design                                    #デザ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]              #発現
write.table(tmp, out_f, sep="\t", append=F,
```

①

②

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

③

帰無仮説の設定

そうすると、残るは①この列情報のみ。今我々が設定したい帰無仮説は「どの群間でも差がない」、つまり「G1 = G2 = G3」です。それを指定できてるっぽいことがわかります

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC

```
in_f <- "sample_blekhman_18.txt"          #入力
out_f <- "hoge1.txt"                      #出力
param_G1 <- 6                             #G1群
param_G2 <- 6                             #G2群
param_G3 <- 6                             #G3群
param_FDR <- 0.05                         #DEG
param_coef <- c(2, 3)                     #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)          #TCC

#本番(正規化)
tcc <- calcNormFactors(tc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)      #正規

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, coef=param
result <- getResult(tcc, sort=FALSE)      #p値
design                                    #デザ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]              #発現
write.table(tmp, out_f, sep="\t", append=F,
```

R Console

```
> design
   (Intercept) as                    0                    0
1            1                    0                    0
2            1                    0                    0
3            1                    0                    0
4            1                    0                    0
5            1                    0                    0
6            1                    0                    0
7            1                    1                    0
8            1                    1                    0
9            1                    1                    0
10           1                    1                    0
11           1                    1                    0
12           1                    1                    0
13           1                    0                    1
14           1                    0                    1
15           1                    0                    1
16           1                    0                    1
17           1                    0                    1
18           1                    0                    1
attr(,"assig
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

①

帰無仮説の設定

だって①入力ファイルは18列分のデータで、デザイン行列designから②18列全てに同じ1という数値が割り振られるようなところのみ残しているのだから。これがreduced modelです

- オプション明示(ANOVA)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC

```
in_f <- "sample_blekhman_18.txt"          #入力
out_f <- "hoge1.txt"                      #出力
param_G1 <- 6                             #G1群
param_G2 <- 6                             #G2群
param_G3 <- 6                             #G3群
param_FDR <- 0.05                         #DEG
param_coef <- c(2, 3)                     #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)          #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)      #正規

#本番(DEG検出)
design <- model.matrix(~ as.factor(data.cl))
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, coef=param
result <- getResult(tcc, sort=FALSE)      #p値
design                                    #デザ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]              #発現
write.table(tmp, out_f, sep="\t", append=F,
```

①

R Console

```
> design
   (Intercept) a                          0                 0
1            1                          0                 0
2            1                          0                 0
3            1                          0                 0
4            1                          0                 0
5            1                          0                 0
6            1                          0                 0
7            1                          1                 0
8            1                          1                 0
9            1                          1                 0
10           1                          1                 0
11           1                          1                 0
12           1                          1                 0
13           1                          0                 1
14           1                          0                 1
15           1                          0                 1
16           1                          0                 1
17           1                          0                 1
18           1                          0                 1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

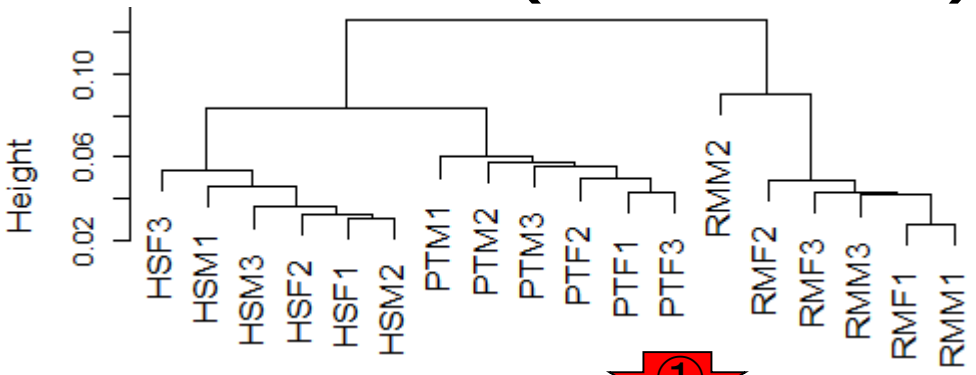
②

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

帰無仮説(G1 = G2)

同じノリで、どうやって「①G1(HS)群 vs. ②G2(PT)群」の検定を行うのかについて述べます。これは帰無仮説G1 = G2をどうやって設定するのかと同義



① ヒト（*Homo sapiens*; HS）

② チンパンジー（*Pan troglodytes*; PT）

アカゲザル（*Rhesus macaque*; RM）

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

帰無仮説(G1 = G2)

①ここです。さきほどのANOVA解析（帰無仮説：G1 = G2 = G3）時との違いは、②で指定するのが2のみいう点。コピペはまだしなくてよい

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群以上 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題2をベースに作成。「G1群 vs. G2群」の比較を行いたいときの指定方法(帰無仮説: G1 = G2)です。「full modelに相当するデザイン行列designの2列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

①

②

```
in_f <- "sample_blekhman_18.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_coef <- 2                         #デザイン行列中の除きたい列を指定(reduced mode

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納
```

帰無仮説(G1 = G2)

これは、①行列designから取り除く列として、②2のみを指定していることに相当。つまり③の部分のみの除去

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | 1(
G2群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの2列目のパラメータを除いたものをreduced modelと

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge2.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 2                         #デサ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

①

②

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

③

帰無仮説(G1 = G2)

何故①この部分の除去が「G1 = G2」という帰無仮説の設定に相当するのかは…

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC
G2群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの2列目のパラメータを除いたものをreduced model

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge2.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 2                         #デサ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

①

帰無仮説(G1 = G2)

①これが「帰無仮説：G1 = G2 = G3」だったことを頼りに考える。つまり、全て1

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC
G2群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの2列目のパラメータを除いたものをreduced model

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge2.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 2                         #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

帰無仮説(G1 = G

（自分が納得しやすいように解釈すればよいが）①をベースにするのが基本。例えば「① − ②」で、G1とG2のみが1になるので、「帰無仮説：G1 = G2」を設定できたと考えればよい

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TC
G2群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの2列目のパラメータを除いたものをreduced model

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge2.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 2                         #デサ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assig
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

>
```

①

②

帰無仮説(G1 =

- 帰無仮説(G1 = G2)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC
G2群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの2列目のパラメータを除いたものをreduced modelと

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge2.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 2                         #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1,
normalized <- getNormalizedData(tcc)    #正規
```

②

R Console

```
> design
   (Intercept) as.
1             1
2             1                    0                    0
3             1                    0                    0
4             1                    0                    0
5             1                    0                    0
6             1                    0                    0
7             1                    1                    0
8             1                    1                    0
9             1                    1                    0
10            1                    1                    0
11            1                    1                    0
12            1                    1                    0
13            1                    0                    1
14            1                    0                    1
15            1                    0                    1
16            1                    0                    1
17            1                    0                    1
18            1                    0                    1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> design[, 1] - design[, 3]
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 1  1  1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0
>
```

①

③

つまり①のような感じ。デザイン行列designの1列目から3列目を引くと、②G1群とG2群に相当する、③最初の12列分が1になる

帰無仮説(G1 = G2)

①このやり方で正しく(*post-hoc* testの一環として)G1 vs. G2の2群間比較ができているかを確認すべくコピペ。②こんな感じになっていればOK(バージョンによって多少数値は異なるかも…)

- 帰無仮説(G1 = G2) ①

「解析 | 発現変動 | 3群以上 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題2を[illegible] G2群」の比較を行いたいときの指定方法(帰無仮説: G1 = G2)です。「full modelに相当するデザイン行列designの2列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_coef <- 2                         #デザイン行列中の除きたい列を指定(reduced mode

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 2282
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 3077
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 4334
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 5445
>
```

②

帰無仮説(G1 = G2)

①出力ファイルを眺め（赤枠内の上位10個）、確かに「②G1群 vs. ③G2群」の結果になっていることを確認し安心する。スライドを見るだけ

- 帰無仮説(G1 = G2)
  「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題2をベースに作成。「G1群 vs. G2群」の比較を行いたいときの指定方法(帰無仮説: G1 = G2)です。「full modelに相当するデザイン行列designの2列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekh_18.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 <- 6                          #G2群のサンプル数を指定
param_G3 <- 6                          #G3群のサンプル数を指定
param_FDR <- 0.05                      #DEG検出時のfalse discovery rate (FDR)閾値を
```

①

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	########	########	1	1
ENSG00000208587	1	1	4	0	0	0	###	###	###	###	###	###	5	2	1	4	9	4	ENS	NA	NA	2.74E-84	2.84E-80	2	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	4.86E-65	3.35E-61	3	1
ENSG00000134391	281	473	201	109	368	449	0	1	0	0	1	0	725	618	451	603	766	373	ENS	NA	NA	1.09E-58	5.65E-55	4	1
ENSG00000163444	4	1	3	5	3	0	127	115	139	143	121	133	84	83	64	69	92	104	ENS	NA	NA	6.32E-54	2.62E-50	5	1
ENSG00000139540	830	###	888	752	###	###	2	2	0	3	0	8	16	78	21	46	38	7	ENS	NA	NA	1.26E-47	4.36E-44	6	1
ENSG00000169218	364	407	###	679	353	352	1	4	2	2	0	0	1	0	9	3	4	1	ENS	NA	NA	9.87E-41	2.92E-37	7	1
ENSG00000174992	37	151	108	93	147	137	0	0	0	0	0	0	0	0	1	0	0	0	ENS	NA	NA	4.19E-39	1.08E-35	8	1
ENSG00000142494	933	530	431	353	883	653	36	16	31	22	23	28	181	69	170	143	77	86	ENS	NA	NA	2.59E-37	5.95E-34	9	1
ENSG00000209005	0	0	0	0	0	0	75	92	118	157	126	41	0	0	0	0	0	0	ENS	NA	NA	3.54E-37	7.31E-34	10	1

②

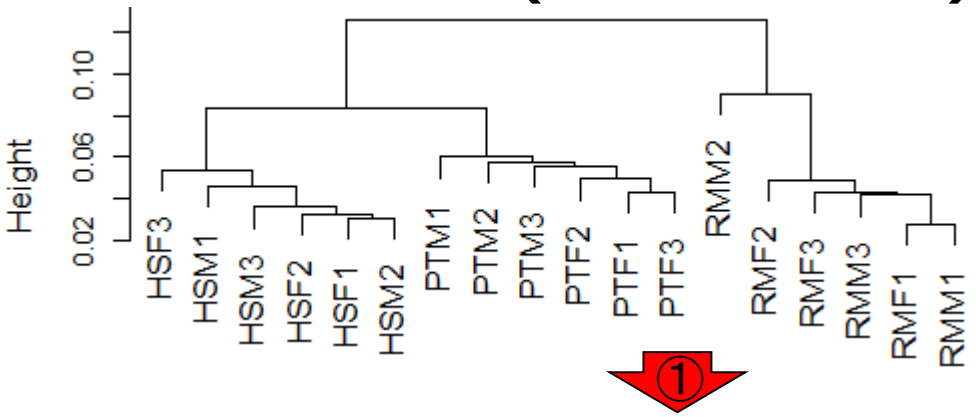
③

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

次は「①G1(HS)群 vs. ②G3(RM)群」

帰無仮説(G1 = G3)



① ヒト（*Homo sapiens*; HS）

チンパンジー（*Pan troglodytes*; PT）

② アカゲザル（*Rhesus macaque*; RM）

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

参考

帰無仮説(G1 = G3)

①ここです。G1 vs. G2(帰無仮説:G1 = G2)との違いは、②で指定するのが2から3に変わった点のみ。コピペはまだしなくてよい

- 帰無仮説(G1 = G3) ①

「解析 | 発現変動 | 3群以上 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題3をベースに作成。「G1群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。「full modelに相当するデザイン行列designの3列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_coef <- 3                         #デザイン行列中の除きたい列を指定(reduced mode

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をt
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納
```

②

帰無仮説(G1 = G3)

- 帰無仮説(G1 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC
G3群」の比較を行いたいときの指定方法(帰無仮説: G1 =
designの3列目のパラメータを除いたものをreduced modelと

```
in_f <- "sample_blekhman_18.txt"        #入力
out_f <- "hoge3.txt"                    #出力
param_G1 <- 6                           #G1群
param_G2 <- 6                           #G2群
param_G3 <- 6                           #G3群
param_FDR <- 0.05                       #DEG
param_coef <- 3                         #デザ

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1,
normalized <- getNormalizedData(tcc)    #正規
```

①

②

参考

同じ論理展開でいくと、①デザイン行列の3列目の除去に相当する。②designの1列目から残った2列目の情報を差し引くと、③G1群とG3群のみが1となり、帰無仮説(G1 = G3)の完成となる

R Console

```
> design
   (Intercept
1           1
2           1
3           1
4           1
5           1
6           1
7           1
8           1
9           1
10          1
11          1
12          1
13          1
14          1
15          1
16          1
17          1
18          1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> design[, 1] - design[, 2]
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 1  1  1  1  1  1  0  0  0  0  0  0  1  1  1  1  1  1
>
```

①

②

③

帰無仮説(G1 = G3)

①G1 vs. G3用コードをコピペ。②こんな感じになっていればOK(バージョンによって多少数値は異なるかも…)

- 帰無仮説(G1 = G3) ①

「解析 | 発現変動 | 3群以上 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題3をベースに作成。「G1群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。「full modelに相当するデザイン行列designの3列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_coef <- 3                         #デザイン行列中の除きたい列を指定(reduced mods

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 5278
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 6324
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 7816
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 9069
>
```

②

参考

帰無仮説(G1 = G3)

①出力ファイルを眺め（赤枠内の上位10個）、確かに「②G1群 vs. ③G3群」の結果になっていることを確認し安心する。スライドを見るだけ

- 帰無仮説(G1 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題3をベースに作成。「G1群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。「full modelに相当するデザイン行列designの3列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekh_18.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 <- 6                          #G1群のサンプル数を指定
param_G2 <- 6                          #G2群のサンプル数を指定
param_G3 <- 6                          #G3群のサンプル数を指定
param_FDR <- 0.05                      #DEG検出時のfalse discovery rate (FDR)閾値を
```

① ←

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.03E-97	4.20E-93	1	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	3.65E-81	3.78E-77	2	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	7.23E-77	4.99E-73	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	5.33E-70	2.76E-66	4	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	6.91E-62	2.86E-58	5	1
ENSG00000112667	159	189	220	125	183	208	291	127	202	270	221	391	0	0	2	0	1	1	ENS	NA	NA	1.30E-61	4.47E-58	6	1
ENSG00000220191	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.56E-59	7.57E-56	7	1
ENSG00000156222	362	510	268	303	424	824	543	266	453	437	428	###	1	2	0	0	1	1	ENS	NA	NA	4.67E-59	1.21E-55	8	1
ENSG00000166780	129	107	145	150	172	175	35	33	88	36	61	60	0	0	0	0	1	0	ENS	NA	NA	8.75E-59	2.01E-55	9	1
ENSG00000214736	147	143	170	171	132	137	52	51	20	61	39	97	0	0	0	0	1	0	ENS	NA	NA	7.45E-56	1.54E-52	10	1

② ③

帰無仮説(G1 = G3)

多く(10個中8個)の発現パターンにおいて、G1群(HS)とG2群(PT)が連動している。もしかしたら「(HS + PT) vs. RM」の比較になっちゃってるのではという疑念を抱くヒトがいるかもしれないが…

- 帰無仮説(G1 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題3を[illegible]G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。「full modelに相当するデザイン行列designの3列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
```

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.03E-97	4.20E-93	1	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	3.65E-81	3.78E-77	2	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	7.23E-77	4.99E-73	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	5.33E-70	2.76E-66	4	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	6.91E-62	2.86E-58	5	1
ENSG00000111[illegible]7	159	189	220	125	183	208	291	127	202	270	221	391	0	0	2	0	1	1	ENS	NA	NA	1.30E-61	4.47E-58	6	1
ENSG00000224[illegible]1	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.56E-59	7.57E-56	7	1
ENSG00000156122	362	510	268	303	424	824	543	266	453	437	428	###	1	2	0	0	1	1	ENS	NA	NA	4.67E-59	1.21E-55	8	1
ENSG00000166780	129	107	145	150	172	175	35	33	88	36	61	60	0	0	0	0	1	0	ENS	NA	NA	8.75E-59	2.01E-55	9	1
ENSG00000214736	147	143	170	171	132	137	52	51	20	61	39	97	0	0	0	0	1	0	ENS	NA	NA	7.45E-56	1.54E-52	10	1

①

②

③

①の部分を眺めることで、その疑念は消せる

帰無仮説(G1 = G3)

- 帰無仮説(G1 = G3)
  「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題3をベースに作成。「G1群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。「full modelに相当するデザイン行列designの3列目のパラメータを除いたものをreduced modelとする」という指定に相当します。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
```

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG0000020	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.03E-97	4.20E-93	1	1
ENSG0000015	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	3.65E-81	3.78E-77	2	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	7.23E-77	4.99E-73	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	5.33E-70	2.76E-66	4	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	6.91E-62	2.86E-58	5	1
ENSG00000112667	159	189	220	125	183	208	291	127	202	270	221	391	0	0	2	0	1	1	ENS	NA	NA	1.30E-61	4.47E-58	6	1
ENSG00000220191	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.56E-59	7.57E-56	7	1
ENSG00000156222	362	510	268	303	424	824	543	266	453	437	428	###	1	2	0	0	1	1	ENS	NA	NA	4.67E-59	1.21E-55	8	1
ENSG00000166780	129	107	145	150	172	175	35	33	88	36	61	60	0	0	0	0	1	0	ENS	NA	NA	8.75E-59	2.01E-55	9	1
ENSG00000214736	147	143	170	171	132	137	52	51	20	61	39	97	0	0	0	0	1	0	ENS	NA	NA	7.45E-56	1.54E-52	10	1

①

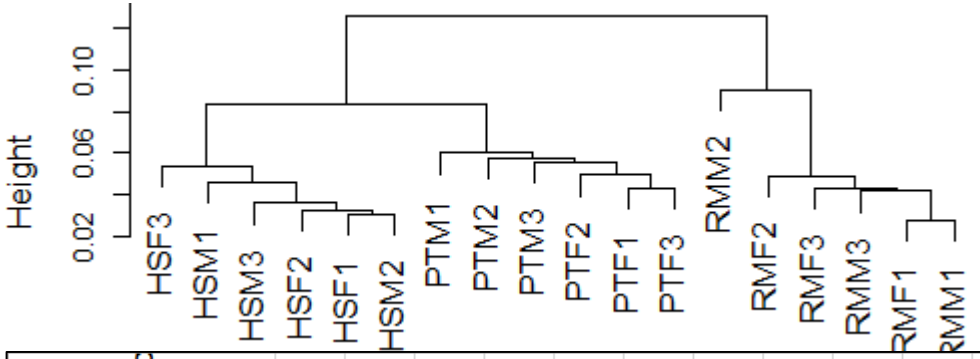
②

③

参考

帰無仮説(G1 = G3)

①の連動の原因は、クラスタリング結果が如実に語っている。つまり、G1群(HS) とG2群(PT)間の発現パターンは、G3群(RM)に比べて似てる

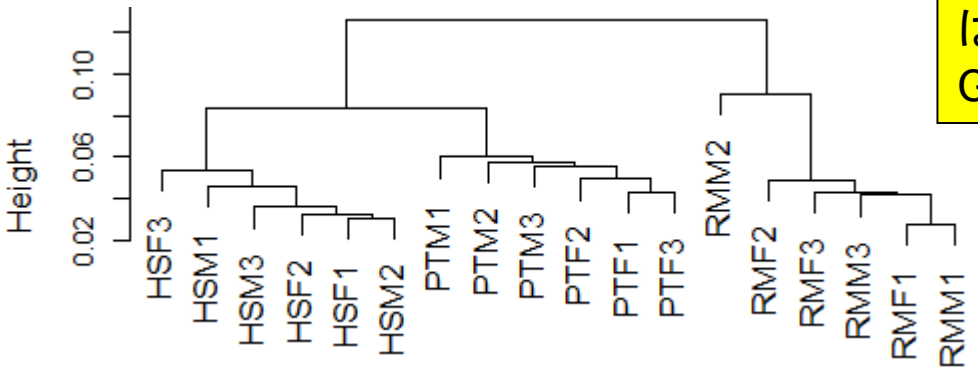


rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000208570	0	0	0	0	0	0	###	###	###	###	###	###	###	###	###	###	###	789	ENS	NA	NA	2.03E-97	4.20E-93	1	1
ENSG00000157399	446	390	298	302	661	462	2	6	3	5	4	2	1	0	0	0	0	2	ENS	NA	NA	3.65E-81	3.78E-77	2	1
ENSG00000209449	0	0	1	0	4	1	1	1	0	2	2	0	656	857	377	718	###	551	ENS	NA	NA	7.23E-77	4.99E-73	3	1
ENSG00000209007	0	0	0	0	0	1	1	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	5.33E-70	2.76E-66	4	1
ENSG00000145244	1	0	1	0	2	3	0	3	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	6.91E-62	2.86E-58	5	1
ENSG00000111	159	189	220	125	183	208	291	127	202	270	221	391	0	0	2	0	1	1	ENS	NA	NA	1.30E-61	4.47E-58	6	1
ENSG00000222	2	5	1	3	6	11	2	5	0	6	10	11	###	918	892	###	935	657	ENS	NA	NA	2.56E-59	7.57E-56	7	1
ENSG00000156122	362	510	268	303	424	824	543	266	453	437	428	###	1	2	0	0	1	1	ENS	NA	NA	4.67E-59	1.21E-55	8	1
ENSG00000166780	129	107	145	150	172	175	35	33	88	36	61	60	0	0	0	0	1	0	ENS	NA	NA	8.75E-59	2.01E-55	9	1
ENSG00000214736	147	143	170	171	132	137	52	51	20	61	39	97	0	0	0	0	1	0	ENS	NA	NA	7.45E-56	1.54E-52	10	1

①

結果のまとめ

FDR 5%と10%の結果のみ表示。それぞれの帰無仮説は、①G1 = G2 = G3、②G1 = G2、③G1 = G3。③のほうが②に比べて遺伝子数（実質的にDEG数）が多いのは、サンプル間クラスタリング結果からも妥当。つまり、G1とG2間の類似度が、G1とG3間の類似度よりも高い



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
```

①

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

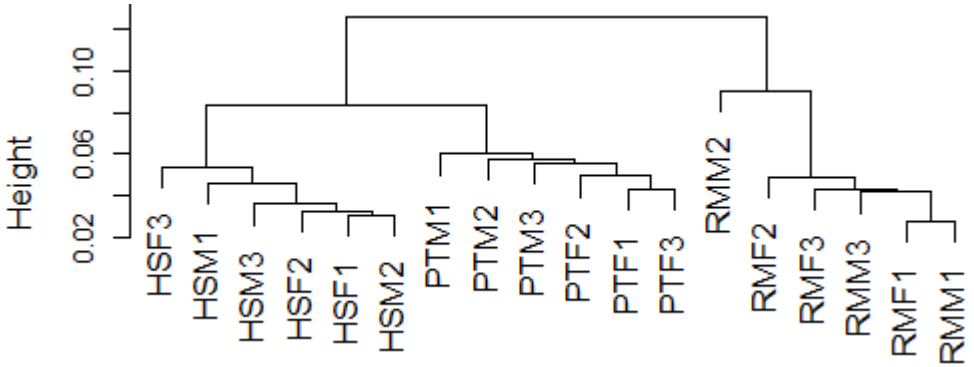
G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

③

妄想

このロジックでいくと、④G2 vs. G3の結果は、③の結果と似てるはず…と妄想する。しかしこれまで使ってきた(G1群を基準とした)実験デザイン行列の枠組みでは、「帰無仮説：G2 = G3」を表現できない



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
```

①

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

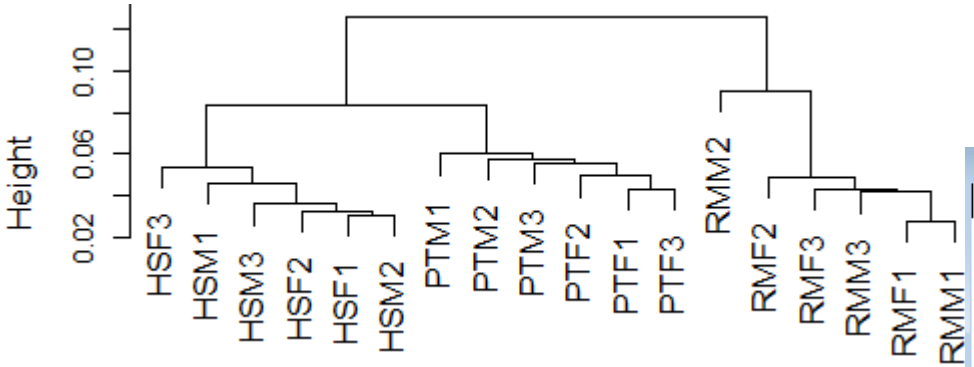
③

G2(PT)群 6 samples　G3(RM)群 6 samples

④

おさらい

(G1群を基準とした)実験デザイン行列designの枠組みで表現できるのは…、①(2列目と3列目を除去して作成する)G1 = G2 = G3



① G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

G1(HS)群 6 samples　G2(PT)群 6 samples

G1(HS)群 6 samples　G3(RM)群 6 samples

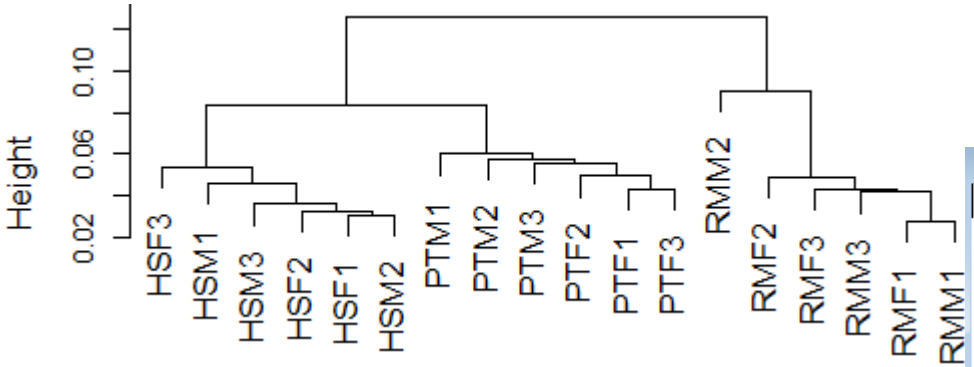
G2(PT)群 6 samples　G3(RM)群 6 samples

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

おさらい

(G1群を基準とした)実験デザイン行列designの枠組みで表現できるのは…、②(2列目を除去して作成する)G1 = G2



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

② G1(HS)群 6 samples　G2(PT)群 6 samples

G1(HS)群 6 samples　G3(RM)群 6 samples

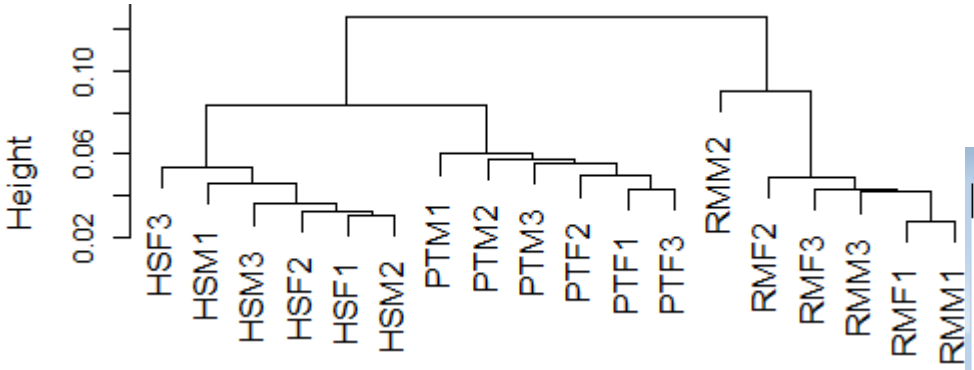
G2(PT)群 6 samples　G3(RM)群 6 samples

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

おさらい

(G1群を基準とした)実験デザイン行列designの枠組みで表現できるのは…、③(3列目を除去して作成する)G1 = G3の計3種類だけ



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

G1(HS)群 6 samples　G2(PT)群 6 samples

③ G1(HS)群 6 samples　G3(RM)群 6 samples

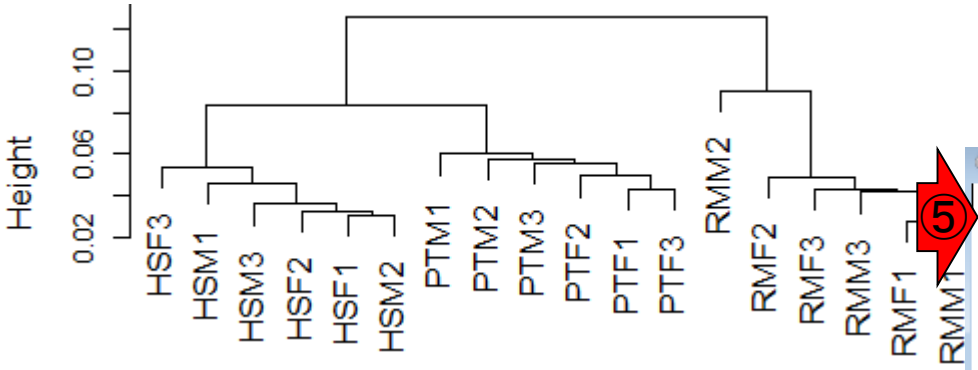
G2(PT)群 6 samples　G3(RM)群 6 samples

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

おさらい

④「帰無仮説：G2 = G3」を表現するためには、⑤デザイン行列designの作成時に、⑥G1群を基準としない書き方にする必要がある



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

G1(HS)群 6 samples　G2(PT)群 6 samples

G1(HS)群 6 samples　G3(RM)群 6 samples

④ G2(PT)群 6 samples　G3(RM)群 6 samples

⑤

R Console

```
> design
   (Intercept) as.factor(data.cl)2 as.factor(data.cl)3
1            1                   0                   0
2            1                   0                   0
3            1                   0                   0
4            1                   0                   0
5            1                   0                   0
6            1                   0                   0
7            1                   1                   0
8            1                   1                   0
9            1                   1                   0
10           1                   1                   0
11           1                   1                   0
12           1                   1                   0
13           1                   0                   1
14           1                   0                   1
15           1                   0                   1
16           1                   0                   1
17           1                   0                   1
18           1                   0                   1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

⑥

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

帰無仮説(G2 = G3)

①ここです。②の赤枠で説明している通りだが、順を追って解説。基本的にはG1群を基準としないデザイン行列を作成したのち、③で設定したい帰無仮説(G2 = G3)を表現可能なコントラスト情報を与えるという流れ

- 帰無仮説(G2 = G3) ①

「解析 | 発現変動 | 3群 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに
G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みで
をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 +
a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2,
a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

②

③

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(0, -1, 1)             #コントラスト情報を指定(reduced model作成用)

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をt
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納
```

帰無仮説(G2 = G3)

①コード下部に移動。②G1群を基準としないデザイン行列作成部分。1行上のコメントアウトしているのが、G1群を基準として作成したこれまでのデザイン行列作成法。違いは③「0 +」がついているか否か

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに作
G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みでは
をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 +
a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2,
a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

```
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
#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(~ 0 + as.factor(data.cl))#デザイン行列を作成した結果をdesignに
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR,#DEG検出を実行した結果をtc
                  design=design, contrast=param_contrast)#DEG検出を実行した結果をtcc
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納
design                                   #デザイン行列の情報を表示

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)#正規化後のデータの右側にDEG検出
tmp <- tmp[order(tmp$rank),]             #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.valueに格納
sum(q.value < 0.05)                      #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数
sum(q.value < 0.10)                      #FDR = 0.10 (q-value < 0.10)を満たす遺伝子数
sum(q.value < 0.20)                      #FDR = 0.20 (q-value < 0.20)を満たす遺伝子数
sum(q.value < 0.30)                      #FDR = 0.30 (q-value < 0.30)を満たす遺伝子数
```

帰無仮説(G2 = G3)

まずはコピペ実行。結果は予想通りG1 vs. G3の結果と似ており妥当。つまり…

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに作成。「G2群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みではうまくG2 vs. G3をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 + a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2, a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

```
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納

#本番(DEG検出)
#design <- model.matrix(~ as.factor(data.cl
design <- model.matrix(~ 0 + as.factor(data
tcc <- estimateDE(tcc, test.method="edger",
                  design=design, contrast=p
result <- getResult(tcc, sort=FALSE)    #p値
design                                   #デ

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalize
tmp <- tmp[order(tmp$rank),]             #発現
write.table(tmp, out_f, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-v
sum(q.value < 0.05)                      #FDR
sum(q.value < 0.10)                      #FDR
sum(q.value < 0.20)                      #FDR
sum(q.value < 0.30)                      #FDR
```

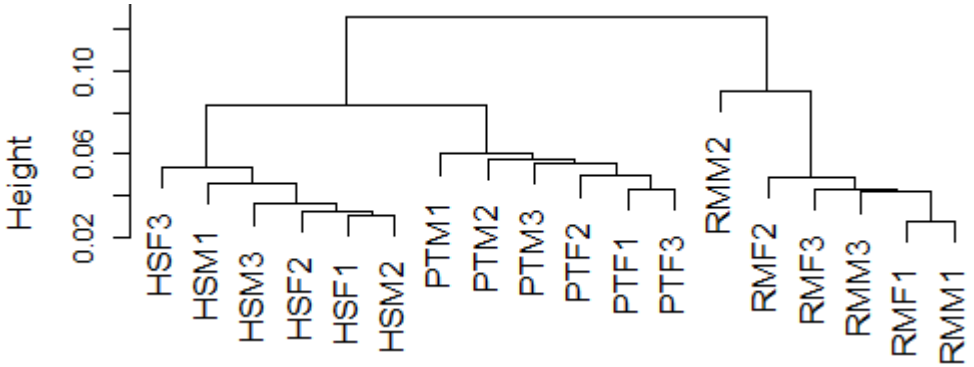
R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]             #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value              #q-valueをq.va$
> sum(q.value < 0.05)                      #FDR = 0.05 (q$
[1] 5451
> sum(q.value < 0.10)                      #FDR = 0.10 (q$
[1] 6514
> sum(q.value < 0.20)                      #FDR = 0.20 (q$
[1] 8041
> sum(q.value < 0.30)                      #FDR = 0.30 (q$
[1] 9408
>
```

①

結果のまとめ

④G2 vs. G3の空白の結果が埋まり、③の結果と似ていることが確認できたということ



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
```

①

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

③

G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

④

帰無仮説(G2 = G3)

①出力ファイルを眺め（赤枠内の上位10個）、確かに「②G2群 vs. ③G3群」の結果になっていることを確認し安心する。スライドを見るだけ

- 帰無仮説(G2 = G3)
  「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに作成。「G2群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みではうまくG2 vs. G3をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 + a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2, a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

rownames(tc	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDE
ENSG00000209449	0	0	1.1	0	3.8	1	0.8	1	0	2.1	2	0	656	857	377	718	###	551	ENS	NA	NA	1.63E-78	3.37E-74	1	1
ENSG00000208587	1.1	1.2	4.5	0	0	0	###	###	###	###	###	###	4.5	2.3	1.4	4.2	9.2	3.5	ENS	NA	NA	7.16E-76	5.73E-72	2	1
ENSG00000220688	27	12	29	27	17	21	201	362	244	424	566	416	0	1.1	0	0	0	0.9	ENS	NA	NA	8.31E-76	5.73E-72	3	1
ENSG00000208978	98	173	118	192	66	142	9.7	19	6.1	16	3	1.1	###	###	###	###	###	###	ENS	NA	NA	2.90E-74	1.50E-70	4	1
ENSG00000134201	16	7.1	18	2.6	15	12	0	1	0	0	0	0	###	###	###	###	###	###	ENS	NA	NA	1.20E-72	4.97E-69	5	1
ENSG00000134391	281	473	201	109	368	449	0	1	0	0	1	0	725	618	451	603	766	373	ENS	NA	NA	2.80E-69	9.59E-66	6	1
ENSG00000112667	159	189	220	125	183	208	291	127	202	270	221	391	0	0	2.1	0	1.3	0.9	ENS	NA	NA	3.24E-69	9.59E-66	7	1
ENSG00000209007	0	0	0	0	0	1	0.8	0	0	0	4	0	627	881	405	483	###	440	ENS	NA	NA	7.11E-67	1.84E-63	8	1
ENSG00000145244	1.1	0	1.1	0	1.9	3	0	3.1	1	0	1	0	216	363	355	263	266	174	ENS	NA	NA	1.42E-65	3.27E-62	9	1
ENSG00000156222	362	510	268	303	424	824	543	266	453	437	428	###	0.9	2.3	0	0	1.3	0.9	ENS	NA	NA	1.58E-61	3.26E-58	10	1

②

③

デザイン行列の説明

①で作成したデザイン行列designについて、②を追加した意味を説明

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに作成。「G2群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みではうまくG2 vs. G3をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 + a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2, a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1に
ます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いませ
きているからです。尚、full modelに相当するデザイン行列の作成手順
model.matrix関数実行時に「0 + 」を追加しています。これによって、最
基準にして作成したデザイン行列ではなく、各群が各列になるようにして

```
normalized <- getNormalizedData(tcc)    #正規化後のデー

#本番(DEG検出)
#design <- model.matrix(~ as.factor(data.cl))#デザイン
design <- model.matrix(~ 0 + as.factor(data.cl))#デザ
tcc <- estimateDE(tcc, test.method="edger", FDR=p, am
                  design=design, contrast=param_contr
result <- getResult(tcc, sort=FALSE)    #p値などの結果を
design                                   #デザイン行列の

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result)
tmp <- tmp[order(tmp$rank),]             #発現変動順にソ
write.table(tmp, out_f, sep="\t", append=F, quote=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value              #q-valueをq.va
sum(q.value < 0.05)                      #FDR = 0.05 (q
sum(q.value < 0.10)                      #FDR = 0.10 (q
sum(q.value < 0.20)                      #FDR = 0.20 (q
sum(q.value < 0.30)                      #FDR = 0.30 (q
```

① ②

R Console

```
> design
   as.factor(data.cl)1 as.factor(data.cl)2 as.factor(data.cl)3
1                    1                   0                   0
2                    1                   0                   0
3                    1                   0                   0
4                    1                   0                   0
5                    1                   0                   0
6                    1                   0                   0
7                    0                   1                   0
8                    0                   1                   0
9                    0                   1                   0
10                   0                   1                   0
11                   0                   1                   0
12                   0                   1                   0
13                   0                   0                   1
14                   0                   0                   1
15                   0                   0                   1
16                   0                   0                   1
17                   0                   0                   1
18                   0                   0                   1
attr(,"assign")
[1] 1 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(data.cl)`
[1] "contr.treatment"

> |
```

デザイン行列の説明

赤下線の「0 +」追加によって、①G1、②G2、③G3の群ごとに計3列が作成され、入力データの対応する列のところに1（それ以外は0）が付与されたデザイン行列となる

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4をベースに作成したG2群vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みではうまくG2 vs. G3をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 + a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2, a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的にはmodel.matrix関数実行時に「0 +」を追加していない基準にして作成したデザイン行列ではなく、各群

```
normalized <- getNormalizedData(tcc

#本番(DEG検出)
#design <- model.matrix(~ as.factor
design <- model.matrix(~ 0 + as.fac
tcc <- estimateDE(tcc, test.method=
                  design=design, co
result <- getResult(tcc, sort=FALSE
design

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), r
tmp <- tmp[order(tmp$rank),]
write.table(tmp, out_f, sep="\t", a

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)
```

R Console

```
> design
   as.factor(data.cl)1 as.factor(data.cl)2 as.factor(data.cl)3
1                    1                   0                   0
2                    1                   0                   0
3                    1                   0                   0
4                    1                   0                   0
5                    1                   0                   0
6                    1                   0                   0
7                    0                   1                   0
8                    0                   1                   0
9                    0                   1                   0
10                   0                   1                   0
11                   0                   1                   0
12                   0                   1                   0
13                   0                   0                   1
14                   0                   0                   1
15                   0                   0                   1
16                   0                   0                   1
17                   0                   0                   1
18                   0                   0                   1
```

コントラストの説明

次は、①で指定したコントラスト情報の説明。この情報からどうやって今設定したい帰無仮説（G2 = G3）を作成するのかという話。ここで指定している(0, −1, 1)は、「$a_1 \times G1 + a_2 \times G2 + a_3 \times G3 = 0$」の係数aの部分に相当する。「$0 \times G1 + (-1) \times G2 + 1 \times G3 = 0$」→「−G2 + G3 = 0」→「G2 = G3」という流れでめでたく帰無仮説G2 = G3の設定完了。
尚、この数値ベクトルの要素数は3だが、それは群数（G1, G2, G3）と同じ

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4
G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coef
をうまく表現できないので、コントラストという別の枠組みで指定しています。思考
a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を
a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで
ます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由
きているからです。尚、full modelに相当するデザイン行列の作成手順も若干異
model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目
基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(0, -1, 1)  ①          #コントラスト情報を指定(reduced model作成用)

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納
```

コントラストの説明

最終的な目的は「G2 = G3」を設定することなので、①のところを(0, −1, 1)の代わりに(0, 1, −1)としてもよい。「$a_1 \times G1 + a_2 \times G2 + a_3 \times G3 = 0$」→「$0 \times G1 + 1 \times G2 + (-1) \times G3 = 0$」→「G2 − G3 = 0」→「G2 = G3」とできるから

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題4
G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coef
をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 +
a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2,
a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 +」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(0, -1, 1) ①           #コントラスト情報を指定(reduced model作成用)

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をt
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納
```

コントラストの説明

もちろん比較したい群を1 or −1で対比（コントラスト）させ、それ以外を0とする、という思考回路でもよい。
もしこの思考回路で行き詰ったら、そのときにまた自分が納得しやすい理解のしかたをすればよい

- 帰無仮説(G2 = G3)

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題[illegible] G3群」の比較を行いたいときの指定方法(帰無仮説: G2 = G3)です。param_coefの枠組みではうまくG2 vs. G3をうまく表現できないので、コントラストという別の枠組みで指定しています。思考回路としては、「a1*G1 + a2*G2 + a3*G3 = 0」として、この場合の目的である「G2 = G3という帰無仮説」を作成できるように、係数a1, a2, a3に適切な数値を代入することです。ここではa1 = 0、a2 = -1, a3 = 1にすることで目的の帰無仮説を作成できます。以下ではc(0, -1, 1)と指定していますが、c(0, 1, -1)でも構いません。理由はどちらでもG2 = G3を表現できているからです。尚、full modelに相当するデザイン行列の作成手順も若干異なります。具体的には、model.matrix関数実行時に「0 + 」を追加しています。これによって、最初の1列目が全て1になるようなG1群を基準にして作成したデザイン行列ではなく、各群が各列になるようにしています。

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(0, -1, 1)             #コントラスト情報を指定(reduced model作成用)

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をt
normalized <- getNormalizedData(tcc)      #正規化後のデータを取り出してnormalizedに格納
```

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

コントラストでG1 = G2

①ここです。②「$a_1 \times G1 + a_2 \times G2 + a_3 \times G3 = 0$」→「$1 \times G1 + (-1) \times G2 + 0 \times G3 = 0$」→「G1 − G2 = 0」→「G1 = G2」という思考回路。
③コピペ実行結果は以前（スライド48）と同じ

- コントラストでG1 = G2 ①

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題5をベー
で「G1群 vs. G2群」の比較を行いたいときの指定方法(帰無仮説: G1 = G2)です。ここではa1 = 1、a2 = -1, a3 = 0にすることで目的の帰無仮説を作成できます。以下ではc(1, -1, 0)と指定していますが、c(-1, 1, 0)でも構いません。理由はどちらでもG1 = G2を表現できているからです。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(1, -1, 0)           #コン

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.n

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

②

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 2282
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 3077
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 4334
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 5445
>
```

③

コントラストでG1 = G3

①ここです。②「a_1 × G1 + a_2 × G2 + a_3 × G3 = 0」→「1 × G1 + 0 × G2 + (−1) × G3 = 0」→「G1 − G3 = 0」→「G1 = G3」という思考回路。
③コピペ実行結果は以前（スライド54）と同じ

印刷版はG1=G2の説明になってますm(_ _)m

- コントラストでG1 = G3 ①

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun 2013)」の例題6をベー
で「G1群 vs. G3群」の比較を行いたいときの指定方法(帰無仮説: G1 = G3)です。ここではa1 = 1、a2 = 0, a3 = -1にすることで目的の帰無仮説を作成できます。以下ではc(1, 0, -1)と指定していますが、c(-1, 0, 1)でも構いません。理由はどちらでもG1 = G3を表現できているからです。

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_contrast <- c(1, 0, -1)           #コン

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)        #TCC

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tm
                       iteration=3, FDR=0.1
normalized <- getNormalizedData(tcc)    #正規
```

②

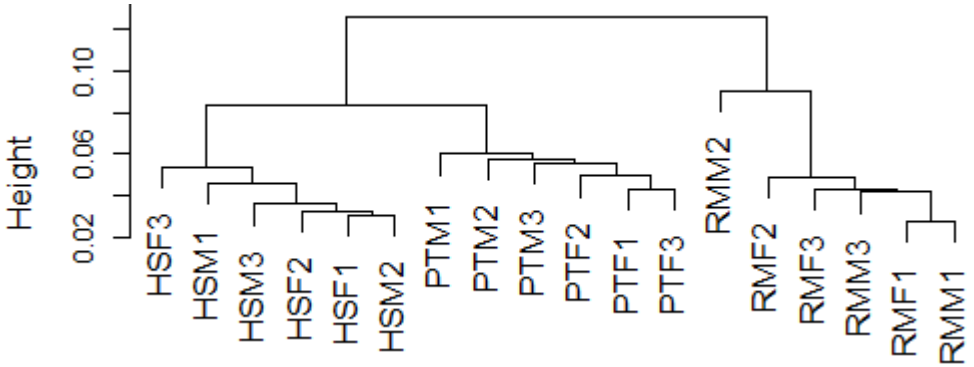
R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, $
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 5278
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 6324
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 7816
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 9069
>
```

③

コントラストの限界

ここまでG1群を基準としないデザイン行列を用いて、コントラスト情報を与えて②～④を行った。直感的にはコントラスト情報を与えるほうがわかりやすいことは確かだろう



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
```

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

③

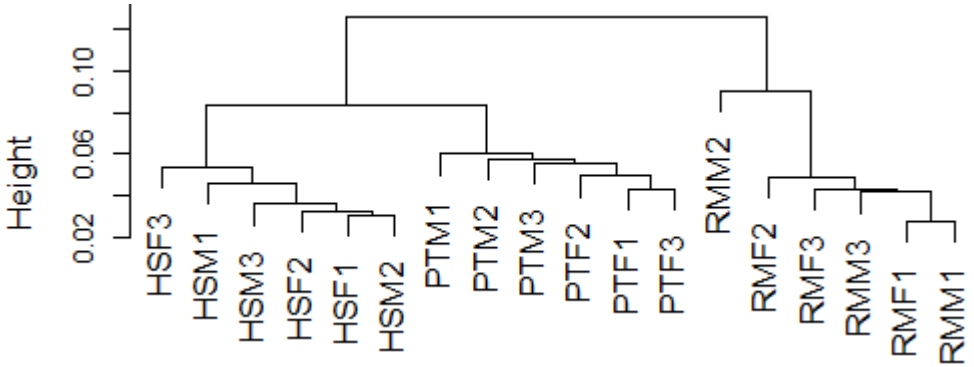
G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

④

コントラストの限界

ではなぜわかりにくい(←個人の感想です)G1群をベースとしたデザイン行列のほうを先に教えたのか?それは①G1 = G2 = G3という帰無仮説は、コントラストの枠組みでは表現できない(ANOVA的な解析はできない)からです。3群以上の比較の場合は、通常ANOVA的な解析(どこかの群間で発現変動している遺伝子)を先に行います



G1(HS)群 6 samples　G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
```

① デザイン行列(G1群をベース)

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

③

G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

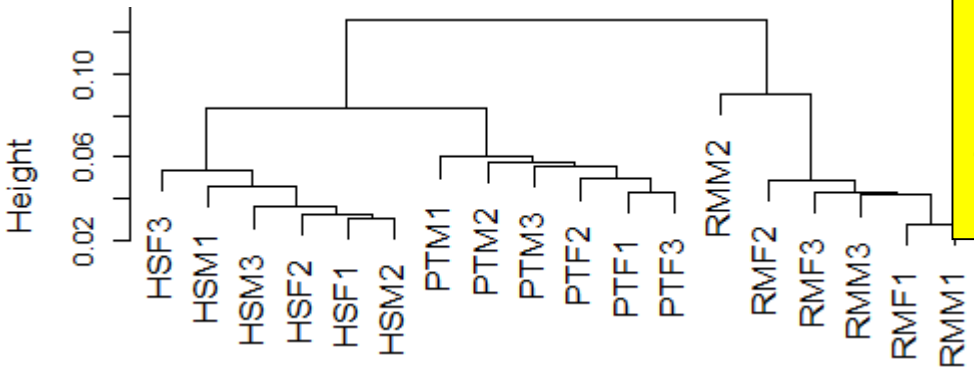
④

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

Post-hoc testの…

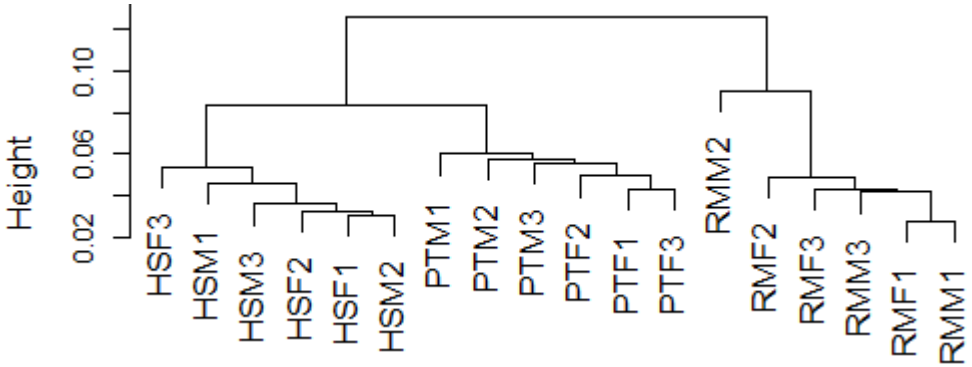
ここまでの結果は、G1, G2, G3群の全てのデータを用いてモデル構築（バラツキの程度の見積もり）を行った結果をベースとしている。つまり、例えば②G1 vs. G2の解析結果は、G3の情報もモデル構築時に利用しています。それゆえ、G1とG2のデータのみを用いて単独で2群間比較を行った結果とは異なります!このあたりは、疲れない程度に手を抜いて聞くところ



G1(HS)群 6 samples	G2(PT)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 7236 > sum(q.value < 0.10) [1] 8485 ①
G1(HS)群 6 samples	G2(PT)群 6 samples		> sum(q.value < 0.05) [1] 2282 > sum(q.value < 0.10) [1] 3077 ②
G1(HS)群 6 samples		G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5278 > sum(q.value < 0.10) [1] 6324 ③
	G2(PT)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5451 > sum(q.value < 0.10) [1] 6514 ④

今議論したいのは、3群間比較ではなく2群間比較なので、①のG1 vs. G2 vs. G3は議論の対象から外す

Post-hoc testの…



G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

②

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

③

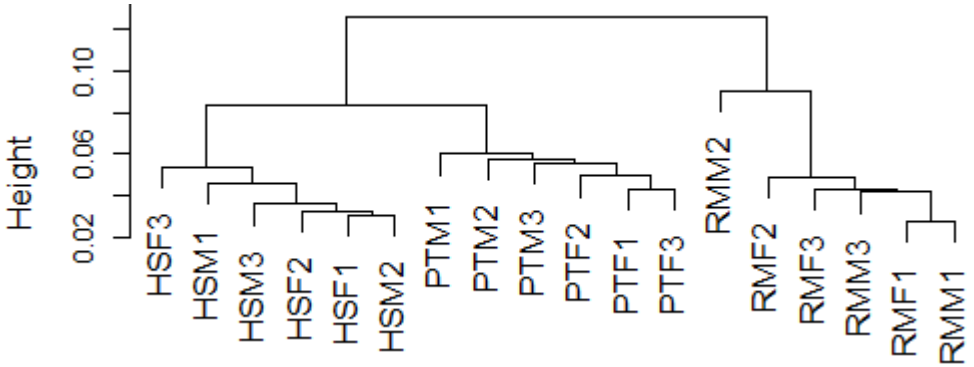
G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

④

Post-hoc testの…

①3群全ての情報を使って2群間比較を行ったときの結果。②比較対象の2群のデータのみを使って2群間比較を行い、①と②の違いについて議論するのが目的



3群全ての情報を利用 ①　　比較する2群の情報のみ ②

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

HS vs. PT（2群間比較）

①計18サンプルの3群間比較用データから、12サンプル分だけ抽出し、6 HS samples vs. 6 PT samples の2群間比較をコピペ実行した結果

- HS vs. PT (2群間比較)

「解析 | 発現変動 | 2群間 | 対応なし | 複製あり | Blekhmanデータ | TCC(Sun 2013)」の例題7をベースに作成。
1:6, 7:12 列目のデータのみ抽出し、通常の6 HS samples vs. 6 PT samplesの2群間比較を行っています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge_2_HS_PT.txt"              #出力ファイル名を指定してout_f1に格納
param_subset <- c(1:6, 7:12)              #取り扱いたいサブセット情報を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(サブセットの抽出とTCCクラスオブジェクト
data <- data[,param_subset]               #para
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)          #TCC
colnames(data)                            #列名

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1,
```

①

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f1, sep="\t", append=F, quote=F,$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 2480
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 3116
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 4254
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 5114
>
```

HS vs. RM（2群間比較）

①計18サンプルの3群間比較用データから、12サンプル分だけ抽出し、6 HS samples vs. 6 RM samples の2群間比較をコピペ実行した結果

- HS vs. RM (2群間比較)

「解析 | 発現変動 | 2群間 | 対応なし | 複製あり | Blekhmanデータ | TCC(Sun 2013)」の例題7をベースに作
1:6, 13:18 列目のデータのみ抽出し、通常の6 HS samples vs. 6 RM samplesの2群間比較を行っています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge_2_HS_RM.txt"              #出力ファイル名を指定してout_f1に格納
param_subset <- c(1:6, 13:18)             #取り扱いたいサブセット情報を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(サブセットの抽出とTCCクラスオブジェクト
data <- data[,param_subset]               #para
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)          #TCC
colnames(data)                            #列名

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1,
```

①

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f1, sep="\t", append=F, quote=F,$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 4934
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 5823
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 7046
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 7953
>
```

PT vs. RM（2群間比較）

①計18サンプルの3群間比較用データから、12サンプル分だけ抽出し、6 PT samples vs. 6 RM samplesの2群間比較をコピペ実行した結果

- PT vs. RM (2群間比較)

「解析 | 発現変動 | 2群間 | 対応なし | 複製あり | Blekhmanデータ | TCC(Sun 2013)」の例題7をベースに作
7:12, 13:18 列目のデータのみ抽出し、通常の6 PT samples vs. 6 RM samplesの2群間比較を行っています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge_2_PT_RM.txt"              #出力ファイル名を指定してout_f1に格納
param_subset <- c(7:12, 13:18)            #取り扱いたいサブセット情報を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.na

#前処理(サブセットの抽出とTCCクラスオブジェクト
data <- data[,param_subset]               #para
data.cl <- c(rep(1, param_G1), rep(2, param_
tcc <- new("TCC", data, data.cl)          #TCC
colnames(data)                            #列名

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm
                       iteration=3, FDR=0.1
```

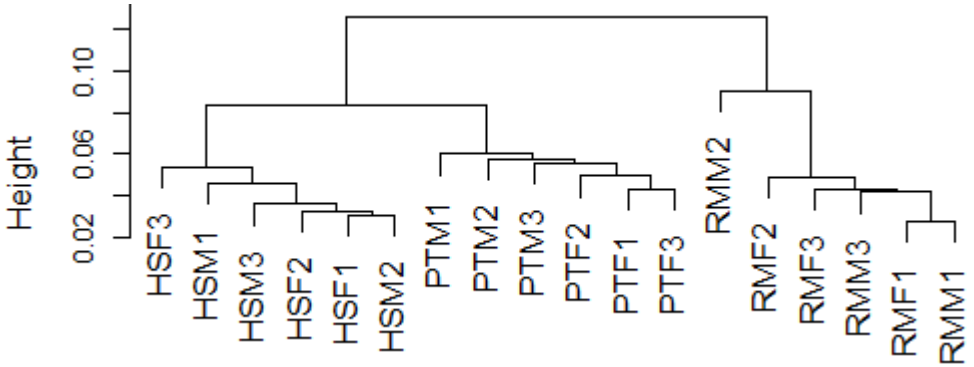
①

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result)$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f1, sep="\t", append=F, quote=F,$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 4502
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 5424
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 6688
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 7676
>
```

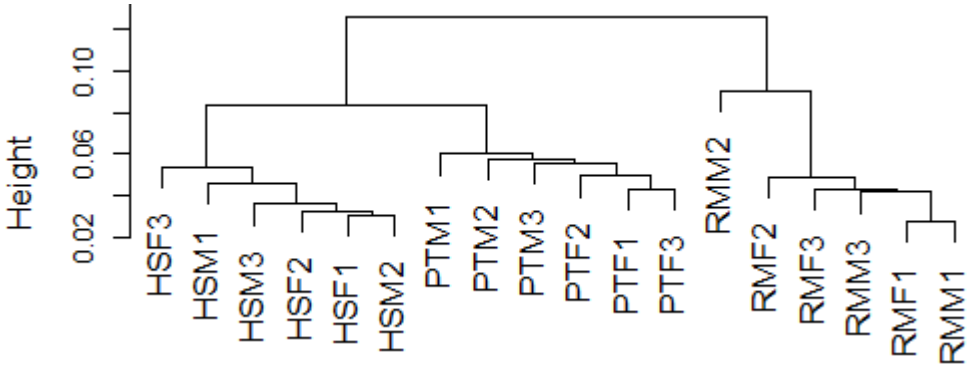
結果のまとめ

①3群全ての情報を使って2群間比較を行ったときの結果。②比較対象の2群のデータのみを使って2群間比較を行い、①と②の違いについて議論するのが目的。とりあえず基礎データは揃った



		3群全ての情報を利用 ①	比較する2群の情報のみ ②
G1(HS)群 6 samples	G2(PT)群 6 samples	> sum(q.value < 0.05) [1] 2282 > sum(q.value < 0.10) [1] 3077	> sum(q.value < 0.05) [1] 2480 > sum(q.value < 0.10) [1] 3116
G1(HS)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5278 > sum(q.value < 0.10) [1] 6324	> sum(q.value < 0.05) [1] 4934 > sum(q.value < 0.10) [1] 5823
G2(PT)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5451 > sum(q.value < 0.10) [1] 6514	> sum(q.value < 0.05) [1] 4502 > sum(q.value < 0.10) [1] 5424

結果のまとめ



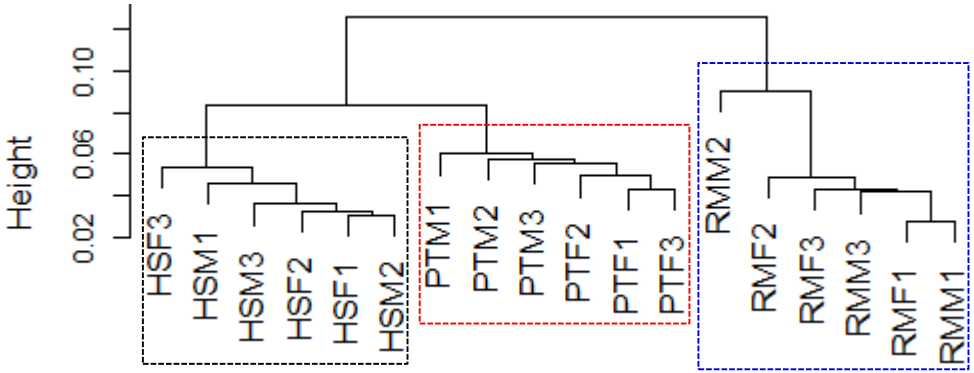
③

私の着眼点（伝えたいこと）は、「① → ②」の遺伝子数が「HS vs. PT」では増え、それ以外（HS vs. RM；およびPT vs. RM）では減るという事実。そしてその結果は、③サンプル間クラスタリング実行段階で予想可能ということ。もちろん①と②で結果が異なるというということ自体も、知らなかったヒトにとっては重要でしょう

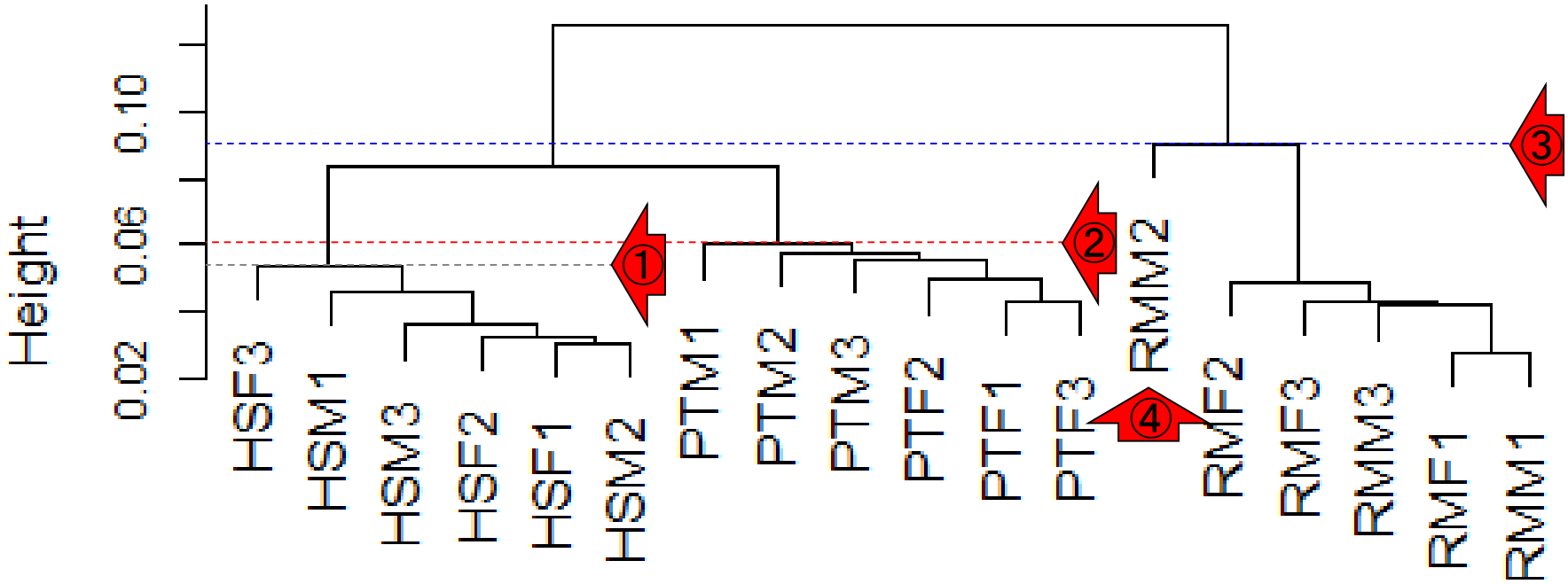
3群全ての情報を利用　①　　比較する2群の情報のみ　②

		①	②
G1(HS)群 6 samples	G2(PT)群 6 samples	> sum(q.value < 0.05) [1] 2282 > sum(q.value < 0.10) [1] 3077	> sum(q.value < 0.05) [1] 2480 > sum(q.value < 0.10) [1] 3116
G1(HS)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5278 > sum(q.value < 0.10) [1] 6324	> sum(q.value < 0.05) [1] 4934 > sum(q.value < 0.10) [1] 5823
G2(PT)群 6 samples	G3(RM)群 6 samples	> sum(q.value < 0.05) [1] 5451 > sum(q.value < 0.10) [1] 6514	> sum(q.value < 0.05) [1] 4502 > sum(q.value < 0.10) [1] 5424

おさらい

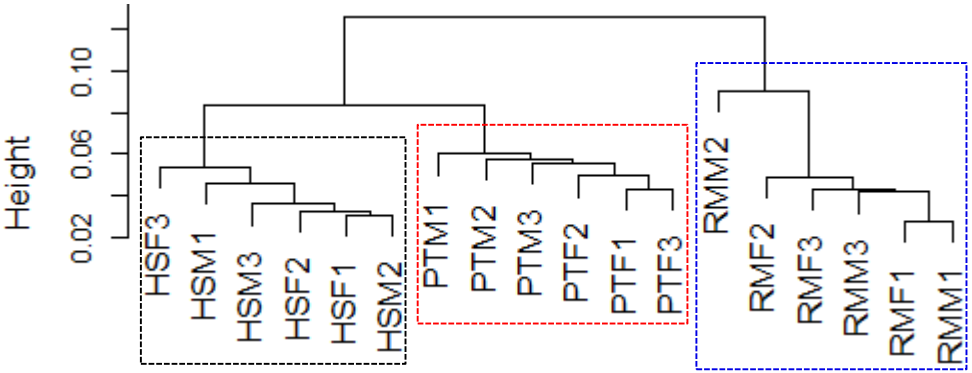


この3群間比較用データは、同一群でクラスターを形成している。①HS群や②PT群は、群内のサンプル間類似度が③RM群に比べて高い（距離が短い or Heightが低い）。④特にRMM2サンプルが、他のRM群との類似度が低いために、全体としてRM群内のバラツキが大きくなっている。それゆえ、RM群がモデル構築（バラツキの見積もり）段階で入っているか否かという観点で結果を眺めなおすとよい



HS vs. PTの解釈

HS vs. PTの結果について解説。①の結果はモデル構築時にバラツキの大きいRM群を含んでいたが、②ではRM群を含んでいないため、FDR閾値を満たす遺伝子数が増えたと解釈すればよい



3群全ての情報を利用　比較する2群の情報のみ

①　②

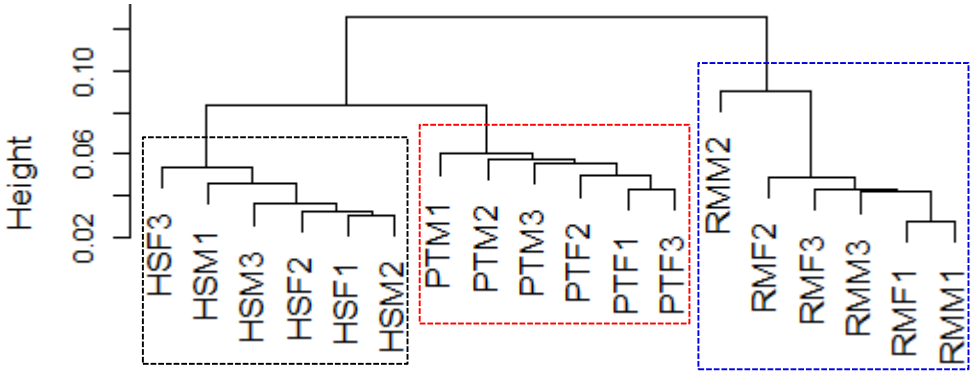
G1(HS)群　G2(PT)群
6 samples　6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

```
> sum(q.value < 0.05)
[1] 2480
> sum(q.value < 0.10)
[1] 3116
```

HS vs. RMの解釈

HS vs. RMの結果について解説。②で遺伝子数が減ったのは、全体としてバラツキが小さくなる方向に寄与していたPTデータが含まれていないから



3群全ての情報を利用　比較する2群の情報のみ

①　②

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

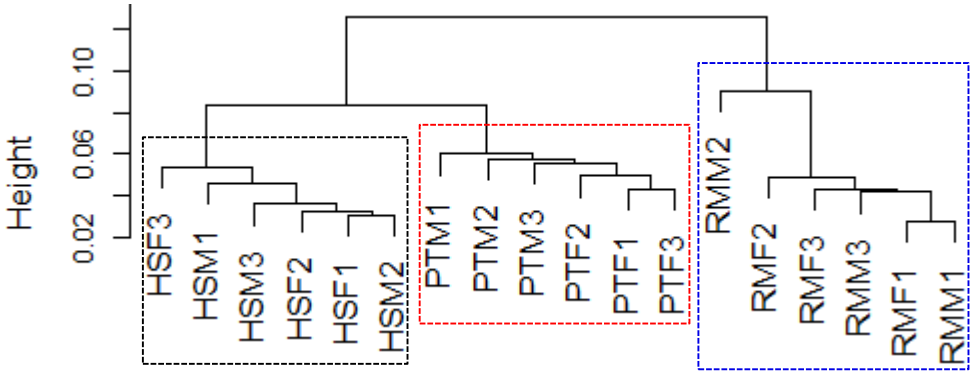
```
> sum(q.value < 0.05)
[1] 2480
> sum(q.value < 0.10)
[1] 3116
```

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

```
> sum(q.value < 0.05)
[1] 4934
> sum(q.value < 0.10)
[1] 5823
```

PT vs. RMの解釈



PT vs. RMの結果について解説。②で遺伝子数が減ったのは、全体としてバラツキが小さくなる方向に寄与していたHSデータが含まれていないから。③HS vs. RMに比べて④PT vs. RMの遺伝子数の減少度合いが大きいのは、HS群内のバラツキがPT群に比べて全体的に小さいため（と私は解釈した）。このあたりは原著論文では、probably due to …とかprobably because …などと記載する

3群全ての情報を利用　比較する2群の情報のみ

①　②

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

```
> sum(q.value < 0.05)
[1] 2480
> sum(q.value < 0.10)
[1] 3116
```

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

```
> sum(q.value < 0.05)
[1] 4934
> sum(q.value < 0.10)
[1] 5823
```

③

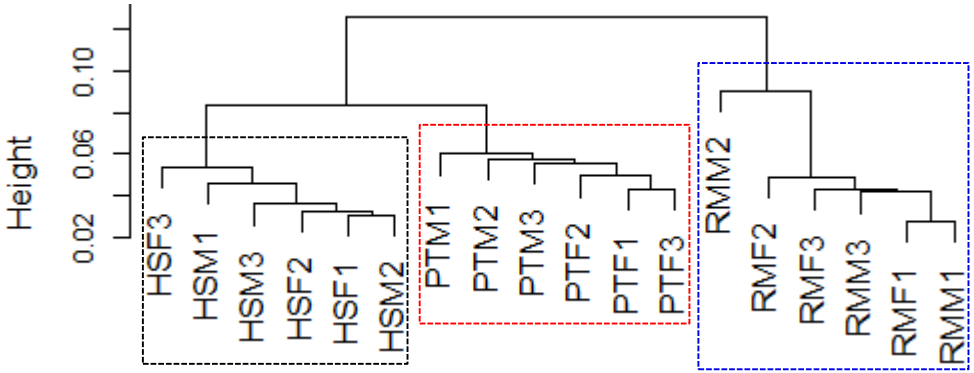
G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

```
> sum(q.value < 0.05)
[1] 4502
> sum(q.value < 0.10)
[1] 5424
```

④

この枠組みの指針…



この枠組み（ANOVA的なことをやってから、総当たりの2群間比較という流れ）で議論を行いたい（特に3群間の結果を基本とする）場合は、**これまで述べてきた事柄を意識しつつ**①の結果のみで議論するのが基本かと思います。もちろんメインストーリー（3群間比較）以外のサブストーリーとして補足情報的に②に関する議論が別にあってもいいとは思います

3群全ての情報を利用　比較する2群の情報のみ

①　②

G1(HS)群 6 samples　G2(PT)群 6 samples

```
> sum(q.value < 0.05)
[1] 2282
> sum(q.value < 0.10)
[1] 3077
```

```
> sum(q.value < 0.05)
[1] 2480
> sum(q.value < 0.10)
[1] 3116
```

G1(HS)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5278
> sum(q.value < 0.10)
[1] 6324
```

```
> sum(q.value < 0.05)
[1] 4934
> sum(q.value < 0.10)
[1] 5823
```

G2(PT)群 6 samples　G3(RM)群 6 samples

```
> sum(q.value < 0.05)
[1] 5451
> sum(q.value < 0.10)
[1] 6514
```

```
> sum(q.value < 0.05)
[1] 4502
> sum(q.value < 0.10)
[1] 5424
```

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

遺伝子クラスタリング

（個人的には結論として非推奨だが）遺伝子間クラスタリングを行っておき、どの遺伝子がどのパターンに属するかという情報を①MBCluster.Seqなどを用いて得ておき、特定のFDR閾値を満たす遺伝子サブセットの分類分けを行えばよい。おそらくこれも王道なので一応紹介

Bioinformatics. 2014 Jan 15;30(2):197-205. doi: 10.1093/bioinformatics/btt632. Epu

Model-based clustering for RNA-seq data.

Si Y[1], Liu P, Li P, Brutnell TP.

Author information

Abstract

MOTIVATION: RNA-seq technology has been widely adopted as an attractive alternative to microarray-based methods to study global gene expression. However, robust statistical tools to analyze these complex datasets are still lacking. By grouping genes with similar expression profiles across treatments, cluster analysis provides insight into gene functions and networks, and hence is an important technique for RNA-seq data analysis.

RESULTS: In this manuscript, we derive clustering algorithms based on appropriate probability models for RNA-seq data. An expectation-maximization algorithm and another two stochastic versions of expectation-maximization algorithms are described. In addition, a strategy for initialization based on likelihood is proposed to improve the clustering algorithms. Moreover, we present a model-based hybrid-hierarchical clustering method to generate a tree structure that allows visualization of relationships among clusters as well as flexibility of choosing the number of clusters. Results from both simulation studies and analysis of a maize RNA-seq dataset show that our proposed methods provide better clustering results than alternative methods such as the K-means algorithm and hierarchical clustering methods that are not based on probability models.

①

AVAILABILITY AND IMPLEMENTATION: An R package, MBCluster.Seq, has been developed to implement our proposed algorithms. This R package provides fast computation and is publicly available at http://www.r-project.org

遺伝子クラスタリング

①MBCluster.Seq単体での利用はこちら。②例題4。③K-meansクラスタリングの一種なので、クラスター数を指定する。50など比較的大きめの値にしても、non-redundantにしてくれるので、最終的に得られるクラスター数は（データの性質にもよるが通常は）減る

- 解析 | クラスタリング | について (last modified 2016/02/12) NEW
- 解析 | クラスタリング | サンプル間 | hclust (last modified 2015/02/26)
- 解析 | クラスタリング | サンプル間 | TCC(Sun_2013) (last modified 2015/11/15)
- 解析 | クラスタリング | 遺伝子間(基礎) | MBCluster.Seq(Si_2014) (last modified 2016/02/1
- 解析 | クラスタリング | 遺伝子間(応用) | MBCluster.Seq(Si_2014)+TCC正規化(Sun_2013)
- 解析 | シミュレーションカウントデータ | について (last modified 2015/11/10)
- 解析 | シミュレー
- 解析 | シミュレー
- 解析 | シミュレー
- 解析 | シミュレー
- 解析 | シミュレー
- 解析 | フィルタリ
- 解析 | 発現変動

①

解析 | クラスタリング | 遺伝子間(基礎) | MBCluster.Seq(Si_2014)

MBCluster.Seqパッケージを用いたやり方を示します。k-means++(Arthur and Vassilvitskii, 2007) と似た方法でクラスター中心を決める方法を内部的に利用しているらしいです。param_clust_numで指定するクラスター数は最初は気持ち多めにしておいても、クラスタリング実行中に重複のないパターン(non-redundant expression patterns)にある程度はしてくれます。例えば、これを利用した最近の論文
終的に20 clust
利用だと思いま
Normalizer=NULL
化係数、logFC、N
入力ファイルや指定
「ファイル」-「ディレ

②

1. サンプルデータ

Biological replica
DEG (最初の180
既知です。このテ
スター)しか存在し

```
in_f <- "dat
out_f <- "ho
param_fig <-
param_clust_
param_G1 <-
param_G2 <-
```

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこに割り振られたかの情報を、クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout_f1に格納(pngファイル)
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout_f2に格納(txtファイル)
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 10                     #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したフ
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G3
dim(data)                                 #オブジェクトdataの行数と列数を表示
```

③

遺伝子クラスタリング

①通常のクラスタリングの際は、入力データのどの列がどの群かという情報を与えない。
②MBCluster.Seqは、同一群内のバラツキを考慮してくれるので、列のラベル情報を与えている。Model-based clusteringというのは、（発現変動解析時にnon-DEGの分布を見積もるのと同様に）同一群内のバラツキを超えた意味のある発現パターンを返してくれると理解すればよい

Bioinformatics. 2014 Jan 15;30(2):197-205. doi: 10.1093/bioinformatics/btt632. Epub 201

Model-based clustering for RNA-seq data.

②

Si Y[1], Liu P, Li P, Brutnell TP.

Author information

Abstract

MOTIVATION: RNA-seq technology has been widely adopted as an attrac
based methods to study global gene expression. However, robust statistical tools to analyze these
complex datasets are still lacki
cluster analysis provides insigh
for RNA-seq data analysis.

RESULTS: In this manuscript,
for RNA-seq data. An expectat
expectation-maximization algo
likelihood is proposed to impro
hierarchical clustering method
among clusters as well as flexi
studies and analysis of a maiz
clustering results than alternati
methods that are not based on

AVAILABILITY AND IMPLEM
implement our proposed algori
at http://www.r-project.org

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこに割り振られたかの情報を、クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout_f1に格納(pngファイル)
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout_f2に格納(txtファイル)
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 10                     #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したこ
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
dim(data)                                 #オブジェクトdataの行数と列数を表示
```

①

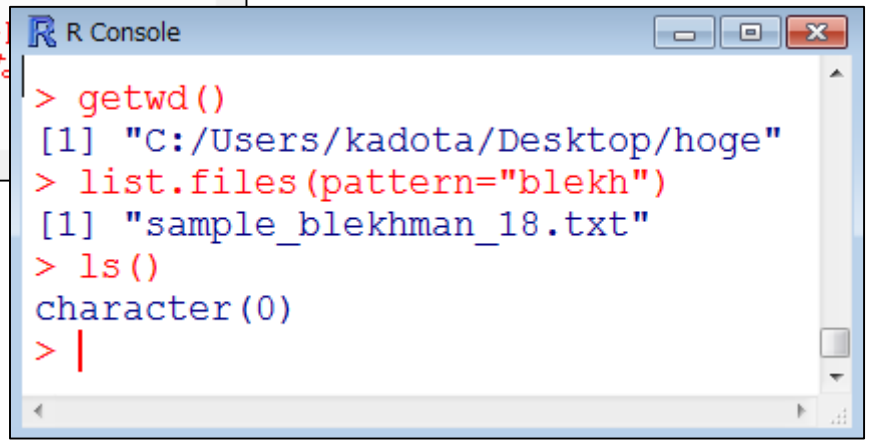
コピペ。約18分

コピペ

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこに割り振られたかの情報を、クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout_f1に格納(pngファイル)
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout_f2に格納(txtファイル)
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 10                     #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定したこ
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G3
dim(data)                                 #オブジェクトdataの行数と列数を表示

#前処理(基礎情報取得)
hoge <- RNASeq.Data(data, Normalizer=NULL,#クラスタリングに必要な基礎情報を取
          Treatment=data.cl, GeneID=rownames(data))#クラスタリングに必要な

#本番(クラスタリング)
```



結果の説明

①800×500ピクセルの②hoge4.pngを内部的に乱数を発生させているので、見栄えはヒトによって異なる

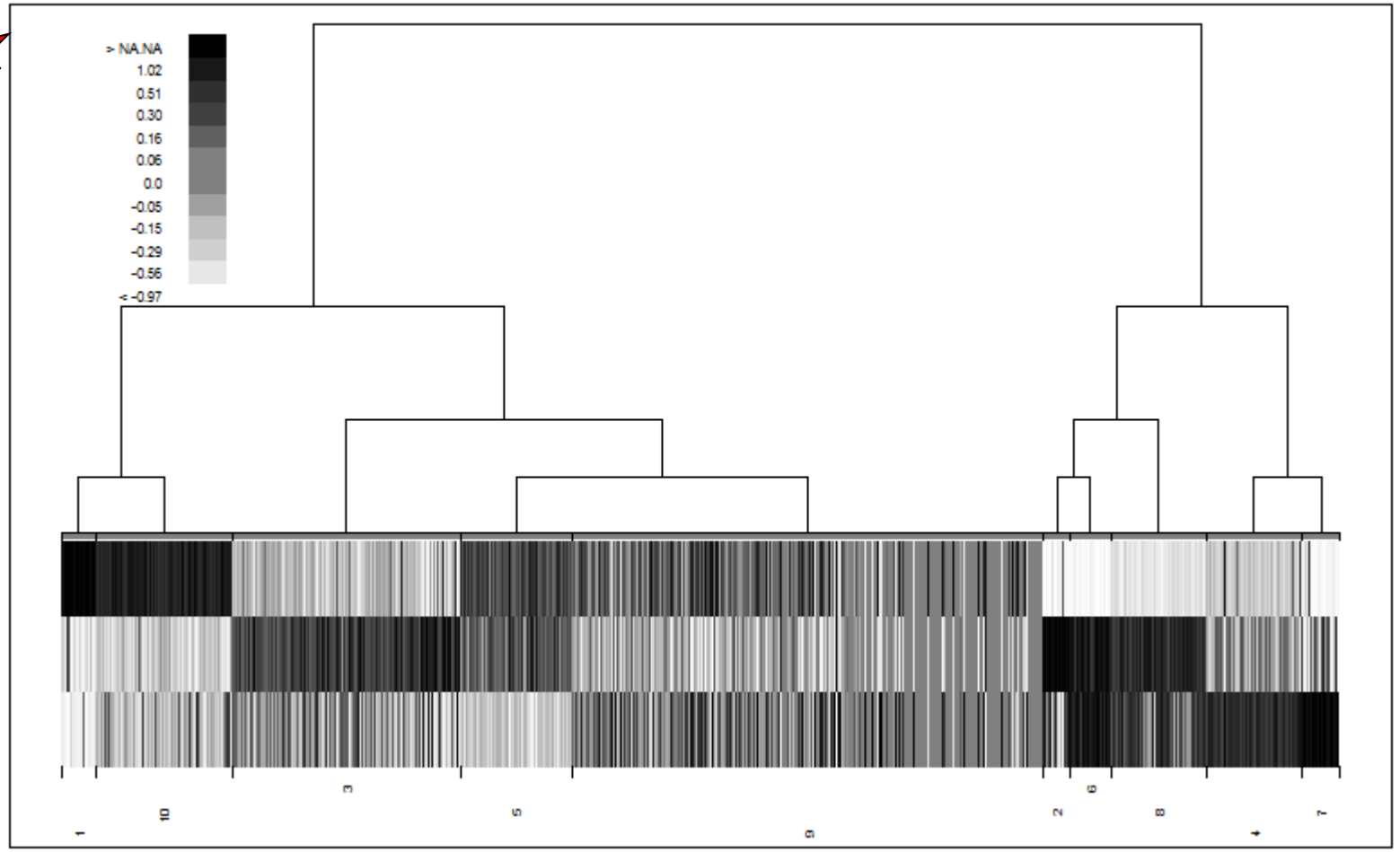
4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこに割り振られたかの情報を、クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhma 18.txt"        #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"
out_f2 <- "hoge4.txt"
param_fig <- c(800, 500)
param_clust_num <- 10
param_G1 <- 6
param_G2 <- 6
param_G3 <- 6

#必要なパッケージをロード
library(MBCluster.Seq)

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

#前処理(基礎情報取得)
hoge <- RNASeq.Data(data,
           Treatment=data.

#本番(クラスタリング)
```

②

①



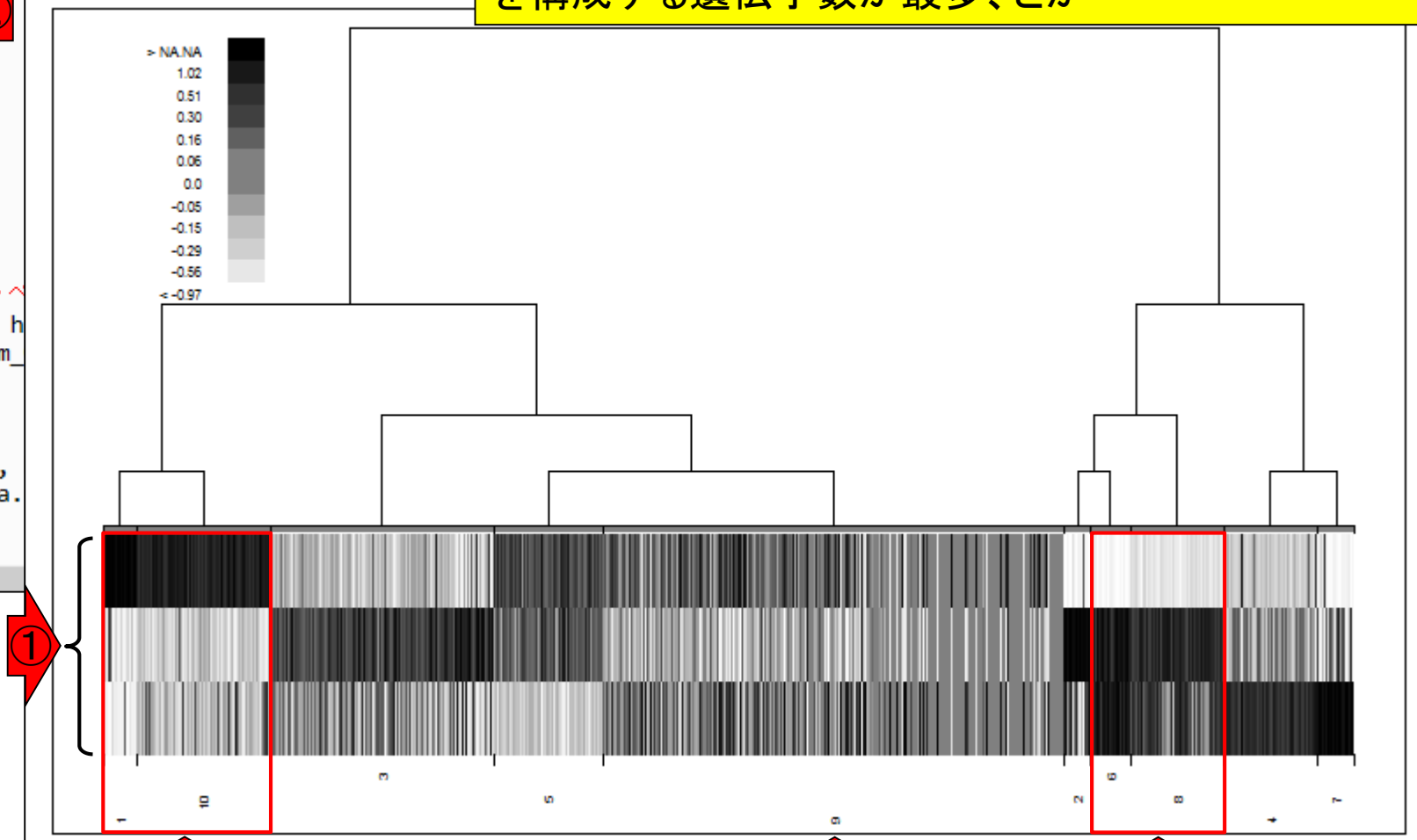
結果の説明

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスター
クラスターごとにソートして保存しています。とりあえずクラスター数を10にして

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定
out_f1 <- "hoge4.png"
out_f2 <- "hoge4.txt"
param_fig <- c(800, 500
param_clust_num <- 10
param_G1 <- 6
param_G2 <- 6
param_G3 <- 6

#必要なパッケージをロード
library(MBCluster.Seq)

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

#前処理(基礎情報取得)
hoge <- RNASeq.Data(data,
            Treatment=data.

#本番(クラスタリング)
```

②



①下から順にG1(HS)、G2(PT)、G3(RM)。順序は、②のテキストファイル(後述)と見比べることでわかる。③cluster 1と10はG3群で高発現、④cluster 6と8はG3群で低発現パターンのものだ、みたいに解釈する。全体を眺めることで各クラスタを構成するメンバー数(遺伝子数)の概要をつかめる。例えば、⑤cluster 9を構成する遺伝子数が最多、とか

①

③

⑤

④

結果の説明

①テキストファイル（hoge4.txt）の中身。②1番右側の列がクラスター番号情報。③例題4の場合、出力ファイルはクラスター番号順にソートされている。確かにデンドログラム（樹形図）でみた通り、cluster 1はG3(RM)群で高発現パターンになっている

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスター
クラスターごとにソートして保存しています。とりあえずクラスター数を10にして

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout_f1に格納(pngファイル)
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout_f2に格納(txtファイル)
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 10                     #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

#必要なパッケージをロード
```

rownames(d	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3	cls$cluster
ENSG00000002726	1	41	56	41	1	4	1	4	0	0	0	0	17492	25226	10548	18797	23216	11485	1
ENSG00000006747	0	1	1	2	0	1	0	2	1	0	0	0	23	8	12	6	1	10	1
ENSG00000007174	1	0	0	0	0	1	1	1	0	0	0	1	55	6	63	52	82	69	1
ENSG00000016490	0	0	0	0	0	0	0	0	0	0	0	0	4	0	4	0	0	0	1
ENSG00000017483	4	1	11	3	3	3	17	10	6	129	4	4	128	212	206	193	118	121	1
ENSG00000039600	1	0	0	0	0	0	0	1	0	0	1	5	19	16	50	18	3	20	1
ENSG00000042813	0	0	0	0	0	0	1	1	0	0	0	0	6	7	10	13	7	4	1
ENSG00000048545	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	1	1
ENSG00000049768	2	3	3	6	2	6	0	1	1	0	2	20	100	131	94	84	13	78	1

コードの解説

①テキストファイル（hoge4.txt）作成部分のコード。②dataオブジェクトは入力ファイルを読み込んだ直後のものなので、正規化前のデータ。③cls$clusterという数値ベクトルが、入力データの遺伝子の並び順に、どの遺伝子がどのクラスターに属するかを示した情報に相当する。出力がクラスター番号順になっている理由は、④cls$clusterの並びでソートしているから。門田亡き後もこのようにコードの中身を自力で解読できるようになっておけば大丈夫

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定し
クラスターごとにソートして保存しています。とりあえずクラスター

```
#前処理(基礎情報取得)
hoge <- RNASeq.Data(data, Normalizer=NULL,#クラ
          Treatment=data.cl, GeneID=rownames(da

#本番(クラスタリング)
c0 <- KmeansPlus.RNASeq(data=hoge, nK=param_clust_num,#K-means clusteringのクラスター中
          model="nbinom", print.steps=F)#K-means clusteringのクラスター中心の初期値を取
cls <- Cluster.RNASeq(data=hoge, model="nbinom",#クラスタリング
          centers=c0$centers, method="EM")#クラスタリング
tr <- Hybrid.Tree(data=hoge, model="nbinom",cluster=cls$cluster)#hybrid-hierarchical
table(cls$cluster)                    #param_clust_numで指定したパターンに属する遺伝子数

#ファイルに保存(pngファイル)
png(out_f1, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイルの各種パ
plotHybrid.Tree(merge=tr, cluster=cls$cluster, logFC=hoge$logFC, tree.title=NULL)#描画
dev.off()                             #おまじない

#ファイルに保存(txtファイル)
tmp <- cbind(rownames(data), data  cls$cluster)#保存したい情報をtmpに格納
tmp <- tmp[order(cls$cluster),]       #クラスター番号順にソートした結果をtmpに格納
write.table(tmp, out_f2, sep="\t"  append=F, quote=F, row.names=F)#tmpの中身をout_f2で
```

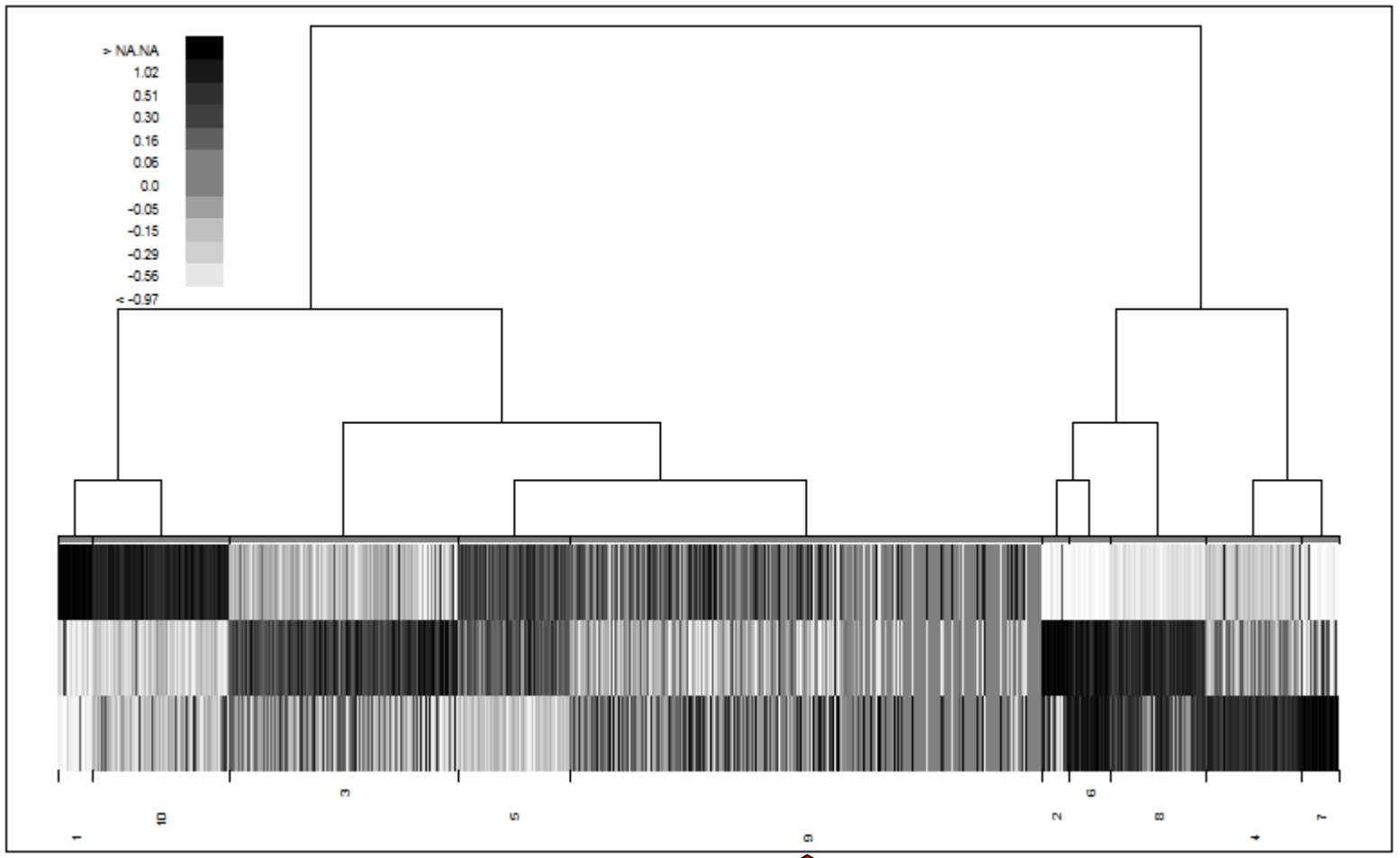
クラスタごとの遺伝子数

hoge4.pngを眺めることで、①cluster 9を構成する遺伝子数が最多とかが一応わかるが、各クラスタを構成するメンバー数(遺伝子数)を正確に調べるやり方を伝授

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこに割り振られたかの情報を、クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"
out_f2 <- "hoge4.txt"
param_fig <- c(800, 500)
param_clust_num <- 10
param_G1 <- 6
param_G2 <- 6
param_G3 <- 6

#必要なパッケージをロード
library(MBCluster.Seq)

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

#前処理(基礎情報取得)
hoge <- RNASeq.Data(data,
            Treatment=data.

#本番(クラスタリング)
```



①

クラスタごとの遺伝子数

入力データの遺伝子の並び順にどの遺伝子がどのクラスターに属するかを示したcls$clusterという数値ベクトルを入力として、①table関数を実行。結果を眺めると、確かに②cluster 9の遺伝子数が最多(7,617個)であり、この結果が妥当であることが分かる。この結果も乱数を発生させているので、ヒトによって異なる。③補足情報

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じですが、遺伝子ごとにparam_clust_numで指定したクラスターのどこ
クラスターごとにソートして保存しています。とりあえずクラスター数を10にしています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してou
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してou
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 10                     #クラスター数を指定
param_G1 <- 6                             #G1群のサン
param_G2 <- 6                             #G2群のサン
param_G3 <- 6                             #G3群のサン

#必要なパッケージをロード
library(MBCluster.Seq)                    #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
dim(data)                                 #オブジ

#前処理(基礎情報取得)
hoge <- RNASeq.Data(data, Normalizer=NULL,#クラス
          Treatment=data.cl, GeneID=rownames(dat

#本番(クラスタリング)
```

①

R Console

```
> dim(data)
[1] 20689    18
> length(cls$cluster)
[1] 20689
> head(cls$cluster)
[1] 5 3 3 3 4 3
> table(cls$cluster)

    1     2     3     4     5     6     7     8     9    10
  565   437  3697  1555  1807   654   621  1535  7617  2201
> x <- c("kk", "apu", "kk", "iu", "iu", "kk")
> x
[1] "kk"  "apu" "kk"  "iu"  "iu"  "kk"
> table(x)
x
apu  iu  kk
  1   2   3
>
```

②

③

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

TCC正規化を合わせる 参考

①TCC正規化後のデータを入力としてMBCluster.Seqを実行するやり方。②基本戦略や手順。③例題4。HPCI講習会(2016.03.03-04)から説明文を修正してあります。スライドを見るだけ

- 解析 | クラスタリング | について (last modified 2016/02/12) NEW
- 解析 | クラスタリング | サンプル間 | hclust (last modified 2015/02/26)
- 解析 | クラスタリング | サンプル間 | TCC(Sun_2013) (last modified 2015/11/15)
- 解析 | クラスタリング | 遺伝子間(基礎) | MBCluster.Seq(Si_2014) (last modified 2016/02/16)
- 解析 | クラスタリング | 遺伝子間(応用) | MBCluster.Seq(Si_2014)+TCC正規化(Sun_2013) ①
- 解析 | シミュレーションカウントデータ | について (last modified 2015/11/10)
- 解析 | シミュレーションカウントデータ | Technical rep (ポアソン分布) (last modified 2015/01/2

解析 | クラスタリング | 遺伝子間(応用) | TCC正規化(Sun_2013) +MBCluster.Seq(Si_2014)

MBCluster.Seqパッケージを用いたやり方を示します。遺伝子間クラスタリングなのでデータの正規化が重要になってきます。ここでは、TCCを実行して得られたTCC正規化係数(iDEGES/edgeR正規化を実行して得られたサンプルごとに与える係数)を入力として与えるやり方を示します。正確には、DESeqパッケージと同じようにsize.factorsに変換してから与えています。理由は、TCCやedgeRパッケージ中で取り扱っている「正規化係数(normalization factors)」は、「総リード数(library sizes)」を補正するための「係数」という位置づけだからです(正規化係数を掛けたあとのライブラリサイズをeffective library sizesともいう)。その一方でMBCluster.Seq 中で述べている正規化は「総リード数 (= 平均発現レベルと同義)」や「発現量の中央値」で補正するためのものです。つまり、TCC正規化係数*総リード数 = effective library sizes、そしてeffective library sizesをさらに正規化したものがsize factorsという関係になります。 ②

「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13の10,0

Biological replicatesを模倣した

③

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータ。ヒトのメス3サンプル(HSF1-3)とオス3サンプル(HSM1-3), チンパンジーのメス3サンプル(PTF1-3)とオス3サンプル(PTM1-3), アカゲザルのメス3サンプル(RMF1-3)とオス3サンプル(RMM1-3)の 並びになっています。入力ファイル以外は、3.と基本的に同じ。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                   #出力ファイル名を指定してout_f1に格納(pngファイル
out_f2 <- "hoge4.txt"                   #出力ファイル名を指定してout_f2に格納(txtファイル
param_fig <- c(800, 500)                #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 5                    #クラスター数を指定
param_G1 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
```

TCC正規化を合わせる 参考

①TCCを実行して**正規化係数(normalization factors)**を得るところ。
②MBCluster.Seqは、RNASeq.Data関数部分で、任意の**サイズファクター(size factors)**をNormalizerオプション部分で与えられるので…

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータ。ヒトのメス3サンプル(F
オス3サンプル(HSM1-3), チンパンジーのメス3サンプル(PTF1-3)とオス3サンプル(PTM1-3), アカゲザル
ンプル(RMF1-3)とオス3サンプル(RMM1-3)の 並びになっています。入力ファイル以外は、3.と基本的に

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout_f1に格納(pngファイル
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout_f2に格納(txtファイル
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル)
param_clust_num <- 5                      #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定した
dim(data)                                 #オブジェクトdataの行数と列数を表示

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
norm.factors <- tcc$norm.factors          #TCC正規化係数をnorm.factorsに格納
ef.libsizes <- colSums(data)*norm.factors#総リード数(library sizes)にTCC正規化係数を掛
size.factors <- ef.libsizes/mean(ef.libsizes)#effective library sizesを正規化してサイ

#前処理(正規化)
hoge <- RNASeq.Data(data, Normalizer=size.factors,#クラスタリングに必要な基礎情報を取得
           Treatment=data.cl, GeneID=rownames(data))#クラスタリングに必要な基礎情報を取得

#本番(クラスタリング)
```

TCC正規化を合わせる

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

Blekhman et al., Genome Res., 2010の 20,689 genes×18 samplesのカウントデータ。ヒトの
オス3サンプル(HSM1-3), チンパンジーのメス3サンプル(PTF1-3)とオス3サンプル(PTM
ンプル(RMF1-3)とオス3サンプル(RMM1-3)の 並びになっています。入力ファイル以外

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_
out_f1 <- "hoge4.png"                     #出力ファイル名を指定してout
out_f2 <- "hoge4.txt"                     #出力ファイル名を指定してout
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅
param_clust_num <- 5                      #クラスター数を指定
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定した
dim(data)                                 #オブジェクトdataの行数と列数を表示

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2、G
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行した結
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtcc
norm.factors <- tcc$norm.factors          #TCC正規化係数をnorm.factorsに格納
ef.libsizes <- colSums(data)*norm.factors#総リード数(library sizes)にTCC正規化係数を掛
size.factors <- ef.libsizes/mean(ef.libsizes)#effective library sizesを正規化してサイ

#前処理(正規化)
hoge <- RNASeq.Data(data, Normalizer=size.factors,#クラスタリングに必要な基礎情報を取得
           Treatment=data.cl, GeneID=rownames(data))#クラスタリングに必要な基礎情報を取得

#本番(クラスタリング)
```

①この部分でnormalization factors(norm.factorsオブジェクト)からsize factors(size.factorsオブジェクト)への変換を行っている。
ここは計算結果云々よりも、edgeR(とTCC)系のnormalization factorsと、DESeq2系のsize factorsの用語と中身の違いや変換の仕方を「こんな感じか。2016.07.22の資料にあったような…」と記憶に留めておく程度でよい

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

TCC発現変動とも合体

実用上は、TCCによるANOVA的解析（どこかの群間で発現変動する遺伝子の検出）結果の分類目的でMBCluster.Seqを利用したいかも。②例題9

- 解析 | 発現変動 | 3群間 | 対応なし | について (last modified 2015/11/05)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq2(Love_2014) (last modified 2015/0
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun_2013) (last modified 2016/05/24)推奨 NE
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | EBSeq(Leng_2013) (last modified 2016/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | SAMseq(Li_2013) (last modified 2015/02/10)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq(Anders_2010) (last modified 2014/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | baySeq(Hardcastle_2010) (last modified 2016/03/07)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | edgeR(Robinson_2010) (last modified 2015/02/03)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) (last modified 2016/05/30)推奨 NE
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+baySeq(Hardcastle_2010) (l
- 解析 | 発現変動
- 解析 | 発現変動
- 解析 | 発現変動
- 解析 | 発現変動
- 解析 | 発現変動

①

解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) NEW

「応用」でやりたいことは、例題1-7「指定した群間で発現に差がある遺伝子の検出」、そして例題8以降が「基礎」の結果と遺伝子間クラスタリングを組み合わせたパターン分類です。例題1-7「指定した群間で発現に差がある遺伝子の検出」デザイン行列design中のparam_coefで指定した列を除くことでreduced modelを作成します。TCCを用いたやり方を示します。内部的にiDEGES/edgeR(Sun_2013)正規化を実行したのち、edgeRパッケージ中のGLM LRT法で発現変動遺伝子(Differentially expressed Genes; DEGs)検出を行っています。TCC原著論文中のiDEGES/edgeR-edgeRという解析パイプラインに相当します。TCC原著論文(Sun et al., BMC Bioinformatics, 2013)では3群間複製ありデータ用の推奨パイプラインを示していませんでしたが、多群間比較用の推奨ガイドライン提唱論文
edgeR"と同じものです。この2つ
「ファイル」-「ディレクトリの変更」で

②

1. サンプルデータ40の10,000 gene

シミュレーションデータ(G1群4サン
(gene_1～gene_1000がG1群で5倍
高発現) gene_3001～gene_10000
のを検出するやり方(帰無仮説: G
目から3000行目のところのほとん
方法が妥当であることが分かりま

```
in_f <- "data_hypodata_4v
```

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
in_f <- "sample_blekhman_18.txt"        #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge9.txt"                   #出力ファイル名を指定してout_f1に格納(txtファ
out_f2 <- "hoge9.png"                   #出力ファイル名を指定してout_f2に格納(pngファ
param_G1 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_fig <- c(800, 500)                #ファイル出力時の横幅と縦幅を指定(単位はピクセル
param_clust_num <- 10                   #クラスター数を指定

#必要なパッケージをロード
library(TCC)                            #パッケージの読み込み
library(MBCluster.Seq)                  #パッケージの読み込み
```

TCC発現変動とも合体

①例題9。約20分かかるので、まずはコード全体をコピペ

① 9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに格納
out_f1 <- "hoge9.txt"                     #出力ファイル名を指定してout_f1に格納(txtファ
out_f2 <- "hoge9.png"                     #出力ファイル名を指定してout_f2に格納(pngファ
param_G1 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_fig <- c(800, 500)                  #ファイル出力時の横幅と縦幅を指定(単位はピクセル
param_clust_num <- 10                     #クラスター数を指定

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2
tcc <- new("TCC", data, data.cl)          #TCCクラスオブジェクトtccを作成

#本番(TCC正規化)
```

TCC発現変動とも合体

①例題9。②TCCによるANOVA的解析（どこかの群間で発現変動する遺伝子の検出）部分。③MBCluster.Seq実行時に用いるsize factors情報取得部分。④の部分で使われている

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
#前処理(TCCクラスオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を
tcc <- new("TCC", data, data.cl)        #TCCクラスオブジェクトtccを作成

#本番(TCC正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を
normalized <- getNormalizedData(tcc)    #正規化後のデータを取り出してnormalizedに格納
norm.factors <- tcc$norm.factors        #TCC正規化係数をnorm.factorsに格納
ef.libsizes <- colSums(data)*norm.factors#総リード数(library sizes)にTCC正規化係数を
size.factors <- ef.libsizes/mean(ef.libsizes)#effective library sizesを正規化してサ

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をt
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#前処理(MBCluster.Seq基礎情報取得)
hoge <- RNASeq.Data(data, Normalizer=size.factors,#クラスタリングに必要な基礎情報を取
           Treatment=data.cl, GeneID=rownames(data))#クラスタリングに必要な基礎情報を

#本番(MBCluster.Seqクラスタリング)
set.seed(2016)                          #おまじない(同じ乱数になるようにするため)
```

TCC発現変動とも合

①例題9。②MBCluster.Seq実行部分。③set.seed関数を用いてタネ番号を指定(ここでは2016としてるが1でも1721などなんでもよい)しているので、全員同じ結果になる。逆に言えば、この部分をコメントアウトすれば毎回異なる結果になる

①

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力として

```
#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果をt
result <- getResult(tcc, sort=FALSE)    #p値などの結果をした結果をresultに格納

#前処理(MBCluster.Seq基礎情報取得)
hoge <- RNASeq.Data(data, Normalizer=size.factors,#クラスタリングに必要な基礎情報を取
          Treatment=data.cl, GeneID=rownames(data))#クラスタリングに必要な基礎情報を

#本番(MBCluster.Seqクラスタリング)
set.seed(2016)                          #おまじない(同じ乱数になるようにするため)
c0 <- KmeansPlus.RNASeq(data=hoge, nK=param_clust_num,#K-means clusteringのクラスタ
          model="nbinom", print.steps=F)#K-means clusteringのクラスター中心の初期値
cls <- Cluster.RNASeq(data=hoge, model="nbinom",#クラスタリング
          centers=c0$centers, method="EM")#クラスタリング
tr <- Hybrid.Tree(data=hoge, model="nbinom",cluster=cls$cluster)#hybrid-hierarchi

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(tcc$count), normalized, result, cls$cluster)#保存したい情報を
tmp <- tmp[order(tmp$rank),]            #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し

#様々なFDR閾値を満たす遺伝子数を表示
```

③

②

TCC発現変動とも合体

①例題9。②TCCによるANOVA的解析（どこかの群間で発現変動する遺伝子の検出）結果表示部分。③以前（スライド11）と同じ結果が得られていることがわかる

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value           #q-valueをq.valueに格納
sum(q.value < 0.05)                   #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数
sum(q.value < 0.10)                   #FDR
sum(q.value < 0.20)                   #FDR
sum(q.value < 0.30)                   #FDR

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13, width=param_fig[1]
plotHybrid.Tree(merge=tr, cluster=cls$cluste
dev.off()                             #おま

#後処理(各クラスターに属する遺伝子数を表示)
table(cls$cluster)                    #各ク
table(cls$cluster[q.value < 0.05])    #各ク
table(cls$cluster[q.value < 0.10])    #各ク
table(cls$cluster[q.value < 0.20])    #各ク
table(cls$cluster[q.value < 0.30])    #各ク
```

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, result,$
> tmp <- tmp[order(tmp$rank),]              #発現変動順に$
> write.table(tmp, out_f1, sep="\t", append=F, quote=F,$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value               #q-valueをq.va$
> sum(q.value < 0.05)                       #FDR = 0.05 (q$
[1] 7236
> sum(q.value < 0.10)                       #FDR = 0.10 (q$
[1] 8485
> sum(q.value < 0.20)                       #FDR = 0.20 (q$
[1] 10068
> sum(q.value < 0.30)                       #FDR = 0.30 (q$
[1] 11467
>
> #ファイルに保存(pngファイル)
> png(out_f2, pointsize=13, width=param_fig[1], height=$
```

TCC発現変動とも合体

①例題9。②table関数を用いて各クラスターに属するメンバー数（遺伝子数）を表示。③クラスター数が10個になっているのは、最初にparam_clust_numのところで10を指定していたから。この数は、（似ているからマージされて）最初に指定した数より少なくなることがある

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果を
write.table(tmp, out_f1, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value           #q-va
sum(q.value < 0.05)                   #FDR
sum(q.value < 0.10)                   #FDR
sum(q.value < 0.20)                   #FDR
sum(q.value < 0.30)                   #FDR

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13, width=param_fig[1]
plotHybrid.Tree(merge=tr, cluster=cls$cluste
dev.off()                             #おま

#後処理(各クラスターに属する遺伝子数を表
table(cls$cluster)                    #各ク
table(cls$cluster[q.value < 0.05])    #各ク
table(cls$cluster[q.value < 0.10])    #各ク
table(cls$cluster[q.value < 0.20])    #各ク
table(cls$cluster[q.value < 0.30])    #各ク
```

R Console

```
> #後処理(各クラスターに属する遺伝子数を表示)
> table(cls$cluster)                          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 875  728 1679  659 1025 2266 7711 3265 2216  265
> table(cls$cluster[q.value < 0.05])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 711  616 1198  560  885  894  556 1022  543  251
> table(cls$cluster[q.value < 0.10])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 780  664 1305  625  941 1119  755 1292  743  261
> table(cls$cluster[q.value < 0.20])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 850  726 1460  657 1003 1402 1051 1656  998  265
> table(cls$cluster[q.value < 0.30])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 874  727 1562  658 1023 1711 1377 2062 1208  265
>
```

TCC発現変動とも合体

①のクラスター番号が、②遺伝子間クラスタリング結果(hoge9.png)の赤枠の番号と対応する。②はクラスター番号順ではない点に注意

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

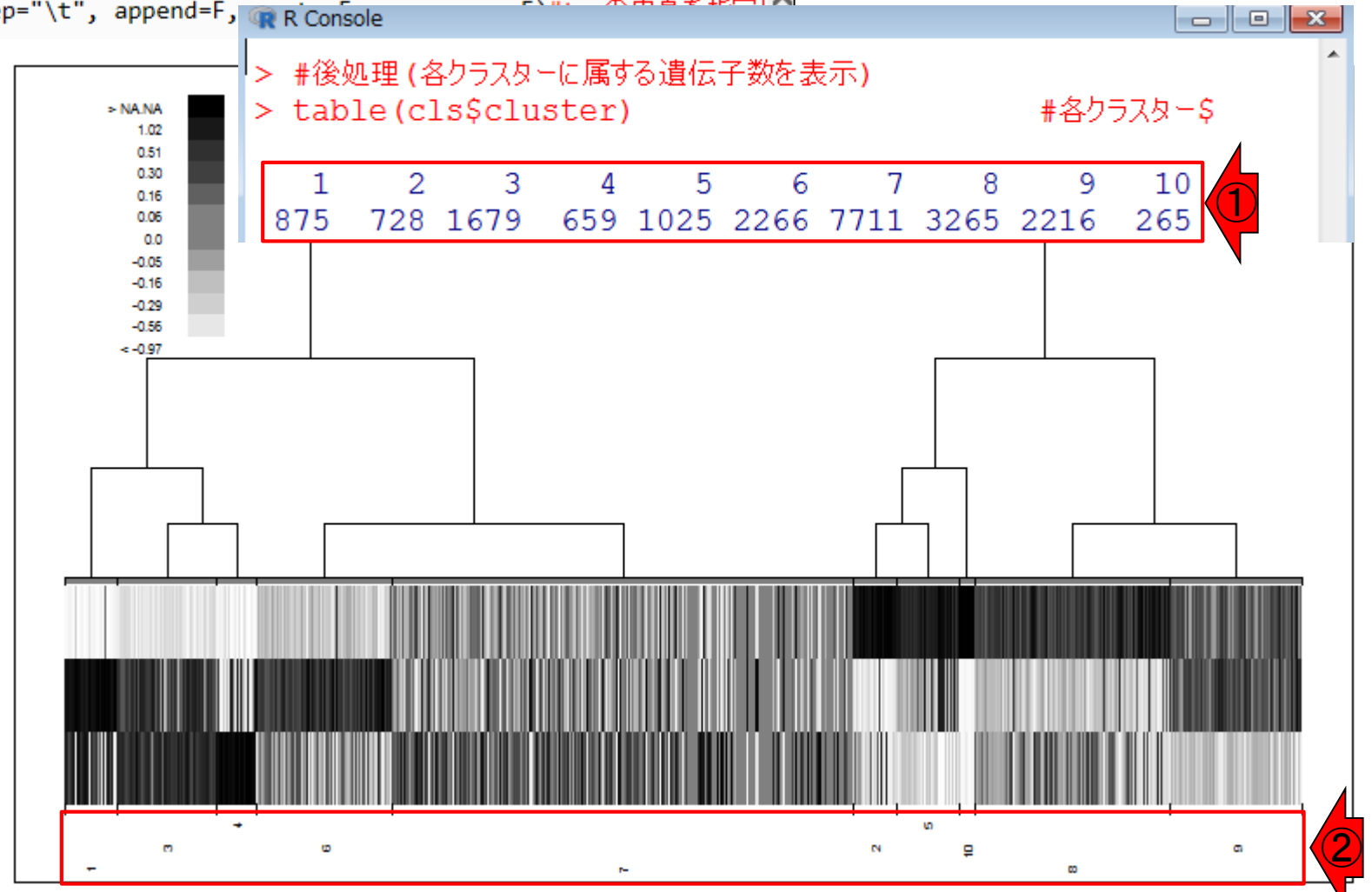
8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子
q.value <- tcc$stat$q.valu
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13,
plotHybrid.Tree(merge=tr,
dev.off()

#後処理(各クラスターに属する
table(cls$cluster)
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
```

R Console

```
> #後処理(各クラスターに属する遺伝子数を表示)
> table(cls$cluster)                    #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 875  728 1679  659 1025 2266 7711 3265 2216  265
```

①



②

TCC発現変動とも合体

①例えばクラスター7の遺伝子数が一番多く、②の発現プロファイルの面積（赤枠の横幅）とも対応する。これはどの群間でも似た色になっているので、non-DEGのクラスターなのだろうと妄想する

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

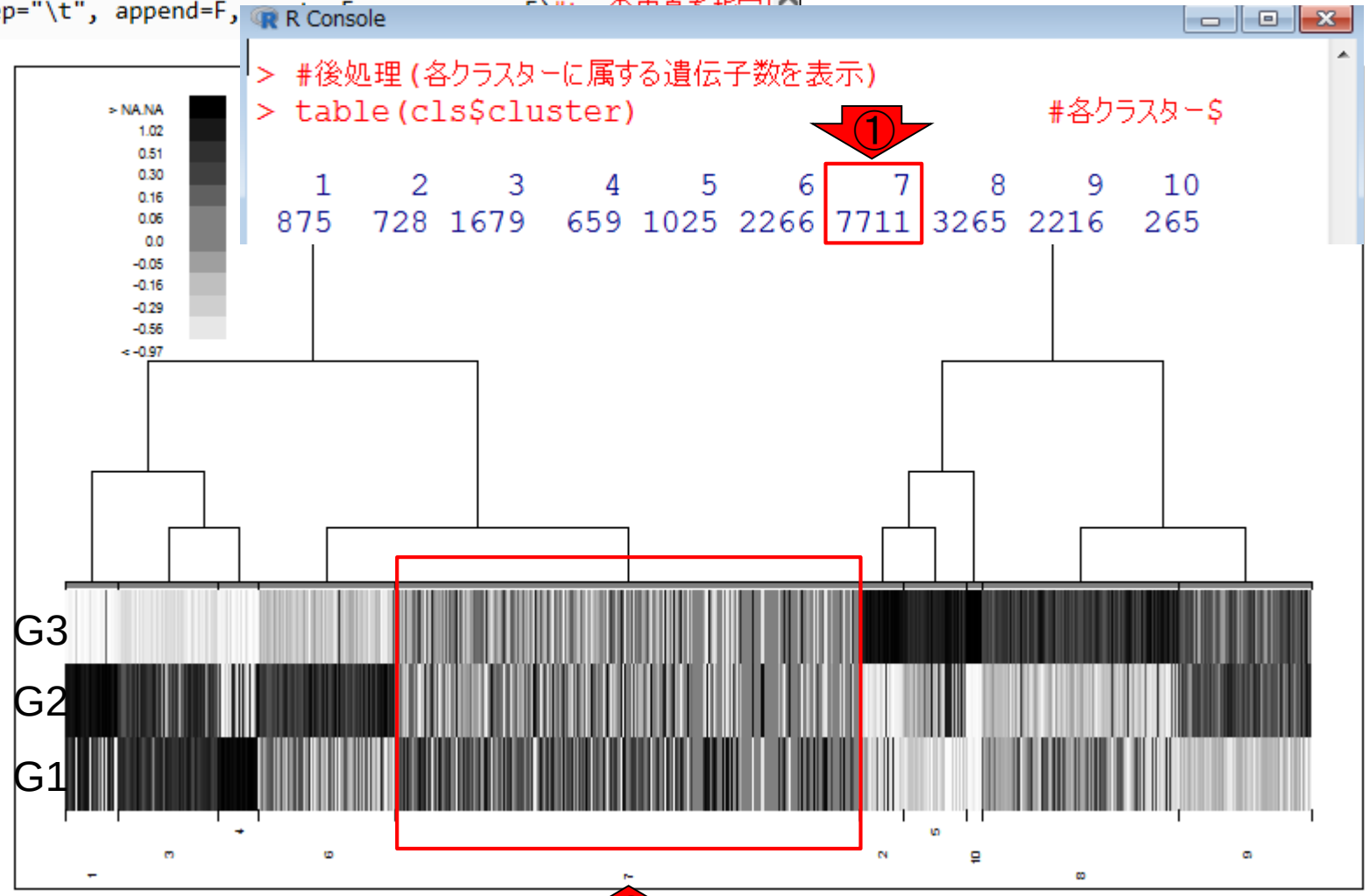
8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子
q.value <- tcc$stat$q.valu
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13,
plotHybrid.Tree(merge=tr,
dev.off()

#後処理(各クラスターに属する
table(cls$cluster)
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
```

R Console

```
> #後処理(各クラスターに属する遺伝子数を表示)
> table(cls$cluster)                    #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 875  728 1679  659 1025 2266 7711 3265 2216  265
```

①



②

TCC発現変動とも合

①TCC発現変動解析と遺伝子間クラスタリングを合体させる一番のメリットの部分。赤下線部分で示すように、FDR閾値を満たすものに限定して、各クラスターに属する遺伝子数を概観可能

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value           #q-va
sum(q.value < 0.05)                   #FDR
sum(q.value < 0.10)                   #FDR
sum(q.value < 0.20)                   #FDR
sum(q.value < 0.30)                   #FDR

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13, width=param_fig[1]
plotHybrid.Tree(merge=tr, cluster=cls$cluste
dev.off()                             #おま

#後処理(各クラスターに属する遺伝子数を表示)
table(cls$cluster)                    #各ク
table(cls$cluster[q.value < 0.05])    #各ク
table(cls$cluster[q.value < 0.10])    #各ク
table(cls$cluster[q.value < 0.20])    #各ク
table(cls$cluster[q.value < 0.30])    #各ク
```

①

R Console

```
> #後処理(各クラスターに属する遺伝子数を表示)
> table(cls$cluster)                          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 875  728 1679  659 1025 2266 7711 3265 2216  265
> table(cls$cluster[q.value < 0.05])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 711  616 1198  560  885  894  556 1022  543  251
> table(cls$cluster[q.value < 0.10])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 780  664 1305  625  941 1119  755 1292  743  261
> table(cls$cluster[q.value < 0.20])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 850  726 1460  657 1003 1402 1051 1656  998  265
> table(cls$cluster[q.value < 0.30])          #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 874  727 1562  658 1023 1711 1377 2062 1208  265
>
```

①

TCC発現変動とも合

①（おそらくnon-DEGの）クラスター7の遺伝子数（7,711個；7,711/20,689 = 37.27%）が、②556個（556/7,236 = 7.68%）に激減していることがわかる。②はFDR 5% (q-value < 0.05)を満たす遺伝子であること、他のクラスターでこの閾値を満たす遺伝子数の減少度合いの相対的な低さからも、クラスター7がnon-DEGクラスターであることの確証を得たことと同じ

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力

```
tmp <- tmp[order(tmp$rank),]                    #発現変動順にソート
write.table(tmp, out_f1, sep="\t", append=F,

#様々なFDR閾値を満たす遺伝子数を表示
q.value <- tcc$stat$q.value                     #q-va
sum(q.value < 0.05)                             #FDR
sum(q.value < 0.10)                             #FDR
sum(q.value < 0.20)                             #FDR
sum(q.value < 0.30)                             #FDR

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13, width=param_fig[1]
plotHybrid.Tree(merge=tr, cluster=cls$cluste
dev.off()                                       #おま

#後処理(各クラスターに属する遺伝子数を表示)
table(cls$cluster)                              #各ク
table(cls$cluster[q.value < 0.05])              #各ク
table(cls$cluster[q.value < 0.10])              #各ク
table(cls$cluster[q.value < 0.20])              #各ク
table(cls$cluster[q.value < 0.30])              #各ク
```

R Console

```
> #後処理(
> table(cls$cluster)                            #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 875  728 1679  659 1025 2266 7711 3265 2216  265
> table(cls$cluster[q.value <                   #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 711  616 1198  560  885  894  556 1022  543  251
> table(cls$cluster[q.value < 0.10])            #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 780  664 1305  625  941 1119  755 1292  743  261
> table(cls$cluster[q.value < 0.20])            #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 850  726 1460  657 1003 1402 1051 1656  998  265
> table(cls$cluster[q.value < 0.30])            #各クラスター$

   1    2    3    4    5    6    7    8    9   10
 874  727 1562  658 1023 1711 1377 2062 1208  265
>
```

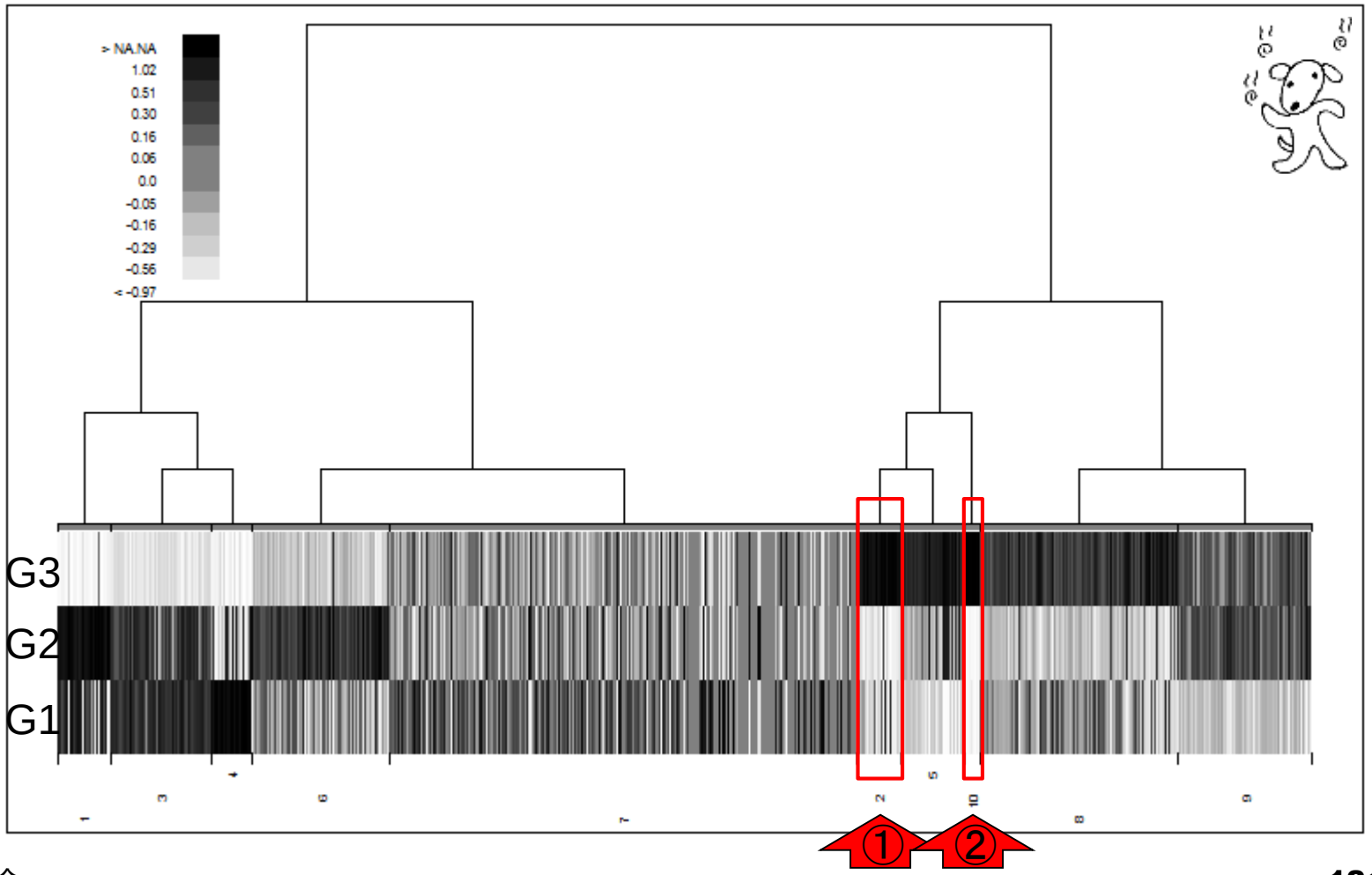
でもキモい…よね?!

門田は遺伝子間クラスタリングはやりません（正確に言うと嫌いです）。理由は、①クラスター2と②クラスター10は、パッと見、同じ「G3群で高発現パターン」なのに、分かれてしまっているからです

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し

#様々なFDR閾値を満たす遺伝子
q.value <- tcc$stat$q.valu
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13,
plotHybrid.Tree(merge=tr,
dev.off()

#後処理(各クラスターに属する
table(cls$cluster)
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
```



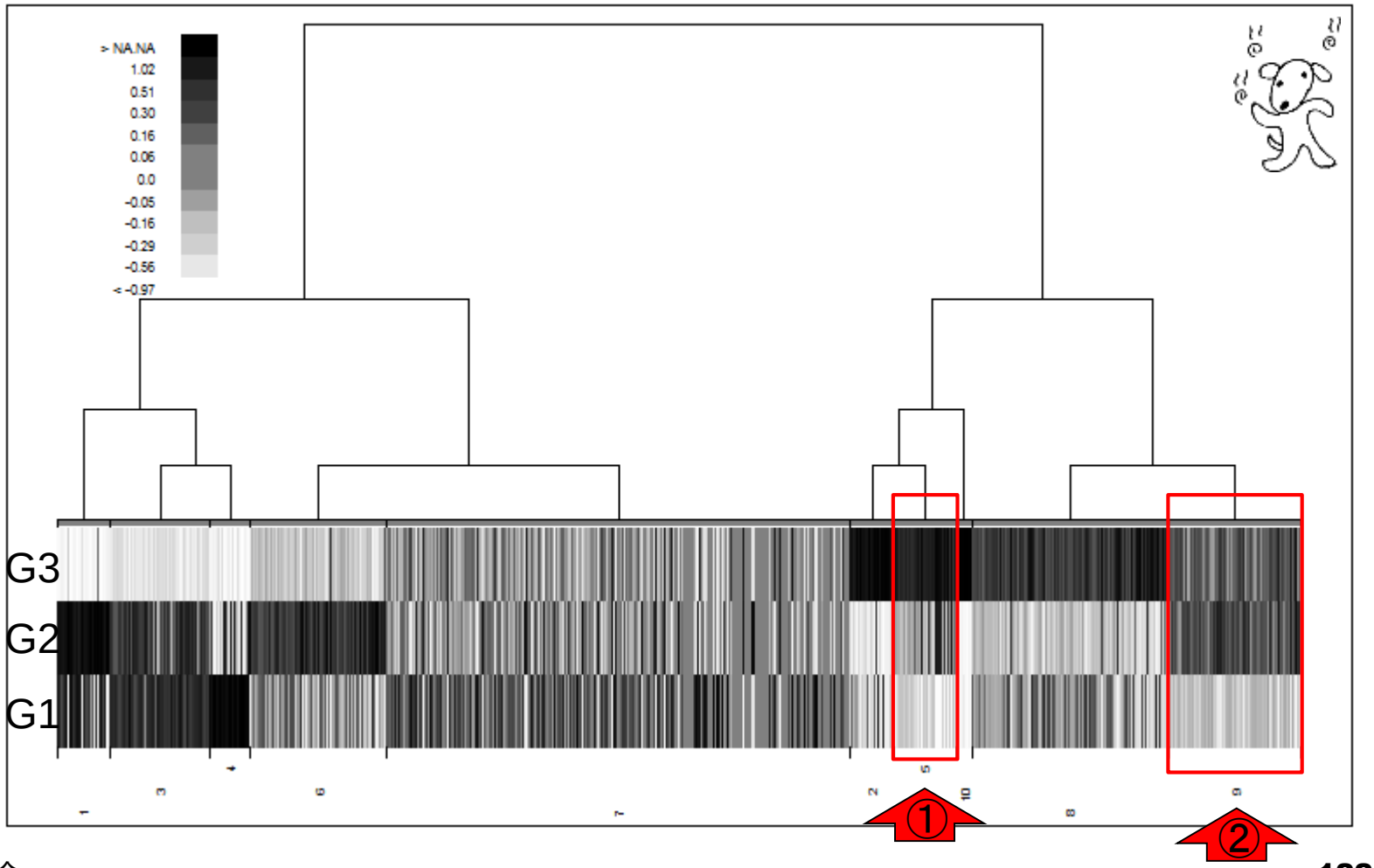
でもキモい…よね?!

他にも、①クラスター5と②クラスター9が同じ発現パターンにならないのもキモい。「パターン1, 2の遺伝子数がどうのこうの」ではなく、「〇〇群で高発現パターンがどうのこうの」という議論がシタい!

9. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

8.と基本的に同じで、MBCluster.Seq実行時にTCC正規化後のデータを入力としています。

```
tmp <- tmp[order(tmp$rank),]          #発現変動順にソートした結果をtmpに格納
write.table(tmp, out_f1, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し

#様々なFDR閾値を満たす遺伝子
q.value <- tcc$stat$q.valu
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)

#ファイルに保存(pngファイル)
png(out_f2, pointsize=13,
plotHybrid.Tree(merge=tr,
dev.off()

#後処理(各クラスターに属する
table(cls$cluster)
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
table(cls$cluster[q.value
```



Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

baySeq

①baySeqの②例題4。発現パターン分類ができます!③ここの数は増やすほど結果の信頼度が高くなりますが、その分計算時間がかかります。④性能評価論文では、ここを5000でやりました

- 解析 | 発現変動 | 3群間 | 対応なし | について (last modified 2015/11/05)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq2(Love_2014) (last modif
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun_2013) (last modified 2016/05/24)推奨 NE
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | EBSeq(Leng_2013) (last modified 2016/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | SAMseq(Li_2013) (last modified 2015/02/10)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | DESeq(Anders_2010) (last modified 2014/03/13)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | baySeq(Hardcastle_2010) (last modified 2016/03/07)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | edgeR(Robinson_2010) (last modified 2015/02/03)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) (last modified 2016/05/30)推奨 NE
- 解析 | 発現変
- 解析 | 発現変
- 解析 | 発現変
- 解析 | 発現変
- 解析 | 発現変
- 解析 | 発現変
- 解析 | 発現変

①

解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | baySeq(Hardcastle_2010) NEW

baySeqを用いて発現変動遺伝子(Differentially expressed Genes; DEGs) 検出を行うやり方を示します。ANOVAのような「どこかの群間で発現に差がある遺伝子を検出」ではなく、baySeqで定義したいくつかの発現パターンのどれに相当するかまで示してくれるので、「どこで発現に差があ

「ファイル」-「ディレクトリの変更」で

②

1. サンプルデータ15の10,000 genes×9

シミュレーションデータ(G1群3サンプ
(gene_1～gene_2100がG1群で3倍高
6倍高発現) gene_3001～gene_10000

```
in_f <- "data_hypodata_3vs3
out_f <- "hoge1.txt"
param_G1 <- 3
param_G2 <- 3
param_G3 <- 3
param_FDR <- 0.05
param_samplesize <- 5000
```

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
in_f <- "sample_blekhman_18.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 <- 6                             #G1群のサンプル数を指定
param_G2 <- 6                             #G2群のサンプル数を指定
param_G3 <- 6                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を
param_samplesize <- 500                   #ブートストラップリサンプリング回数(100000が推

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2

#前処理(countDataオブジェクトの作成)
                                          m_G3))#non-DEGのパターンを作
```

③

④

Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

①例題4。②baySeqもFDR閾値を満たす遺伝子を得ることができる。まずはコピペ。約4分

baySeq

①

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

②

```
in_f <- "sample_blekhman_18.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 <- 6                           #G1群のサンプル数を指定
param_G2 <- 6                           #G2群のサンプル数を指定
param_G3 <- 6                           #G3群のサンプル数を指定
param_FDR <- 0.05                       #DEG検出時のfalse discovery rate (FDR)閾値を
param_samplesize <- 500                 #ブートストラップリサンプリング回数(100000が推

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

#入力ファイルの読み込みとラベル情報の作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep(3, param_G3))#G1群を1、G2群を2

#前処理(countDataオブジェクトの作成)
NDE <- c(rep(1, param_G1), rep(1, param_G2), rep(1, param_G3))#non-DEGのパターンを作
DE <- c(rep("G1", param_G1), rep("G2", param_G2), rep("G3", param_G3))#DEGのパター
ba <- new("countData", data=as.matrix(data), replicates=data.cl)#countDataオブジェ
groups(ba) <- list(NDE=NDE, DE=DE)      #non-DEGとDEGのパターン情報を格納

#本番(正規化)
```

baySeq

①

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時
います。約4分。

```
#前処理(countDataオブジェクトの作成)
NDE <- c(rep(1, param_G1), rep(1, param_G2), rep(1, param_G3))#no
DE <- c(rep("G1", param_G1), rep("G2", param_G2), rep("G3", param
ba <- new("countData", data=as.matrix(data), replicates=data.cl)#countDataオブジェ
groups(ba) <- list(NDE=NDE, DE=DE)      #non-DEGとDEGのパターン情報を格納

#本番(正規化)
libsizes(ba) <- getLibsizes(ba, estimationType="edgeR")#edgeRパッケージ中の正規化法

#本番(DEG検出)
set.seed(2015)                          #おまじない(同じ乱数になるようにするため)
ba <- getPriors.NB(ba, samplesize=param_samplesize, estimation="QL", cl=NULL)#事前
ba <- getLikelihoods(ba, pET="BIC", nullData=FALSE, cl=NULL)#事後確率を計算
res <- topCounts(ba, group="D", normaliseData=T, number=nrow(data))#(発現変動順に
res <- res[rownames(data),]             #解析結果をオリジナルの並びに再ソート
q.value <- res$FDR.DE                   #FDR.DEの情報ををq.valueに格納
ranking <- rank(q.value)                #q.valueでランキングした結果をrankingに格納

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報を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
```

②

③

④

②このあたりがbaySeq実行部分。デフォルトは発現変動順だが、③ここではオリジナルの入力データの並びに変更している。複数のパッケージの結果をマージしたい場合には、オリジナルの並びにしておいたほうが便利なので、そうしたいとき用のやり方として覚えておくとよい。④set.seedを埋め込んでいるので、皆同じ結果

baySeq

①baySeqの結果。②TCCの結果(スライド11)と比べて、全体的に少なめの遺伝子数となっていることがわかる

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)
sum(q.value < 0.10)
sum(q.value < 0.20)
sum(q.value < 0.30)

#結果のまとめ(全遺伝子)
head(res$ordering)
table(res$ordering)
table(res$ordering)/length(res$ordering)

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)
sum(obj)
table(res$ordering[obj])
table(res$ordering[obj])/length(res$ordering[obj]
```

①

```
#様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
> sum(q.value < 0.20)
[1] 10068
> sum(q.value < 0.30)
[1] 11467
```

②

R Console

```
> q.value <- res$F
> ranking <- rank(
>
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(res), res, ranking)#保存した$
> write.table(tmp, out_f, sep="\t", append=F, quote=$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> sum(q.value < 0.05)                    #FDR = 0.05$
[1] 4065
> sum(q.value < 0.10)                    #FDR = 0.10$
[1] 4795
> sum(q.value < 0.20)                    #FDR = 0.20$
[1] 5934
> sum(q.value < 0.30)                    #FDR = 0.30$
[1] 7070
>
```

①

baySeq

①baySeqの一番の長所は、①遺伝子ごとに予め定義した(predefined)発現パターンの中から、一番近いパターンを割り当ててくれること。③baySeqで3群間比較を行うときは、計10パターンを定義している

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR =

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最
table(res$ordering)                    #パ
table(res$ordering)/length(res$ordering)#パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターンご
table(res$ordering[obj])/length(res$ordering[obj]
```

①

②

R Console

```
> #結果のまとめ(全遺伝子)
> head(res$ordering)                                    #最初の6遺$
[1] G3>G2>G1 G2>G3>G1 G2>G3>G1 G2>G3>G1 G1>G2>G3
[6] G2>G1>G3
10 Levels: G1=G2=G3 G1>G2=G3 G1>G2>G3 ... G3>G2>G1
> table(res$ordering)                                   #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
    2803      460     3121     1803      486
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
    3288     1951      605     3120     3052
> table(res$ordering)/length(res$ordering)#パターン$

  G1=G2=G3    G1>G2=G3    G1>G2>G3    G1>G3>G2
0.13548262 0.02223404 0.15085311 0.08714776
  G2>G1=G3    G2>G1>G3    G2>G3>G1    G3>G1=G2
0.02349074 0.15892503 0.09430132 0.02924259
  G3>G1>G2    G3>G2>G1
0.15080478 0.14751800
>
```

baySeq

赤下線のres$orderingが遺伝子ごとに割り振られた発現パターン情報ベクトル。このベクトル中には、遺伝子数分の要素数がある。①head関数で最初の6遺伝子分のパターンを表示

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR =
sum(q.value < 0.10)                    #FDR =
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子
head(res$ordering)                     #最初の6遺
table(res$ordering)                    #パターンご
table(res$ordering)/length(res$ordering)#パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターンご
table(res$ordering[obj])/length(res$ordering[obj]
```

①

R Console

```
> #結果のまとめ(全遺伝子)
> head(res$ordering)                                    #最初の6遺$
[1] G3>G2>G1 G2>G3>G1 G2>G3>G1 G2>G3>G1 G1>G2>G3
[6] G2>G1>G3
10 Levels: G1=G2=G3 G1>G2=G3 G1>G2>G3 ... G3>G2>G1
> table(res$ordering)                                   #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
    2803      460     3121     1803      486
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
    3288     1951      605     3120     3052
> table(res$ordering)/length(res$ordering)#パターン$

  G1=G2=G3   G1>G2=G3   G1>G2>G3   G1>G3>G2
0.13548262 0.02223404 0.15085311 0.08714776
  G2>G1=G3   G2>G1>G3   G2>G3>G1   G3>G1=G2
0.02349074 0.15892503 0.09430132 0.02924259
  G3>G1>G2   G3>G2>G1
0.15080478 0.14751800
>
```

baySeq

①table関数で、どんなパターンがあるかや、パターンごとの遺伝子数を概観。全部で20,689遺伝子のうち、②たった2,803個しかG1=G2=G3のパターン（つまり完全なnon-DEG）に当てはまらないという結果には違和感を覚える。が、それ以外のパターンに当てはまっているとしてもDEGとは限らないと考えれば…一応は納得できる

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行
います。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最初の
table(res$ordering)                    #パターン
table(res$ordering)/length(res$ordering)#パター

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターン
table(res$ordering[obj])/length(res$ordering[obj]
```

①

②

R Console

```
> #結果のまとめ(全遺伝子)
> head(res$ordering)                                    #最初の6遺$
[1] G3>G2>G1 G2>G3>G1 G2>G3>G1 G2>G3>G1 G1>G2>G3
[6] G2>G1>G3
10 Levels: G1=G2=G3 G1>G2=G3 G1>G2>G3 ... G3>G2>G1
> table(res$ordering)                                   #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
    2803      460     3121     1803      486
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
    3288     1951      605     3120     3052
> table(res$ordering)/length(res$ordering)#パターン$

  G1=G2=G3   G1>G2=G3   G1>G2>G3   G1>G3>G2
0.13548262 0.02223404 0.15085311 0.08714776
  G2>G1=G3   G2>G1>G3   G2>G3>G1   G3>G1=G2
0.02349074 0.15892503 0.09430132 0.02924259
  G3>G1>G2   G3>G2>G1
0.15080478 0.14751800
>
```

baySeq

①の赤下線部分は遺伝子数情報に相当。全遺伝子数で割ることで、各パターンの占める割合を表示しているだけです。②例えばG1=G2=G3パターンは、2,803/20,689 = 0.13548262となる。③この発現パターン分類をそのまま使っているのが…

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実
います。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最初の6遺
table(res$ordering)                    #パターンご
table(res$ordering)/length(res$ordering)  #パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パター
table(res$ordering[obj])/length(res$ordering[o
```

①

②

R Console

```
> #結果のまとめ(全遺伝子)
> head(res$ordering)                                    #最初の6遺$
[1] G3>G2>G1 G2>G3>G1 G2>G3>G1 G2>G3>G1 G1>G2>G3
[6] G2>G1>G3
10 Levels: G1=G2=G3 G1>G2=G3 G1>G2>G3 ... G3>G2>G1
> table(res$ordering)                                   #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
    2803      460     3121     1803      486
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
    3288     1951      605     3120     3052
> table(res$ordering)/length(res$ordering)#パターン$

  G1=G2=G3   G1>G2=G3   G1>G2>G3   G1>G3>G2
0.13548262 0.02223404 0.15085311 0.08714776
  G2>G1=G3   G2>G1>G3   G2>G3>G1   G3>G1=G2
0.02349074 0.15892503 0.09430132 0.02924259
  G3>G1>G2   G3>G2>G1
0.15080478 0.14751800
>
```

③

性能評価論文のTable 4

①Tang et al., 2015の、②Table 4の、③all_genesのところ。ほぼ同じ数値であることがわかる

②

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

③

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

①

Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

性能評価論文のTable 4

①の一番右側の数値(3,975)は、baySeqでFDR = 0.05を満たす遺伝子数。さきほど行ったリサンプリング500回の結果(②4,065個;スライド127)との違いは、バージョンの違いとリサンプリング回数(①の結果は5000回)の違いに起因します

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G1	G2	G2	G1	
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0					19.7	19.9	7585
voom	0.0	0.7	21.3	7.7					18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7					18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9					20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5					19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0					20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns ... dicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3.

```
> #様々なFDR閾値を満たす遺伝子数を表示
> sum(q.value < 0.05)
[1] 4065
> sum(q.value < 0.10)
[1] 4795
> sum(q.value < 0.20)
[1] 5934
> sum(q.value < 0.30)
[1] 7070
```

②

①

性能評価論文のTable 4

①と同じようなFDR = 0.05を満たすものに限定した結果を得るところが…

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

①

Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

baySeq

①のコード。②の割合の数値分布はTable 4のものと非常に似ていることがわかる

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最初の6遺
table(res$ordering)                    #パターンご
table(res$ordering)/length(res$ordering)#パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターンご
table(res$ordering[obj])/length(res$ordering[obj]
```

①

R Console

```
> #結果のまとめ(FDR閾値を満たすものを概観)
> obj <- (q.value < param_FDR)                    #条件を満た$
> sum(obj)                                        #条件を満た$
[1] 4065
> table(res$ordering[obj])                        #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
       0       30      851       237        53
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
     954      261      112       780       787
> table(res$ordering[obj])/length(res$ordering[obj])$

   G1=G2=G3    G1>G2=G3    G1>G2>G3    G1>G3>G2
0.000000000 0.007380074 0.209348093 0.058302583
   G2>G1=G3    G2>G1>G3    G2>G3>G1    G3>G1=G2
0.013038130 0.234686347 0.064206642 0.027552276
   G3>G1>G2    G3>G2>G1
0.191881919 0.193603936
>
```

②

baySeq

例えば、①や②のパターンの割合が多いが、Table 4で対応する箇所は…

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに格納
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最初の6遺
table(res$ordering)                    #パターンご
table(res$ordering)/length(res$ordering)#パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターンご
table(res$ordering[obj])/length(res$ordering[obj]
```

R Console

```
> #結果のまとめ(FDR閾値を満たすものを概観)
> obj <- (q.value < param_FDR)                    #条件を満た$
> sum(obj)                                        #条件を満た$
[1] 4065
> table(res$ordering[obj])                        #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
       0       30      851       237        53
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
     954      261      112       780       787
> table(res$ordering[obj])/length(res$ordering[obj])$

   G1=G2=G3    G1>G2=G3    G1>G2>G3    G1>G3>G2
0.000000000 0.007380074 0.209348093 0.058302583
   G2>G1=G3    G2>G1>G3    G2>G3>G1    G3>G1=G2
0.013038130 0.234686347 0.064206642 0.027552276
   G3>G1>G2    G3>G2>G1
0.191881919 0.193603936
>
```

①

②

①や②のあたり

性能評価論文のTable 4

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

② ①

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

性能評価論文のTable 4

ちなみに①の方法名が*EEE-E*となっているのが、②TCCの結果(スライド11)に対応する。③FDR = 0.05を満たす遺伝子数が若干異なるのはバージョンの違いに起因

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9				
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2				
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2				
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4				
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3				
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1				

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" col
example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

①

```
> #様々なFDR閾値を満たす遺伝子数を表示
> q.value <- tcc$stat$q.value
> sum(q.value < 0.05)
[1] 7236
> sum(q.value < 0.10)
[1] 8485
> sum(q.value < 0.20)
[1] 10068
> sum(q.value < 0.30)
[1] 11467
```

③

②

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

ポイントは、結果の並びを入力データと同じにしておくこと

何故TCCでパターン分類できるのか？

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

baySeq

①res$orderingは、②行列resの、③ordering列の部分を取り出したものに相当する。④そしてその並びは、入力データと同じ

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実行時間を大幅に短縮しています。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存した
write.table(tmp, out_f, sep="\t", append=F, quote

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.0
sum(q.value < 0.10)                    #FDR = 0.1
sum(q.value < 0.20)                    #FDR = 0.2
sum(q.value < 0.30)                    #FDR = 0.3

#結果のまとめ(全遺伝子
head(res$ordering)                     #最初の6遺
table(res$ordering)                    #パターンご
table(res$ordering)/length(res$ordering)#パターン

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (q.value < param_FDR)           #条件を満た
sum(obj)                               #条件を満た
table(res$ordering[obj])               #パターンご
table(res$ordering[obj])/length(res$ordering[obj]
```

①

R Console

②

```
> head(res, n=3)
                  rowID HSF1 HSF2 HSF3 HSM1 HSM2 HSM3
ENSG00000000003     1  363  348  183  151  396  351
ENSG00000000005     2    0    0    0    0    1    0
ENSG00000000419     3   89   71   61   49   73   61
                  PTF1 PTF2 PTF3 PTM1 PTM2 PTM3 RMF1
ENSG00000000003  468  439  397  415  674  483  453
ENSG00000000005    1    4    1    0    1    1    0
ENSG00000000419   81   67   67   60   57  105   59
                  RMF2 RMF3 RMM1 RMM2 RMM3
ENSG00000000003  521  329  354 1726  626
ENSG00000000005    1    1    2    0    0
ENSG00000000419   81   39   41  105   80
                   Likelihood ordering     FDR.DE
ENSG00000000003 0.1841684742 G3>G2>G1 0.2445615
ENSG00000000005 0.0534156268 G2>G3>G1 0.4570674
ENSG00000000419 0.0009908059 G2>G3>G1 0.7251691
                  FWER.DE
ENSG00000000003       1
ENSG00000000005       1
ENSG00000000419       1
>
```

④

③

baySeq

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

3.と基本的に同じでparam_samplesizeで指定するリサンプリング回数を1/10にして実
います。約4分。

```
#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(res), res, ranking)#保存したい情報をtmpに
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定し

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                    #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数
sum(q.value < 0.10)                    #FDR = 0.10 (q-value < 0.10)を満たす遺伝子数
sum(q.value < 0.20)                    #FDR = 0.20 (q-value < 0.20)を満たす遺伝子数
sum(q.value < 0.30)                    #FDR = 0.30 (q-value < 0.30)を満たす遺伝子数

#結果のまとめ(全遺伝子)
head(res$ordering)                     #最初の6遺伝子分のパターンを表示
table(res$ordering)                    #パターンごとの出現頻度を表示
table(res$ordering)/length(res$ordering)#パターンごとの出現確率を表示

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (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
sum(obj)                               #条件を満たす数を表示
table(res$ordering[obj])               #パターンごとの出現頻度を表示(条件を満たすもの
table(res$ordering[obj])/length(res$ordering[obj])#パターンごとの出現確率を表示(条件
```

①がFDR閾値を満たす遺伝子にTRUE、それ以外にFALSEを割り当てた論理値ベクトルobj作成部分。②res$ordering[obj]とすることで条件を満たすものだけでtable関数などを実行している。それゆえ、res$orderingを保持しつつ、条件判定される大元をTCCのq-value情報ベクトルに置換してやればいいのです。つまり③ここをTCC結果に置換

TCCのq-valueを…

①をコピペすればほぼ同じ結果を得ることができる。②がTCCによる発現変動解析部分。③得られたTCCのq-value情報(tcc$stat$q.value)をここで利用するのが基本形

- TCCのq-valueを…

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun 2013)」の例題7をベース
小限のコードを実行。baySeq実行結果のres$ordering情報を利用した発現パターン分類を行う。

①

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

#前処理(TCCクラスオブジェクトの作成)
tcc <- new("TCC", data, data.cl)            #TCCクラスオブジェクトtccを作成

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", test.method="edger",#正規化を実行し
                       iteration=3, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果を

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=param_FDR)#DEG検出を実行した結果を

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (tcc$stat$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
sum(obj)                                    #条件を満たす数を表示
table(res$ordering[obj])                    #パターンごとの出現頻度を表示(条件を満たすもの
table(res$ordering[obj])/length(res$ordering[obj])#パターンごとの出現確率を表示(条件
```

②

③

TCCのq-valueを…

コピペ実行結果。②FDR = 0.05を満たす遺伝子数が同じ7,236個(スライド11)だったので安心。③の分布はTable 4の*EEE-E*とほとんど同じ結果になってますね

- TCCのq-valueを…

「解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | TCC(Sun 2013)」の例題7をベースにして、必要最小限のコードを実行。baySeq実行結果のres$ordering情報を利用した発現パターン分類を行う。

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

#前処理(TCCクラスオブジェクトの作成)
tcc <- new("TCC", data, data.cl)            #TCCクラス

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="tmm", te
                       iteration=3, FDR=0.1, floo

#本番(DEG検出)
tcc <- estimateDE(tcc, test.method="edger", FDR=p

#結果のまとめ(FDR閾値を満たすものを概観)
obj <- (tcc$stat$q.value < param_FDR)#条件を満たす
sum(obj)                                    #条件を満た
table(res$ordering[obj])                    #パターンご
table(res$ordering[obj])/length(res$ordering[obj
```

①

R Console

```
> #結果のまとめ(FDR閾値を満たすものを概観)
> obj <- (tcc$stat$q.value < param_FDR)#条件を満たす$
> sum(obj)                                    #条件を満た$
[1] 7236
> table(res$ordering[obj])                    #パターンご$

G1=G2=G3 G1>G2=G3 G1>G2>G3 G1>G3>G2 G2>G1=G3
       0       46     1499      533       51
G2>G1>G3 G2>G3>G1 G3>G1=G2 G3>G1>G2 G3>G2>G1
    1588      580      114     1440     1385
> table(res$ordering[obj])/length(res$ordering[obj])$

   G1=G2=G3    G1>G2=G3    G1>G2>G3    G1>G3>G2
0.000000000 0.006357103 0.207158651 0.073659480
   G2>G1=G3    G2>G1>G3    G2>G3>G1    G3>G1=G2
0.007048093 0.219458264 0.080154782 0.015754561
   G3>G1>G2    G3>G2>G1
0.199004975 0.191404091
>
```

②

③

ほらね

①のTCCと、②のbaySeqの結果は、FDR = 0.05を満たす遺伝子数に随分違いがあるものの、発現パターン分類結果では似てますね、的なDiscussionができる

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

他のDiscussion1

他には、例えば③baySeqでG1=G2=G3というパターンに分類されたのは、2,803個(13.5%)。②baySeqのFDR = 0.05を満たす遺伝子中に、このパターンに含まれるのが0個だったのはほぼ当然と言っていいだろう。注目すべきは、①TCC (*EEE–E*)を含む他の全ての方法の結果でも、FDR = 0.05を満たす遺伝子群の中にG1=G2=G3のパターンのものは1つもなかった。これはリーズナブルと言えるでしょう、というDisucussionを④で行っています

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2						
all_genes	13.5	2.2	15.1	8.7						
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.						19.2	7247
DDD-D	0.0	0.4	25.0	7.						17.1	3850
SSS-S	0.0	0.2	19.3	7.						21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.						19.3	7247
edgeR_robust	0.0	0.3	20.6	8.						18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.						18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.						19.9	7585
voom	0.0	0.7	21.3	7.						19.0	7016
SAMseq	0.0	0.2	20.9	9.						18.3	9453
PoissonSeq	0.0	0.0	19.5	8.						19.3	6613
baySeq	0.0	0.8	21.0	5.						19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

We next classified the expression patterns of the DEGs obtained from the 12 pipelines (Table 4). We here assigned individual DEGs to one of the ten possible patterns defined in baySeq [22]; this package returns one of these patterns to each gene. The *background* information for this data is shown in the "all_genes" row in Table 4. The "common" row indicates the percentages of individual expression patterns for the 2376 common DEGs. The remaining rows (from *EEE-E* to EBSeq) show the distributions for each of the pipelines. It is reasonable that no DEGs identified by individual pipelines are assigned as a flat expression pattern (i.e., G1 = G2 = G3) for the HS (G1) vs. PT (G2) vs. RM (G3) comparison. We found that most DEGs were assigned preferably to one of four patterns

Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

他のDiscussion2

赤枠の4つのパターン（①G1>G2>G3, ②G2>G1>G3, ③G3>G1>G2, ④G3>G2>G1）に属する遺伝子の割合は、多い

Table 4 – Classification of expression patterns for DEGs

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3 ①	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3 ②	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2 ③	G3 > G2 > G1 ④	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

他のDiscussion2

赤枠の2つのパターン（⑤G1>G3>G2, ⑥G2>G3>G1）に属する遺伝子数は、相対的に少ない。この理由はG1(HS)とG2(PT)の発現パターン間の類似度が、G3(RM)との類似度に比べて高いということで説明がつく

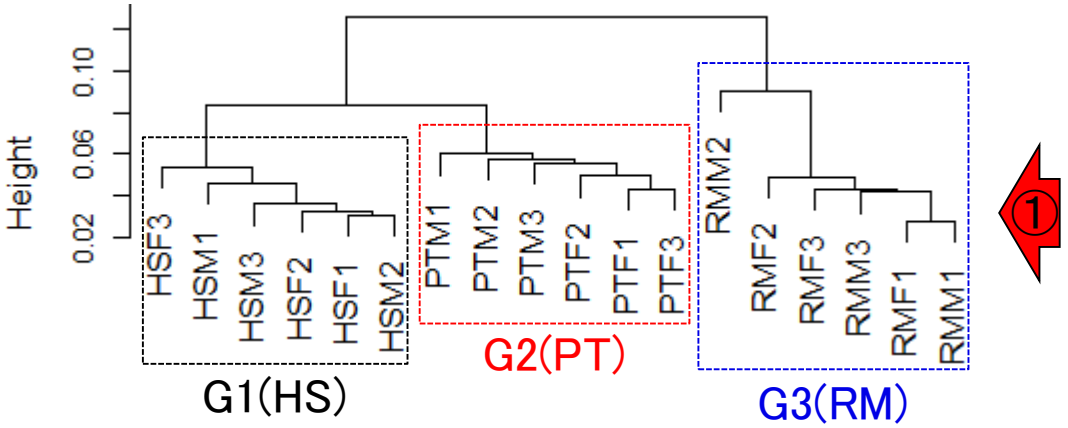
Table 4 – Classification of expression patterns for ⑤ ⑥

	G1 = G2 = G3	G1 > G2 = G3	G1 > G2 > G3	G1 > G3 > G2	G2 > G1 = G3	G2 > G1 > G3	G2 > G3 > G1	G3 > G1 = G2	G3 > G1 > G2	G3 > G2 > G1	Total
all_genes	13.5	2.2	15.1	8.7	2.3	15.9	9.4	2.9	15.1	14.8	20689
common	0.0	0.1	23.2	5.8	0.2	26.4	5.7	0.7	18.6	19.2	2376
EEE-E	0.0	0.6	20.7	7.4	0.7	21.9	8.1	1.6	19.9	19.2	7247
DDD-D	0.0	0.4	25.0	7.3	0.6	25.0	6.0	1.4	17.3	17.1	3850
SSS-S	0.0	0.2	19.3	7.1	0.3	21.7	9.4	0.9	19.9	21.2	7295
E-E (edgeR)	0.0	0.6	20.4	7.3	0.7	22.1	8.3	1.6	19.7	19.3	7247
edgeR_robust	0.0	0.3	20.6	8.4	0.5	22.0	8.8	1.2	19.1	18.9	8076
D-D (DESeq)	0.0	0.4	24.3	7.2	0.6	24.2	6.0	1.4	17.8	18.1	3832
S-S (DESeq2)	0.0	0.2	20.4	8.0	0.3	21.8	8.9	0.8	19.7	19.9	7585
voom	0.0	0.7	21.3	7.7	0.7	22.5	8.2	1.3	18.7	19.0	7016
SAMseq	0.0	0.2	20.9	9.7	0.3	21.8	9.2	0.8	18.9	18.3	9453
PoissonSeq	0.0	0.0	19.5	8.9	0.1	22.2	9.4	0.3	20.3	19.3	6613
baySeq	0.0	0.8	21.0	5.5	1.3	23.7	6.3	2.8	19.0	19.6	3975
EBSeq	0.0	0.0	21.0	7.0	0.1	23.7	7.1	0.3	20.8	19.9	5699

Percentages of genes assigned to each of the ten possible patterns defined as baySeq. Numbers in the "Total" column indicate the numbers of genes. For example, baySeq assigned 13.5 % of 20,689 genes as "G1 = G2 = G3."

他のDiscussion2

つまり、①このサンプル間クラスタリング結果の解釈と同じ。②原著論文中では、赤下線のようなDiscussionを行っている。パターン分類結果の解釈として、こんな感じのことが書けますよという一つの指針を述べただけです



①

We next classified the expression patterns of the DEGs obtained from the 12 pipelines (Table 4). We here assigned individual DEGs to one of the ten possible patterns defined in baySeq [22]; this package returns one of these patterns to each gene. The *background* information for this data is shown in the "all_genes" row in Table 4. The "common" row indicates the percentages of individual expression patterns for the 2376 common DEGs. The remaining rows (from *EEE-E* to EBSeq) show the distributions for each of the pipelines. It is reasonable that no DEGs identified by individual pipelines are assigned as a flat expression pattern (i.e., G1 = G2 = G3) for the HS (G1) vs. PT (G2) vs. RM (G3) comparison. We found that most DEGs were assigned preferably to one of four patterns (G1 > G2 > G3, G2 > G1 > G3, G3 > G1 > G2, and G3 > G2 > G1) and unpreferably to one of two patterns (G1 > G3 > G2 and G2 > G3 > G1). That is, up- (or down-) regulation in G1 for DEGs tends to coincide with G2 more than G3. This can also be seen in the results from sample clustering of the raw count data (see Additional file 6), implying that we can roughly predict the DE results such as those shown in Table 4 from the overall similarities of samples on the raw count data.

②

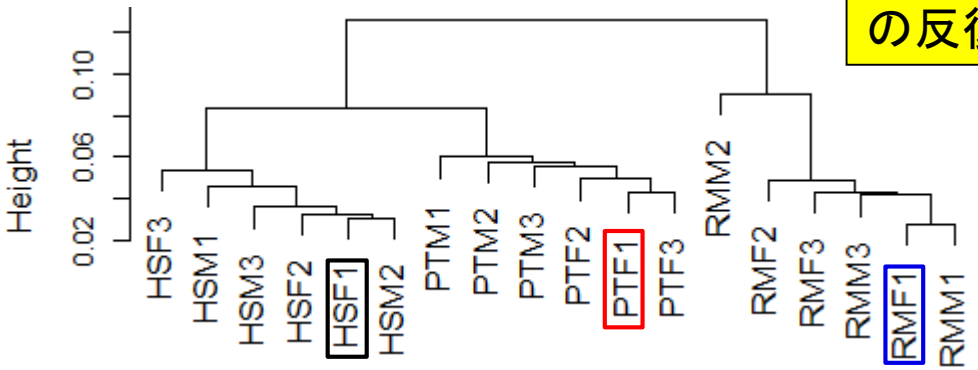
Tang et al., *BMC Bioinformatics*, **16**: 361, 2015

Contents

- 反復あり3群間比較（TCCによるANOVA的な解析）
- デザイン行列、post-hoc test
  - 「G1 vs. G2」、「G1 vs. G3」、「G2 vs. G3」、コントラストで「G1 vs. G2」と「G1 vs. G3」
  - 「Post-hoc testの2群間比較」と「通常の2群間比較」の違い
- 遺伝子間クラスタリング
  - MBCluster.Seq単体での利用
  - 「TCC正規化 + MBCluster.Seq」
  - 「TCC正規化 + MBCluster.Seq」とTCC発現変動解析の組み合わせ
- 反復あり3群間比較（EBSeqやbaySeqによる発現パターン分類）
  - baySeq
  - TCC結果を含めてbaySeqの発現パターン分類結果上で議論する
- 反復なし3群間比較(TCC)、および結果の解釈

解析データ

sample_blekhman_18.txtを入力として、①1列目のHSF1、②7列目のPTF1、③13列目のRMF1の3列分のデータのみ取り出し、「G1群(HSF1) vs. G2群(PTF1) vs. G3群(RMF1)」の反復なし3群間比較を行う



ヒト（*Homo sapiens*; HS）　チンパンジー（*Pan troglodytes*; PT）　アカゲザル（*Rhesus macaque*; RM）

20,689 genes

	HSF1	HSF2	HSF3	HSM1	HSM2	HSM3	PTF1	PTF2	PTF3	PTM1	PTM2	PTM3	RMF1	RMF2	RMF3	RMM1	RMM2	RMM3
ENSG00000000003	329	300	168	121	421	359	574	429	386	409	685	428	511	464	480	424	1348	705
ENSG00000000005	0	0	0	0	1	0	1	4	1	0	1	1	0	1	2	2	0	0
ENSG00000000419	81	61	56	39	78	62	100	66	65	59	58	93	67	72	57	49	82	90
ENSG00000000457	91	62	76	114	73	95	131	229	87	274	239	149	89	69	118	117	114	163
ENSG00000000460	6	17	12	15	7	17	8	8	5	12	7	10	4	4	10	7	3	4
ENSG00000000938	44	65	210	73	43	65	84	104	76	198	31	58	73	28	54	80	34	72
ENSG00000000971	4765	7225	3405	3600	6383	5546	5382	8331	4335	2568	5019	2653	13566	9964	18247	14236	5196	11834
ENSG00000001036	297	251	189	200	234	249	305	301	313	254	151	331	292	106	379	201	88	140
ENSG00000001084	630	737	306	336	984	459	417	328	885	298	569	218	1062	786	1110	873	664	1752
ENSG00000001167	36	30	36	29	33	28	63	80	25	69	74	41	62	34	108	97	35	61
ENSG00000001460	3	1	5	1	4	2	0	1	1	1	1	3	1	1	1	0	1	3
ENSG00000001461	49	37	34	28	62	32	75	69	40	90	69	60	210	92	176	247	81	117
ENSG00000001497	117	93	89	80	131	110	125	99	75	108	130	131	139	95	197	137	159	172

① ② ③

反復なし3群間(TCC)

②例題4。③ここで抽出したいデータ列を指定（ここでは1, 7, 13列目）

- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | edgeR(Robinson_2010) (last modified 2015/02/03)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) (last modified 2016/05/31)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+baySeq(Hardcastle_2010) (l
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+EBSeq(Leng_2013) (last mo
- 解析 | 発現変動 | 3群間 | 対応なし | 複製なし | DESeq2(Love_201 (last modified 2016/06/01)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製なし | TCC(Sun_2013) ① modified 2016/06/01)推奨
- 解析 | 発現変動 | 5群間 | 対応なし | 複製あり | TCC(Sun_2013) (st modified 2015/11/05)推奨
- 解析 | 発現変動 | 時系列 | について (last modified 2015/11/11)
- 解析 | 発現変動 | 時系
- 解析 | 発現変動 | 時系
- 解析 | 発現変動 | exor
- 解析 | 発現変動 | exor
- 解析 | 機能解析 | 遺伝
- 解析 | 機能解析 | 遺伝

解析 | 発現変動 | 3群間 | 対応なし | 複製なし | TCC(Sun_2013) NEW

TCCを用いたやり方を示します。内部的にDESeq2パッケージ中の関数を利用して、複製なしデータに対応済みのDESeq2の通常の手順を複数回繰り返す(DEGES-based normalization; Kadota et al., 2012)ことでより正確なデータ正規化が実現された発現変動解析結果を得ることができます。TCC原著論文(Sun et al., BMC Bioinformatics, 2013)では 3群間複製なしデータ用の推奨パイプラインを示していませんでしたが、多群間比較用の推奨ガイドライン提唱論文 (Tang et al., BMC Bioinformatics, 2015) で推奨しているパイプライン"SSS-S"が"iDEGES/DES g2-DESeq2"と同じものです。この2つの論文を引用し、安心してご利用ください。これに関連して、TCC ver. 1.8 ② DESeq が動いていました なっています。
「ファイル」−「ディレクトリの変

1. サンプルデータ43の10,000

シミュレーションデータ(G1群 がDEG (gene_1〜gene_210 gene_2701〜gene_3000がG す。約1分かかります。

```
in_f <- "data_hypoda
```

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3群間比較(HSF1 vs. PTF1 vs. RMF1)としています。例題3と基本的に同じですが、正規化後のデータで発現変動順にソートした結果を出力しています。

```
in_f <- "sample_blekhman_18.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_subset <- c(1, 7, 13)               #取り扱いたいサブセット情報を指定
param_G1 <- 1                             #G1群のサンプル数を指定
param_G2 <- 1                             #G2群のサンプル数を指定
param_G3 <- 1                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し
```

③

反復なし3群間(TCC)

①例題4のコピペ実行結果。かなり緩めのFDR閾値でもDEGが検出されない。理由は、2016.07.21の最後で議論した内容と同じ

①

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3群間比較(HSF1 vs. PTF1 vs. RMF1)としています。例題3と基本的に同じですが、正規化後のデータで発現変動順にソートした結果を出力しています。

```
in_f <- "sample_blekhman_18.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_subset <- c(1, 7, 13)               #取り扱いたいサブセット情報を指定
param_G1 <- 1                             #G1群のサンプル数を指定
param_G2 <- 1                             #G2群のサンプル数を指定
param_G3 <- 1                             #G3群のサ
param_FDR <- 0.05                         #DEG検出時

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=

#前処理(サブセットの抽出とTCCクラスオブジェクトの作成
data <- data[,param_subset]               #param_su
data.cl <- c(rep(1, param_G1), rep(2, param_G2),
tcc <- new("TCC", data, data.cl)          #TCCクラス
colnames(data)                            #列名を表示

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="deseq2"
```

R Console

```
> #ファイルに保存(テキストファイル)
> tmp <- cbind(rownames(tcc$count), normalized, p.va$
> tmp <- tmp[order(ranking),]                #発現変動順$
> write.table(tmp, out_f, sep="\t", append=F, quote=$
>
> #様々なFDR閾値を満たす遺伝子数を表示
> sum(q.value < 0.05)                        #FDR = 0.05$
[1] 0
> sum(q.value < 0.10)                        #FDR = 0.10$
[1] 0
> sum(q.value < 0.30)                        #FDR = 0.30$
[1] 0
> sum(q.value < 0.50)                        #FDR = 0.50$
[1] 0
> sum(q.value < 0.70)                        #FDR = 0.70$
[1] 0
>
```

反復なし3群間(TCC)

①出力ファイル(hoge4.txt)を概観。②1位のq-valueも1なので、DEGはないと判断せざるを得ない。が、反復なしの場合の統計的手法出力結果の特徴を論理的に述べられれば、上位x個についての議論を行うのはアリだろう。③発現プロファイル自体は、確かに発現変動順になっているようなので妥当

4. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3vs. PTF1 vs. RMF1)としています。例題3と基本的に同じですが、正規化後のデータで発現結果を出力しています。

```
in_f <- "sample_blekhman_18.txt"          #入力ファイル名を指定してin_fに
out_f <- "hoge4.txt"                      #出力ファイル名を指定してout_fに格納
param_subset <- c(1, 7, 13)               #取り扱いたいサブセット情報を指定
param_G1 <- 1                             #G1群のサンプル数を指定
param_G2 <- 1                             #G2群のサンプル数を指定
param_G3 <- 1                             #G3群のサンプル数を指定
param_FDR <- 0.05                         #DEG検出時のfalse discovery rate (FDR)閾値を

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

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, s

#前処理(サブセットの抽出とTCCクラスオブジェクトの作成)
data <- data[,param_subset]               #param_subset
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep
tcc <- new("TCC", data, data.cl)          #TCCクラスオブ
colnames(data)                            #列名を表示

#本番(正規化)
tcc <- calcNormFactors(tcc, norm.method="deseq2",#正
```

	HSF1	PTF1	RMF1	p.value	q.value	ranking
ENSG00000165272	430.2	18.6	3.9	0.0195	1.0000	1
ENSG00000187608	67.3	1995.8	58.8	0.0197	1.0000	2
ENSG00000130513	20.4	830.3	24.1	0.0198	1.0000	3
ENSG00000134202	206.7	0.9	4.8	0.0203	1.0000	4
ENSG00000085552	224.7	0.9	7.7	0.0206	1.0000	5
ENSG00000198846	3.6	4.4	268.1	0.0208	1.0000	6
ENSG00000215218	1.2	15.9	272.0	0.0216	1.0000	7
ENSG00000170775	199.5	5.3	1.0	0.0218	1.0000	8
ENSG00000126838	713.7	90.2	5800.0	0.0220	1.0000	9
ENSG00000145708	408.5	634.0	3.9	0.0222	1.0000	10
ENSG00000009694	187.4	4.4	1.0	0.0223	1.0000	11
ENSG00000140505	942.1	5717.7	76.2	0.0223	1.0000	12
ENSG00000126709	42.1	2909.2	457.1	0.0233	1.0000	13
ENSG00000139899	30.0	23.0	760.9	0.0235	1.0000	14
ENSG00000133048	109.3	1085.9	6752.9	0.0238	1.0000	15

反復なし3群間(DESeq2)

おまけ。①DESeq2だとどうなるのか一応確認。②例題2

- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 基礎 | edgeR(Robinson_2010) (last modified 2015/02/03)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC(Sun_2013) (last modified 2016/05/31)推奨
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+baySeq(Hardcastle_2010) (l
- 解析 | 発現変動 | 3群間 | 対応なし | 複製あり | 応用 | TCC正規化(Sun_2013)+EBSeq(Leng_2013) (last mo
- 解析 | 発現変動 | 3群間 | 対応なし | 複製なし | DESeq2(Love_2014) (last modified 2016/06/01)
- 解析 | 発現変動 | 3群間 | 対応なし | 複製なし | TCC(Sun_2013) (last modified 2016/06/01)推奨
- 解析 | 発現変動 | 5群間 | 対応なし | 複製あり | TCC(Sun_2013) (last modified 2015/11/05)推奨
- 解析 | 発現変動 | 時系列 |について (last modified 2015/11/11)
- 解析 | 発現変動 | 時
- 解析 | 発現変動 | 時
- 解析 | 発現変動 | e
- 解析 | 発現変動 | e
- 解析 | 機能解析 | 遺
- 解析 | 機能解析 | 遺

①

解析 | 発現変動 | 3群間 | 対応なし | 複製なし | DESeq2(Love_2014) NEW

DESeq2パッケージ (Love et al., Genome Biol., 2014)を用いるやり方を示します。
「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43の10,000 genes×3 samplesのカウントデータ(data_hypodata_1vs1vs1.txt)の場合:

シミュレーションデータ(G1群
がDEG (gene_1～gene_2100
gene_2701～gene_3000がG3群
す。約1分かかります。

```
in_f <- "data_hypodata_
out_f <- "hoge1.txt"
param_G1 <- 1
param_G2 <- 1
param_G3 <- 1

#必要なパッケージをロード
library(DESeq2)

#入力ファイルの読み込み
data <- read.table(in_f
```

②

2. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3群間比較(HSF1 vs. PTF1 vs. RMF1)としています。正規化後のデータで発現変動順にソートした結果を出力しています。

```
in_f <- "sample_blekhman_18.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_subset <- c(1, 7, 13)               #取り扱いたいサブセット情報を指定
param_G1 <- 1                             #G1群のサンプル数を指定
param_G2 <- 1                             #G2群のサンプル数を指定
param_G3 <- 1                             #G3群のサンプル数を指定

#必要なパッケージをロード
library(DESeq2)                           #パッケージの読み込み

#入力ファイルの読み込みとサブセットの作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し
data <- data[,param_subset]               #param_subsetで指定した列の情報のみ抽出

#前処理(DESeqDataSetオブジェクトの作成)
                                                    rep(3, param_G3))#G1群を1, G2群を
```


Love et al., *Genome Biol.*, **15**: 550, 2014

反復なし3群間(DESeq2

①

2. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の
です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3群
vs. PTF1 vs. RMF1)としています。正規化後のデータで発現変動順にソートした結果を出力し

```
#本番(DEG検出)
d <- DESeq(d)                                #DESeq2を実行
#d <- estimateSizeFactors(d)                 #正規化を実行した結果をdに格納
##sizeFactors(d) <- sizeFactors(d)/mean(sizeFactors(d))#size factorsをさらに正規化(
#d <- estimateDispersions(d)                 #モデル構築(ばらつきの程度を見積もっている)
#d <- nbinomLRT(d, full= ~condition, reduced= ~1)#検定
normalized <- counts(d, normalized=T)        #正規化後のデータを取り出してnormalizedに格納

tmp <- results(d)                            #実行結果を抽出
p.value <- tmp$pvalue                        ##p-valueをp.valueに格納
p.value[is.na(p.value)] <- 1                 #NAを1に置換している
q.value <- tmp$padj                          #adjusted p-valueをq.valueに格納
q.value[is.na(q.value)] <- 1                 #NAを1に置換している
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(テキストファイル)
tmp <- cbind(rownames(data), normalized, p.value, q.value, ranking)#正規化後のデー
tmp <- tmp[order(ranking),]                  #発現変動順にソートした結果を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

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                          #FDR = 0.05 (q-value < 0.05)を満たす遺伝子数
sum(q.value < 0.10)                          #FDR = 0.10 (q-value < 0.10)を満たす遺伝子数
sum(q.value < 0.30)                          #FDR = 0.30 (q-value < 0.30)を満たす遺伝子数
sum(q.value < 0.50)                          #FDR = 0.50 (q-value < 0.50)を満たす遺伝子数
sum(q.value < 0.70)                          #FDR = 0.70 (q-value < 0.70)を満たす遺伝子数
sum(q.value < 0.80)                          #FDR = 0.80 (q-value < 0.80)を満たす遺伝子数
sum(p.adjust(p.value, method="BH") < 0.05)#FDR閾値(q.value < 0.05)を満たす遺伝子数を
sum(p.adjust(p.value, method="BH") < 0.10)#FDR閾値(q.value < 0.10)を満たす遺伝子数を
sum(p.adjust(p.value, method="BH") < 0.30)#FDR閾値(q.value < 0.30)を満たす遺伝子数を
sum(p.adjust(p.value, method="BH") < 0.50)#FDR閾値(q.value < 0.50)を満たす遺伝子数を
sum(p.adjust(p.value, method="BH") < 0.70)#FDR閾値(q.value < 0.70)を満たす遺伝子数を
sum(p.adjust(p.value, method="BH") < 0.80)#FDR閾値(q.value < 0.80)を満たす遺伝子数を
```

②

③

①例題2。ここでの主な目的は、「反復なし2群間比較」のDESeq2実行結果で見られたq-value計算方法による違い(2016.07.21のスライド148)が、「反復なし3群間比較」でも見られるのかどうかの確認。③がDESeq2オリジナルのadjusted p-valueによるFDR閾値を満たす遺伝子数表示部分

反復なし3群間(DESe

③は、②DESeq2のp-value情報をもとに、TCCが採用しているp.adjust関数を用いたBH法によるq-value算出手順に従った結果。結論としては同じ

2. リアルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(R
です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復な
vs. PTF1 vs. RMF1)としています。正規化後のデータで発現変動順にソートした結果を出

```
#本番(DEG検出)
d <- DESeq(d)                                  #DESeq2を実行
#d <- estimateSizeFactors(d)                   #正規化を実行した結果をdに格納
##sizeFactors(d) <- sizeFactors(d)/mean(sizeFactors(d))#size factor
#d <- estimateDispersions(d)                   #モデル構築(ばらつきの程度を見
#d <- nbinomLRT(d, full= ~condition, reduced= ~1)#検定
normalized <- counts(d, normalized=T)          #正規化後のデータを取り出して

tmp <- results(d)                              #実行結果を抽出
p.value <- tmp$pvalue                          ##p-valueをp.valueに格納
p.value[is.na(p.value)] <- 1                   #NAを1に置換している
q.value <- tmp$padj                            #adjusted p-valueをq.valueに
q.value[is.na(q.value)] <- 1                   #NAを1に置換している
ranking <- rank(p.value)                       #p.valueでランキングした結果を

#ファイルに保存(テキストファイル)
tmp <- cbind(rownames(data), normalized, p.value, q.value, ranking
tmp <- tmp[order(ranking),]                    #発現変動順にソートした結果を
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#

#様々なFDR閾値を満たす遺伝子数を表示
sum(q.value < 0.05)                            #FDR = 0.05 (q-value < 0.05
sum(q.value < 0.10)                            #FDR = 0.10 (q-value < 0.10
sum(q.value < 0.30)                            #FDR = 0.30 (q-value < 0.30
sum(q.value < 0.50)                            #FDR = 0.50 (q-value < 0.50
sum(q.value < 0.70)                            #FDR = 0.70 (q-value < 0.70
sum(q.value < 0.80)                            #FDR = 0.80 (q-value < 0.80
sum(p.adjust(p.value, method="BH") < 0.05)     #FDR閾値(q.value < 0.05)
sum(p.adjust(p.value, method="BH") < 0.10)     #FDR閾値(q.value < 0.10)
sum(p.adjust(p.value, method="BH") < 0.30)     #FDR閾値(q.value < 0.30)
sum(p.adjust(p.value, method="BH") < 0.50)     #FDR閾値(q.value < 0.50)
sum(p.adjust(p.value, method="BH") < 0.70)     #FDR閾値(q.value < 0.70)
sum(p.adjust(p.value, method="BH") < 0.80)     #FDR閾値(q.value < 0.80)
```

R Console

```
> #様々なFDR閾値を満たす遺伝子数を表$
> 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] 0
> sum(q.value < 0.80)                          $
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
> sum(p.adjust(p.value, method="BH")$
[1] 0
>
```

反復なし3群間(DESeq2)

①出力ファイル(hoge2.txt)を概観。②1位のq-valueも1なので、DEGはないと判断する。③発現プロファイル自体は、確かに発現変動順になっているようなので妥当

2. サンプルデータ42のリアルデータ(sample_blekhman_18.txt)の場合:

20,689 genes×18 samplesのカウントデータで、ヒト(HS)、チンパンジー(PT)、アカゲザル(RM)の3生物種間比較です (Blekhman et al., Genome Res., 2010)。1, 7, 13列目のデータのみ抽出して、反復なし3群間比較(HSF1 vs. PTF1 vs. RMF1)としています。正規化後のデータで発現変動順にソートした結果を出力しています。

```
in_f <- "sample_blekhman_18.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_subset <- c(1, 7, 13)               #取り扱いたいサブセット情報を指定
param_G1 <- 1                             #G1群のサンプル数を指定
param_G2 <- 1                             #G2群のサンプル数を指定
param_G3 <- 1                             #G3群のサンプル数を指定

#必要なパッケージをロード
library(DESeq2)                           #パッケージの読み込み

#入力ファイルの読み込みとサブセットの作成
data <- read.table(in_f, header=TRUE, row.names=1, s
data <- data[,param_subset]               #param_subset

#前処理(DESeqDataSetオブジェクトの作成)
data.cl <- c(rep(1, param_G1), rep(2, param_G2), rep
colData <- data.frame(condition=as.factor(data.cl))#
d <- DESeqDataSetFromMatrix(countData=data, colData=

#本番(DEG検出)
d <- DESeq(d)                             #DESeq2を実行
#d <- estimateSizeFactors(d)              #正規化を実行し
```

①

③

②

	HSF1	PTF1	RMF1	p.value	q.value	ranking
ENSG00000133048	107.9	1067.3	6716.6	0.0048	1.0000	1
ENSG00000137154	564.4	127.8	6.7	0.0072	1.0000	2
ENSG00000165272	424.5	18.3	3.8	0.0074	1.0000	3
ENSG00000170345	1993.0	237.3	67.1	0.0078	1.0000	4
ENSG00000108515	353.3	62.6	4.8	0.0082	1.0000	5
ENSG00000142871	521.7	226.0	13.4	0.0085	1.0000	6
ENSG00000101104	271.5	43.5	5.8	0.0098	1.0000	7
ENSG00000148795	16.6	836.9	1551.1	0.0099	1.0000	8
ENSG00000215218	1.2	15.6	270.5	0.0103	1.0000	9
ENSG00000104267	954.4	557.1	45.1	0.0104	1.0000	10
ENSG00000120738	1381.3	455.4	84.4	0.0109	1.0000	11
ENSG00000198846	3.6	4.3	266.7	0.0115	1.0000	12
ENSG00000170775	196.8	5.2	1.0	0.0121	1.0000	13
ENSG00000104332	87.7	100.8	2181.3	0.0122	1.0000	14
ENSG00000009694	185.0	4.3	1.0	0.0127	1.0000	15

TCC vs. DESeq2

2つの方法ともに、上位15位まででざっと見ても6個共通しており、妥当といえば妥当。赤枠で示すように、正規化後のデータは若干異なる。その理由は、(反復なしデータの場合)TCCは内部的にDESeq2の通常の解析手順を3回繰り返して頑健な正規化を行う処理をしているから。それゆえ、p-valueが多少異なるのも当然

TCC (*SSS-S*)

	HSF1	PTF1	RMF1	p.value
ENSG00000165272	430.2	18.6	3.9	②195
ENSG00000187608	67.3	1995.8	58.8	0.0197
ENSG00000130513	20.4	830.3	24.1	0.0198
ENSG00000134202	206.7	0.9	4.8	0.0203
ENSG00000085552	224.7	0.9	7.7	0.0206
ENSG00000198846	3.6	4.4	268.1	③208
ENSG00000215218	④1.2	15.9	272.0	0.0216
ENSG00000170775	99.5	5.3	1.0	⑤218
ENSG00000126838	713.7	90.2	5800.0	0.0220
ENSG00000145708	408.5	634.0	3.9	0.0222
ENSG00000009694	187.4	4.4	1.0	⑥223
ENSG00000140505	942.1	5717.7	76.2	0.0223
ENSG00000126709	42.1	2909.2	457.1	0.0233
ENSG00000139899	30.0	23.0	760.9	0.0235
ENSG00000133048	109.3	1085.9	6752.9	①238

DESeq2 (*S-S*)

	HSF1	PTF1	RMF1	p.value	q.value	ranking
ENSG00000133048	107.9	1067.3	6716.6	①048	1.0000	1
ENSG00000137154	564.4	127.8	6.7	0.0072	1.0000	2
ENSG00000165272	424.5	18.3	3.8	②074	1.0000	3
ENSG00000170345	1993.0	237.3	67.1	0.0078	1.0000	4
ENSG00000108515	353.3	62.6	4.8	0.0082	1.0000	5
ENSG00000142871	521.7	226.0	13.4	0.0085	1.0000	6
ENSG00000101104	271.5	43.5	5.8	0.0098	1.0000	7
ENSG00000148795	16.6	836.9	1551.1	0.0099	1.0000	8
ENSG00000215218	④1.2	15.6	270.5	0.0103	1.0000	9
ENSG00000104267	54.4	557.1	45.1	0.0104	1.0000	10
ENSG00000120738	1381.3	455.4	84.4	0.0109	1.0000	11
ENSG00000198846	3.6	4.3	266.7	③115	1.0000	12
ENSG00000170775	⑤6.8	5.2	1.0	0.0121	1.0000	13
ENSG00000104332	87.7	100.8	2181.3	0.0122	1.0000	14
ENSG00000009694	185.0	4.3	1.0	⑥127	1.0000	15