Rでトランスクリプトーム解析

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ユニット
門田 幸二（かどた　こうじ）
http://www.iu.a.u-tokyo.ac.jp/~kadota/
kadota@iu.a.u-tokyo.ac.jp

自己紹介

- 2002年3月
  - 東京大学・大学院農学生命科学研究科　博士課程修了
  - 学位論文：「cDNAマイクロアレイを用いた遺伝子発現解析手法の開発」（指導教官：清水謙多郎教授）
- 2002/4/1~
  - **産総研・生命情報科学研究センター（CBRC）** 産総研特別研究員
- 2003/11/1~
  - 放医研・先端遺伝子発現研究センター　研究員
- 2005/2/16~
  - 東京大学・大学院農学生命科学研究科　特任助手
- 2007/4/1~現在
  - 東京大学・大学院農学生命科学研究科　特任助教

} アグリバイオインフォマティクスプログラム

様々なMotivation

- ～の原因遺伝子（ガン関連遺伝子とか）を同定したい
- FASTQ以降の一通りの解析ができるようになりたい
- （Windowsの）Rでできることとできないこと
- モデル生物と非モデル生物の解析戦略の違い
- 倍率変化で解析 vs. 分布を使って解析
- いろんなRパッケージがあるけれど…

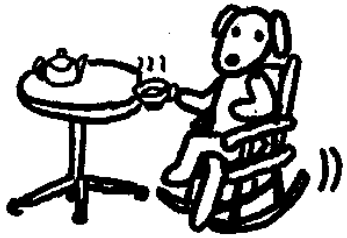
RNA-seqで二つのサンプルを比較し、発現変動遺伝子同定までを行うまでの流れを一通り紹介

A群
腎臓

正常組織
wildtype

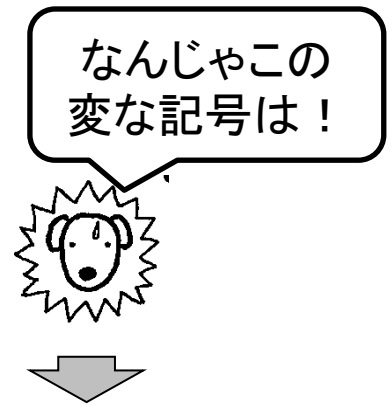
B群
肝臓

腫瘍組織
mutant

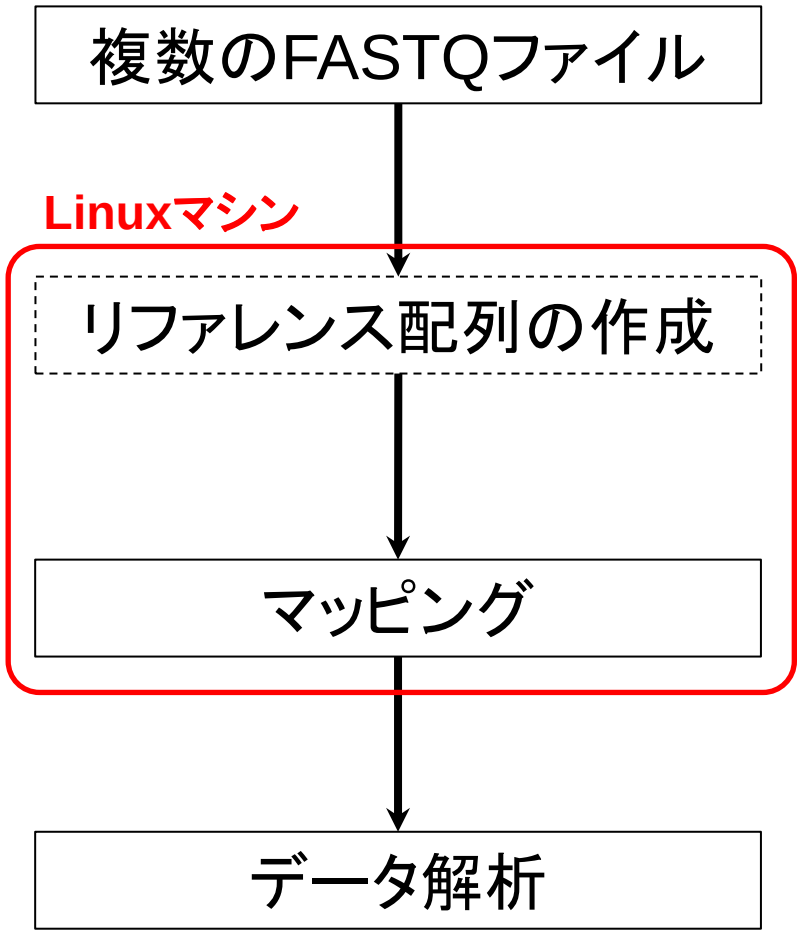


データ解析のスタート地点

- NGSから得られたFASTQ形式ファイル



比較トランスクリプトーム解析の流れ



A1.fq, A2.fq, B1.fq, B2.fq

複数サンプルの混合アセンブルにより、RefSeqのような転写物配列集合（multi-fastaファイル）を得るイメージ

どの転写物にどれだけの数のリードがマップされたかという、いわゆる「遺伝子発現行列」を得るイメージ

発現変動遺伝子のリストアップや、作図など

すべてが（Windowsの）Rでできるわけではありません

Linuxマシン使用部分の解決策

- 自前で大容量メモリ計算サーバ(Linux)を購入し、必要なソフトのインストールからスタート

  特徴:難易度は高いが思い通りの解析が可能

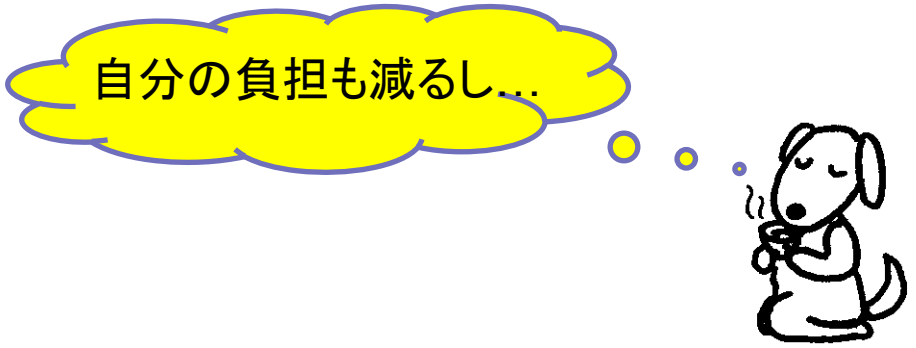
- Linuxサーバをもつバイオインフォ系の人にお願いする

  特徴:気軽に頼める知り合いがいればいいが、その人次第

- DDBJ Read Annotation Pipelineを利用

  特徴:一番お手軽な選択肢だが、サポートしているプログラムのみ

  データ登録が前提?!だが、手取り足取り丁寧に教えてくれるので個人的にはこちらを推奨



DDBJ
DNA Data Bank of Japan

DDBJ Read Annotation Pipeline

>>English

DDBJ Read Annotation Pipelineは、次世代シーケンサ配列のクラウド型データ解析プラットフォームです。

LOGIN

新規アカウント作成　guest としてログイン

User ID:

Password:

Login

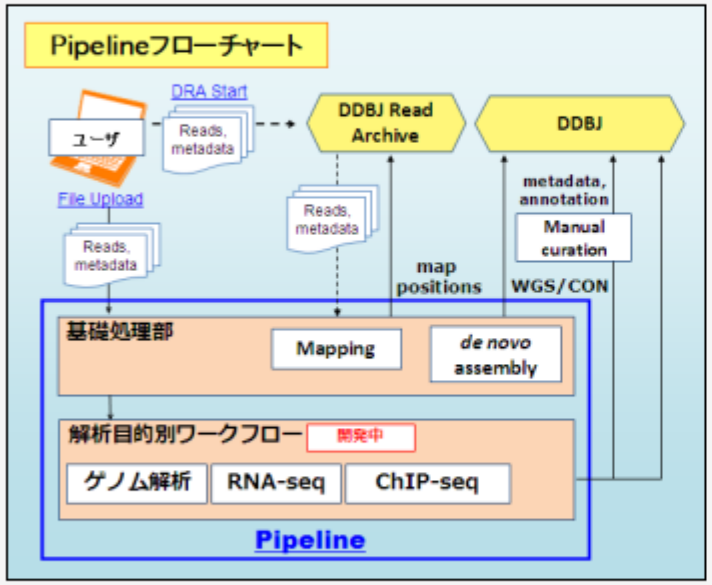
新規アカウントの作成は こちらです。

動作中JOBの確認

PipelineのIDをお持ちでない場合、
ゲストとしてログインすることができます。

User ID, Passwordの入力は、Query Fileの指定方法により異なります。

DRAへ登録したQueryファイルを使用する場合
　DRA登録後にDRAのID、Passwordでログイン
Queryファイルをアップロードして使用する場合
　新規アカウントを作成（DRAの登録なしでご利用になれます。）



マニュアルおよびチュートリアル

- 日本語マニュアル
- Manual(英語)
- FTPクライアント資料
- DBCLS 統合TV チュートリアル1 - 今日からはじめるDDBJ Read Annotation Pipeline

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

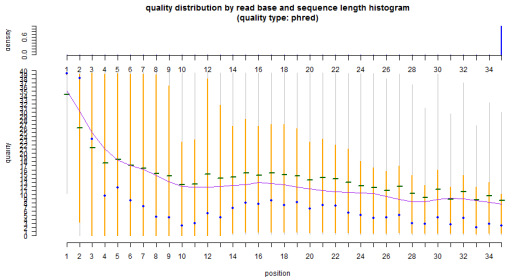
Linuxマシン

リファレンス配列の作成

マッピング

データ解析

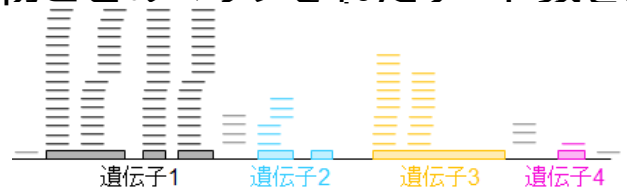
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

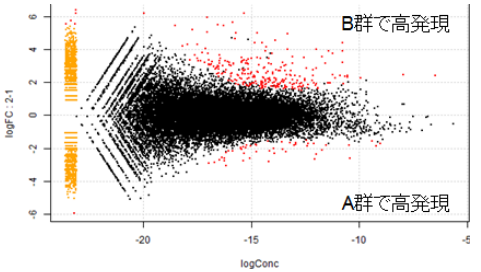
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



大規模計算部分以外は一通りできます

参考ウェブページ

(Rで)塩基配列解析(主に次世代シーケンサーのデータ) by 門田幸二 (last modified 2012/02/06)

What's new?
・2/22の横浜理研のRでつなぐ次世代オミックス情報統合解析研究会では(しゃべり時間50分しかないので)デモのみ行います。また、3/9のお台場でのHPCI チュートリアルセミナーでは(4時間ほどあるので)実際に手を動かしてもらいます。また、時間があれば私の最新の手法の使用法も伝授できればと思っています。興味ある方はどうぞ。(2012/02/03)NEW

・最新のパッケージ中の使用法の記述への更新が相当遅れていますので、ご利用時はもとのパッケージ中のマニュアルを各自チェックしてください。(2011/12/26)
・R2.14.1がリリースされていたのでこれに変更しました。(2011/12/26)

- はじめに (last modified 2011/07/19)
- Rのインストールと起動(last modified 2012/01/04)
- サンプルデータ(last modified 2011/02/03)
- イントロダクション | NGS | 各種覚書(last modified 2010/12/10)
- イントロダクション | NGS | 様々なプラットフォーム(last modified 2011/07/15)
- イントロダクション | NGS | リファレンス配列取得(マップされる側)(last modified 2011/02/03)
- イントロダクション | NGS | リファレンス配列取得後の各種情報抽出(特にRefSeq)(last modified 2011/03/20)
- イントロダクション | NGS | リファレンス配列取得後の各種情報抽出2(readFASTA関数の利用)(last modified 2011/04/07)
- イントロダクション | NGS | アノテーション情報取得(refFlatファイル)(last modified 2010/12/07)
- イントロダクション | NGS | アノテーション情報取得(BioMart and biomaRt)(last modified 2011/08/26)
- イントロダクション | 一般 | 配列取得(last modified 2010/7/7)
- イントロダクション | 一般 | 指定した範囲の配列を取得(last modified 2012/01/05)
- イントロダクション | 一般 | 翻訳配列(translate)を取得(last modified 2011/07/27)
- イントロダクション | 一般 | 相補鎖(complement)を取得(last modified 2011/07/27)
- イントロダクション | 一般 | 逆相補鎖(reverse complement)を取得(last modified 2011/07/27)
- イントロダクション | 一般 | 逆鎖(reverse)を取得(last modified 2011/07/27)
- イントロダクション | 一般 | 二連続塩基の出現頻度情報を取得(last modified 2011/07/25)
- イントロダクション | 一般 | 三連続塩基の出現頻度情報を取得(last modified 2011/07/25)
- イントロダクション | 一般 | multi-fastaファイルから指定した配列長のもののみ抽出(last modified 2012/02/06) NEW
- イントロダクション | NGS | NGSデータ取得(last modified 2011/07/19)

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

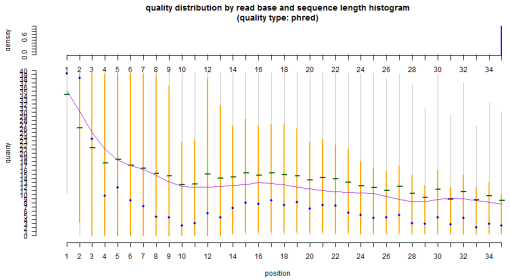
↓

マッピング

↓

データ解析

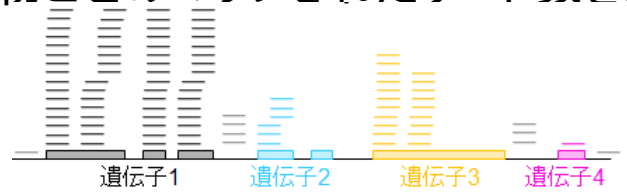
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

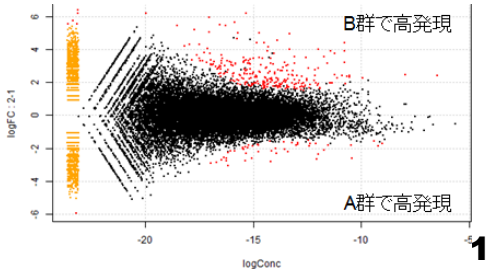
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



データのクオリティチェック

- ファイルの読み込み | マップ前 | Illuminaの*_qseq.txt (last modified 2011/11/28)
- ファイルの読み込み | マップ後 | BAM形式 (last modified 2010/7/13)
- ファイルの読み込み | マップ後 | GFF3形式 (last modified 2010/6/9)
- ファイルの読み込み | マップ後 | SOAP形式 (last modified 2011/07/20)
- クオリティチェック | NGS(一般) |grqc (Quick Read Quality Control) (last modified 2011/12/15)
- フィルタリング | 一般 | ACGTのみからなる配列(重複あり) (last modified 2010/7/6)
- フィルタリング | 一般 | ACGTのみからなる配列(重複なし) (last modified 2010/7/6)
- フィルタリング | NGS(miRNA) | アダプター配列除去0(基本的なところ) (last modified 2010/5/27)

• クオリティチェック | NGS(一般) | qrqc (Quick Read Quality Control)

FastQCのR版のようなものです。Sanger FASTQ形式ファイルを読み込んで、positionごとの「クオリティスコア(quality score)」、「どんな塩基が使われているのか(base frequency and base proportion)」、「リード長の分布」、「GC含量」、「htmlレポート」などを出力してくれます。
ここではSRR037439.fastqファイルに対して解析を行う例を示します。
下記を実行すると「SRR037439-report」という名前のフォルダが作成されます。中にあるreport.htmlをダブルクリックするとhtmlレポートを見ることができます。

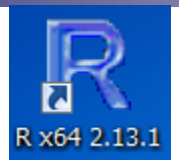
「ファイル」－「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し以下をコピペ。

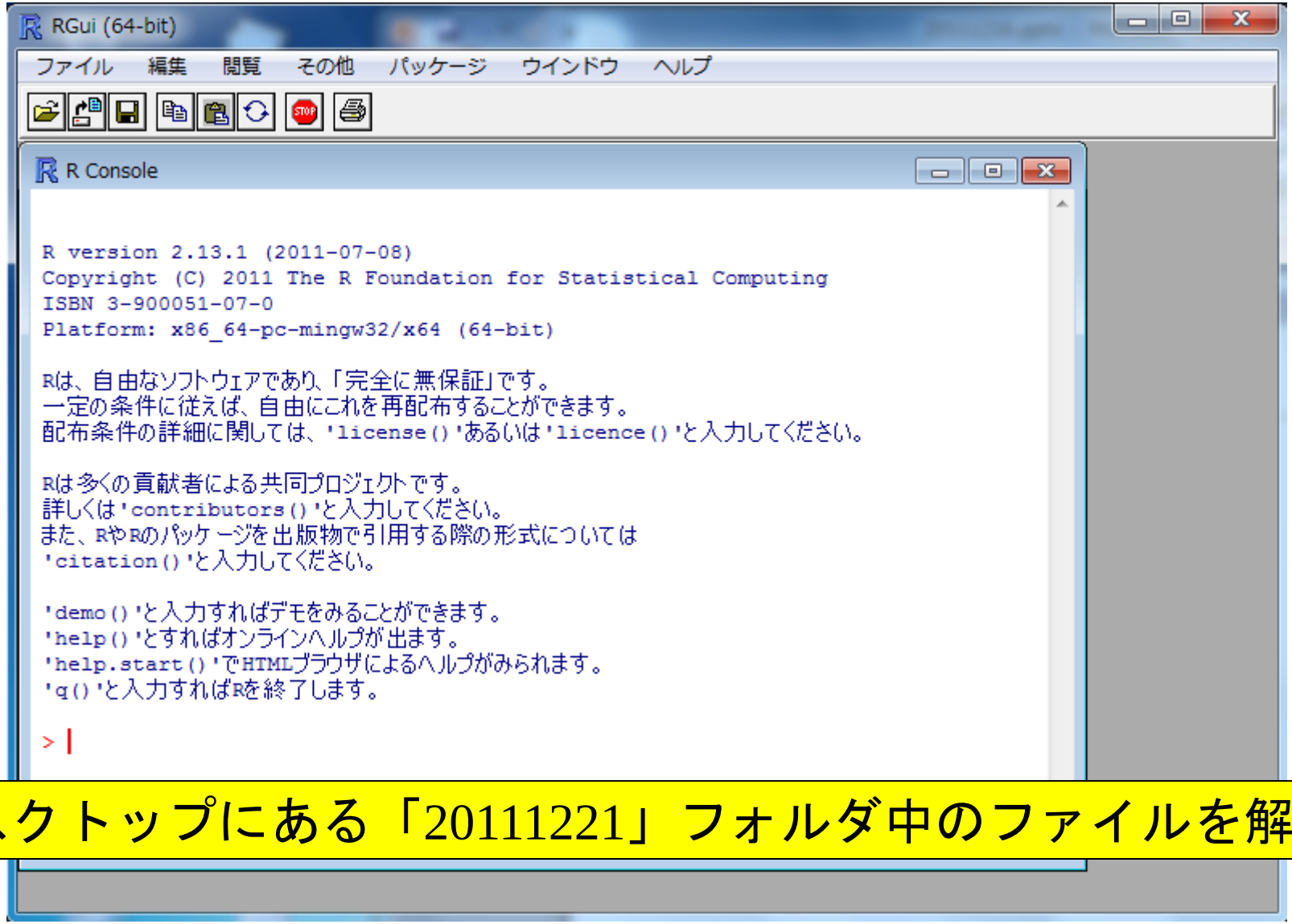
```
------    ここから    ------
in_f <- "SRR037439.fastq"                          #読み込みたいFASTA形式のファイル名を指定してin_fに格納

library(qrqc)                                      #パッケージの読み込み
reads <- readSeqFile(in_f, quality="phred")        #FASTQ形式ファイルの読み込み (Sanger FASTQ形式の場合は"phred", Ill
makeReport(reads)                                  #htmlレポートの作成

------    ここまで    ------
```

Biocon

デスクトップにある「20111221」フォルダ中に
「SRR037439.fastq」というファイルが存在する、という前提

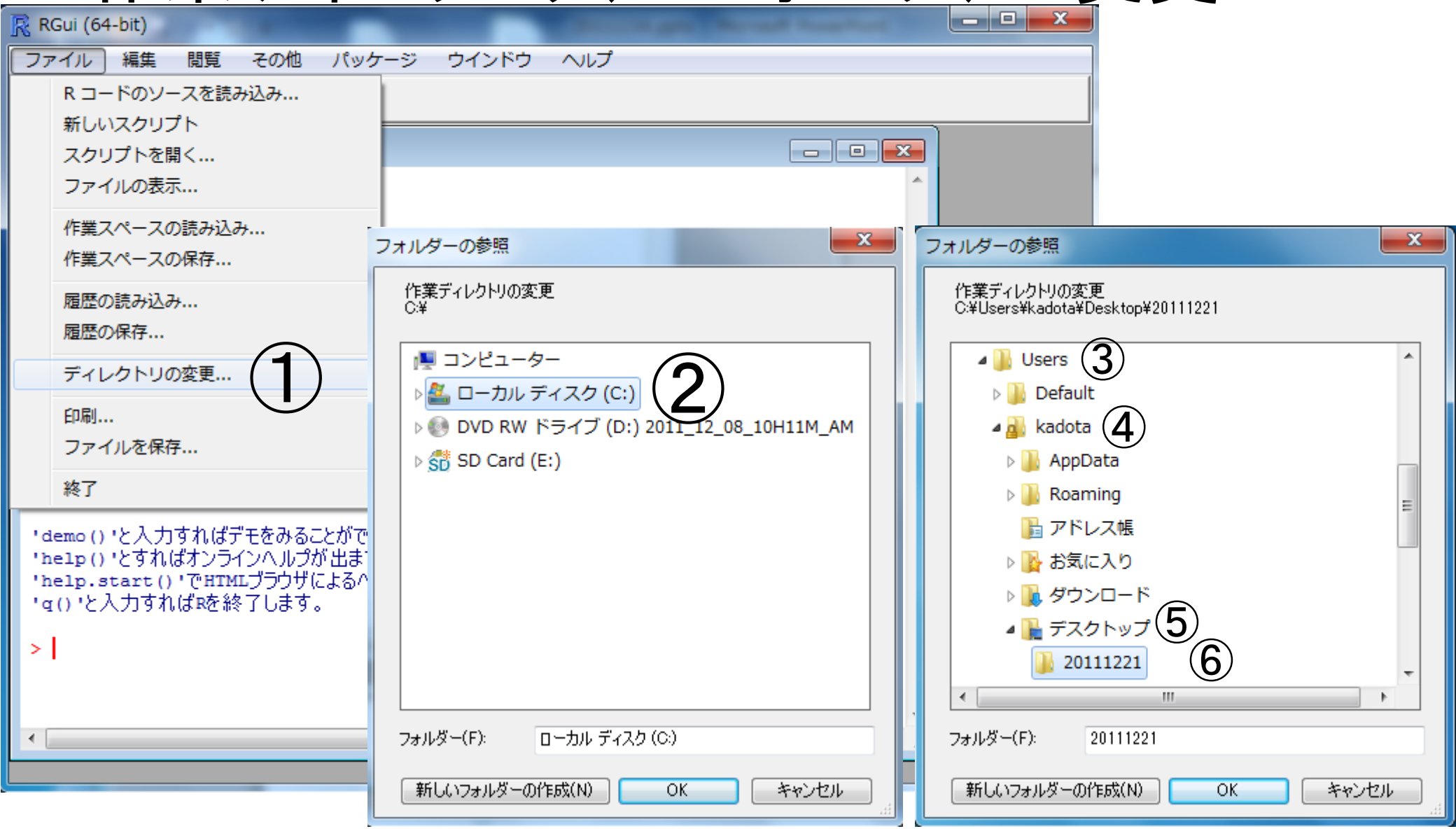
Rの起動



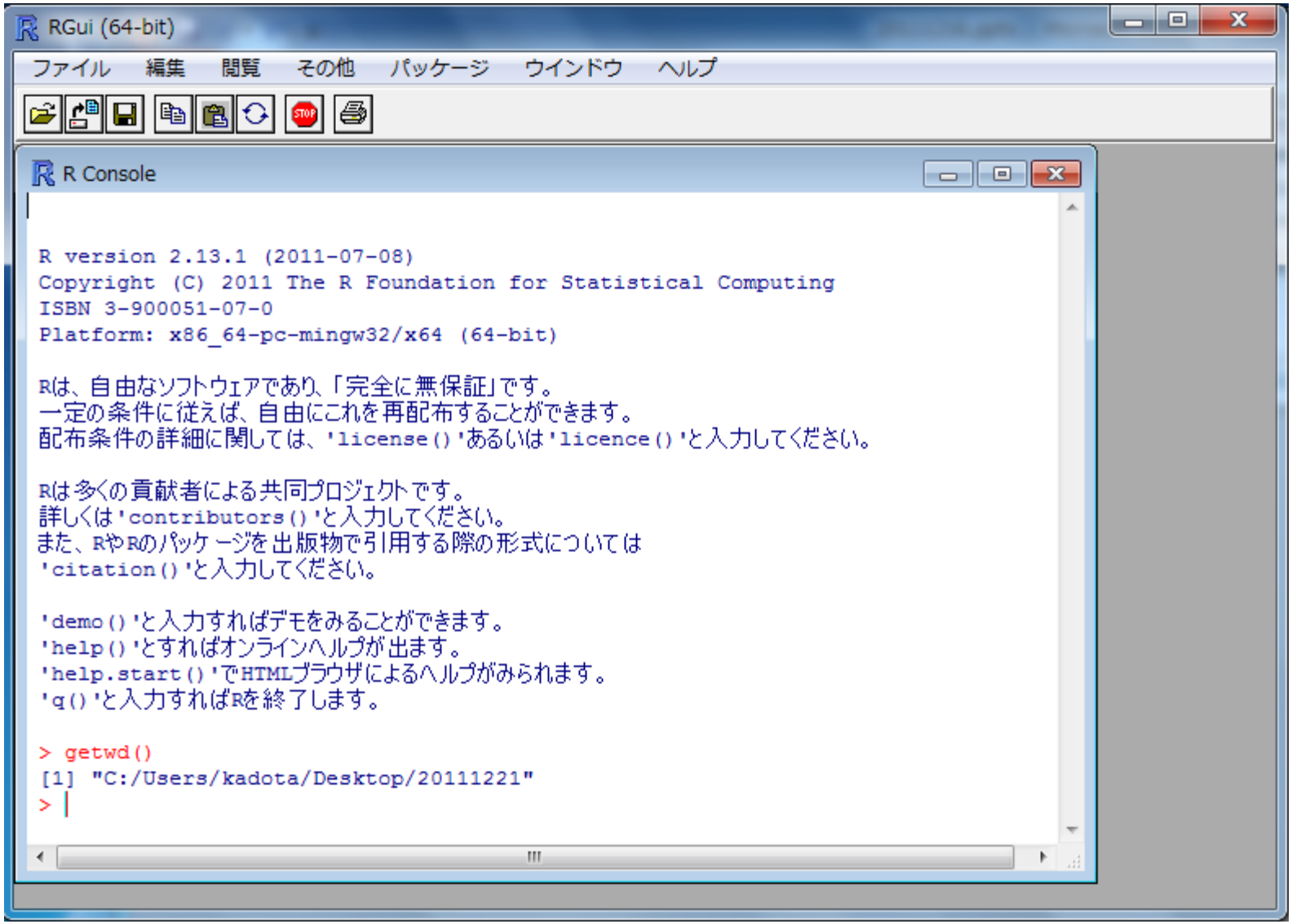


デスクトップにある「20111221」フォルダ中のファイルを解析

作業ディレクトリ(＝フォルダ)の変更



「getwd()」と打ち込んで確認



コピー＆ペースト （こぴぺ）

- クオリティチェック | NGS(一般) | qrqc (Quick Read Quality Control)

FastQCのR版のようなものです。Sanger FAST
われているのか(base frequency and base prop
ここではSRR037439.fastqファイルに対して解析
下記を実行すると「SRR037439-report」という名
ができます。

「ファイル」－「ディレクトリの変更」で解析したい

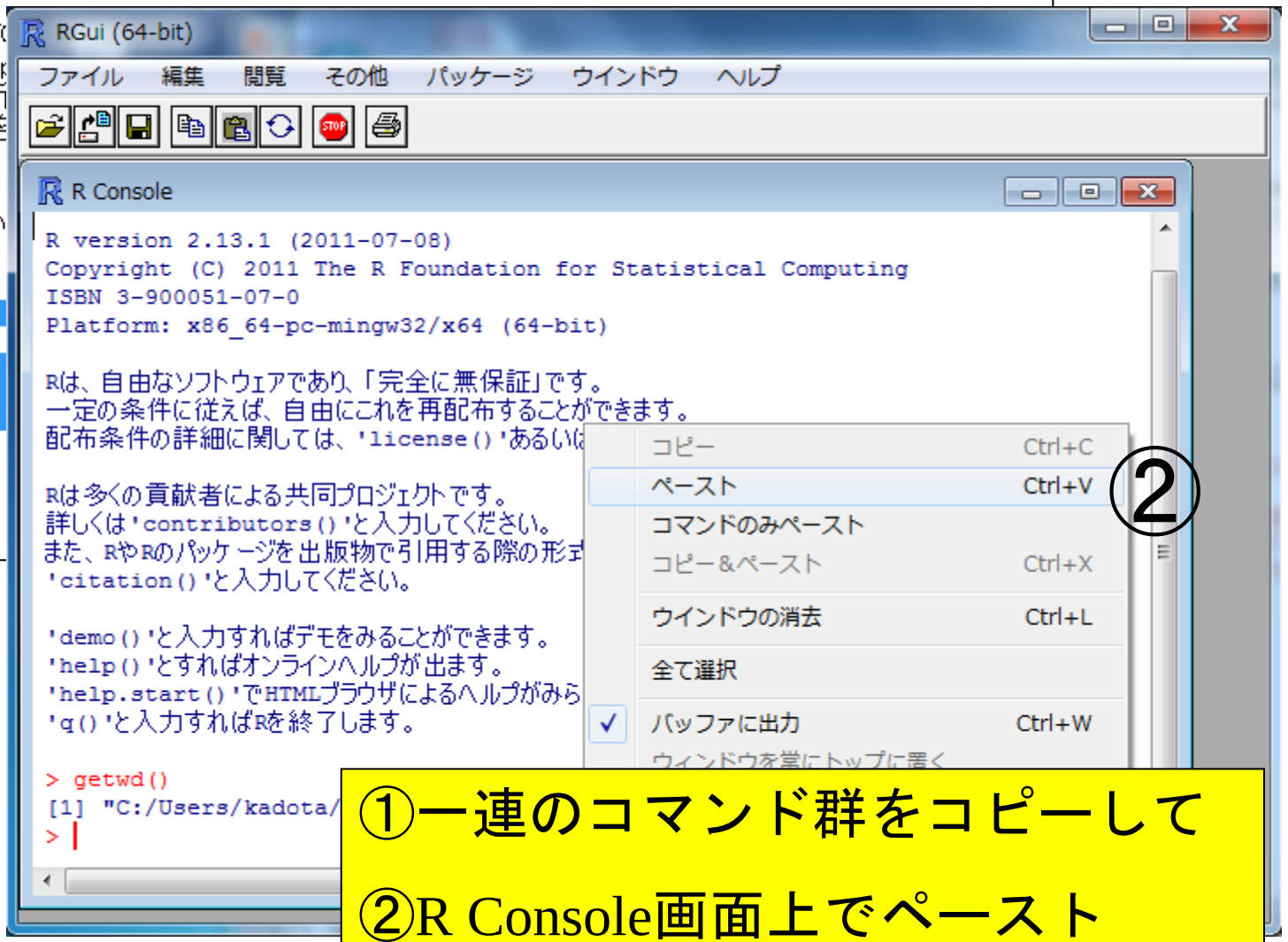
------　ここから　------

```
in_f <- "SRR037439.fastq"

library(qrqc)
reads <- readSeqFile(in_f, quality="phred")
makeReport(reads)
```

------　ここまで　------

BioconductorのqrqcのwebページBioconductorのqrqcのwebページ

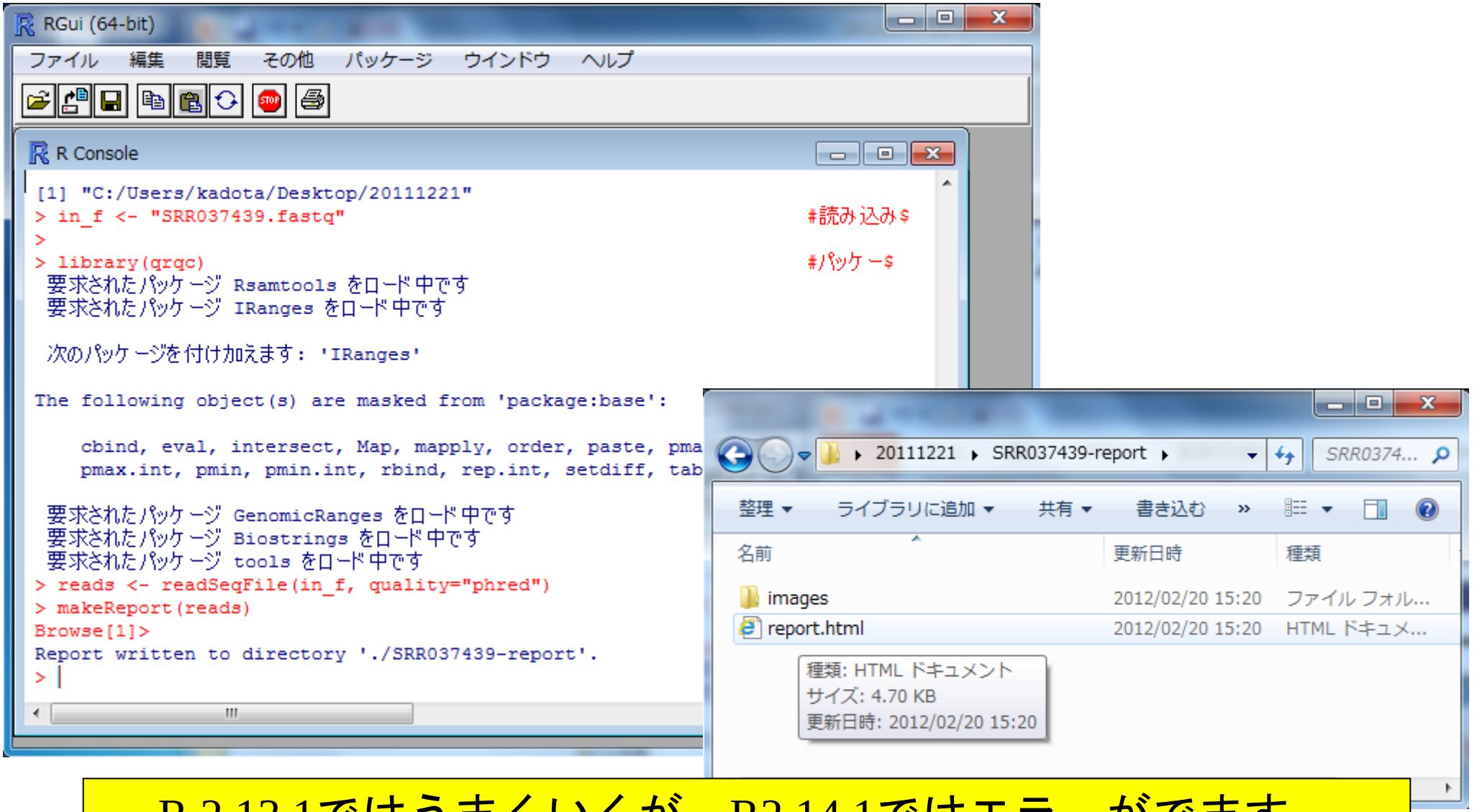
①



①一連のコマンド群をコピーして

②R Console画面上でペースト

htmlレポートが作成される



R 2.13.1ではうまくいくが、R2.14.1ではエラーがでます…

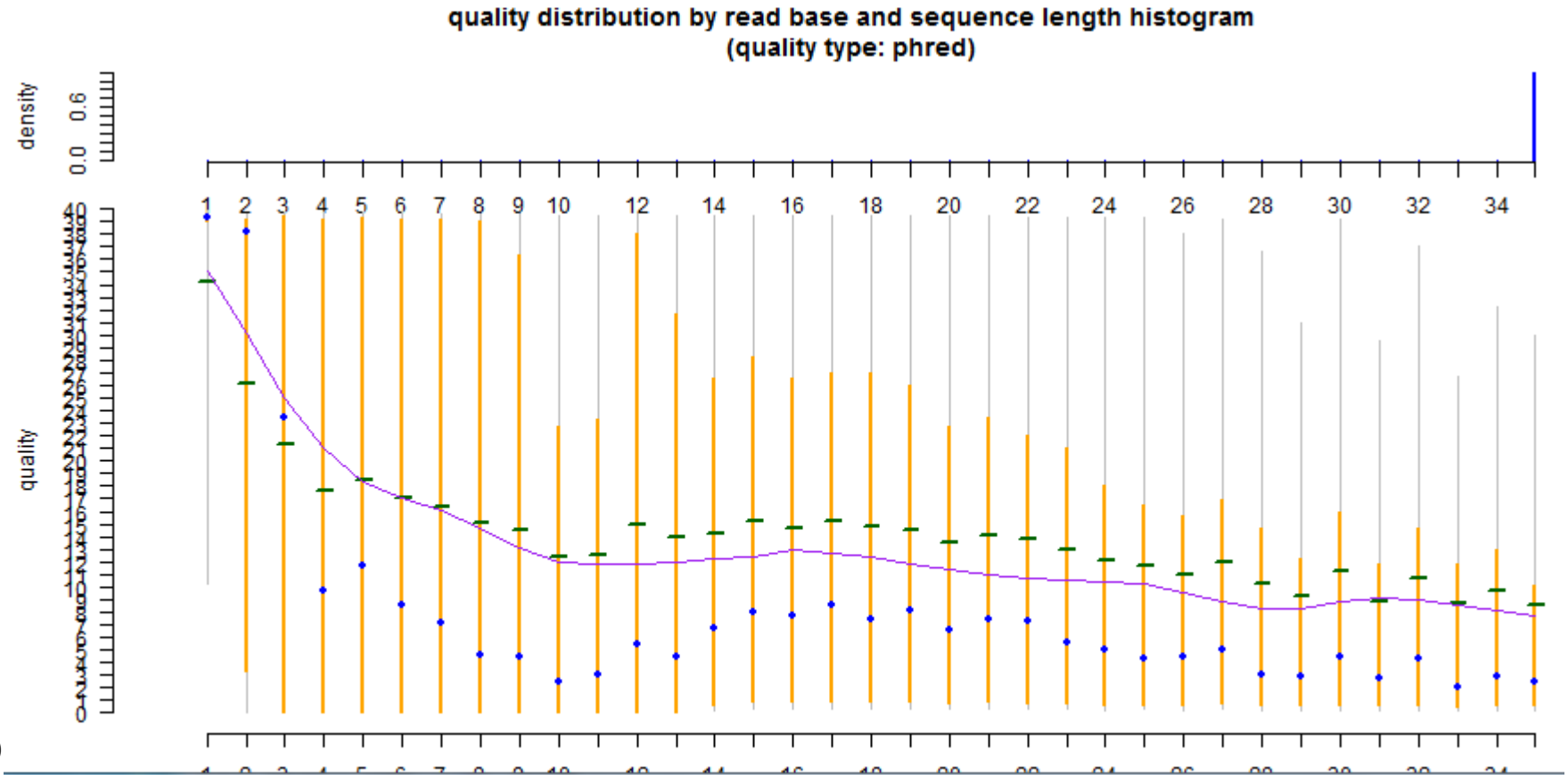
h ...)

General Information

File: SRR037439.fastq
Type: FASTQ
Sequence Length Range: 35 to 35
Total Sequences: 500
Unique Sequences: 498

ポジションごとのクオリティスコアなどの情報が得られます

Quality by Position



FASTQ形式（とFASTA形式）

- FASTA形式
  - 「“>”ではじまる一行のdescription行」と「配列情報」からなる形式
  - NGSのread長は短いので、実質的に一つのリードを二行で表現

```
>SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
```

- FASTQ形式
  - 一行目：「“@”ではじまる一行のdescription行」
  - 二行目：「配列情報」
  - 三行目：「”+”からはじまる一行（のdescription行）」
  - 四行目：「クオリティ情報」

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*(((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

http://en.wikipedia.org/wiki/FASTQ_format

塩基配列のクオリティ情報といえば…

- Phredスコア
  - Phredというベースコールプログラムから得られるQuality Value（QV値）のこと

http://en.wikipedia.org/wiki/Phred_quality_score

なぜFASTQ形式では、Phredスコアそのものでクオリティ情報を表現しないの？

理由:(容量)節約のため

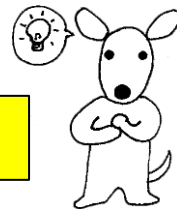
Phred スコア	ASCII印字可能文字		Phred スコア	ASCII印字可能文字	
0	!	ASCII 33	21	6	ASCII 54
1	"	ASCII 34	22	7	ASCII 55
2	#	ASCII 35	23	8	ASCII 56
3	$	ASCII 36	24	9	ASCII 57
4	%	ASCII 37	25	:	ASCII 58
5	&	ASCII 38	26	;	ASCII 59
6	'	ASCII 39	27	<	ASCII 60
7	(	ASCII 40	28	=	ASCII 61
8	)	ASCII 41	29	>	ASCII 62
9	*	ASCII 42	30	?	ASCII 63
10	+	ASCII 43	31	@	ASCII 64
11	,	ASCII 44	32	A	ASCII 65
12	-	ASCII 45	33	B	ASCII 66
13	.	ASCII 46	34	C	ASCII 67
14	/	ASCII 47	35	D	ASCII 68
15	0	ASCII 48	36	E	ASCII 69
16	1	ASCII 49	37	F	ASCII 70
17	2	ASCII 50	38	G	ASCII 71
18	3	ASCII 51	39	H	ASCII 72
19	4	ASCII 52	40	I	ASCII 73
20	5	ASCII 53	…	…	…

- FASTQ形式中のクオリティ情報部分

```
@SRR037439.375
GCGGTGTGTTTGTGGTATAGTGGTGCCCCGCCCCG
+SRR037439.375
IIII&IIIII?223<(<I2B*4@#/I"#"#''""+
```

- Phredスコア(QUAL形式)

```
40 40 40 40  5 40 40 40 40 40 30 17 17 18 27  7 27 40 17 33
 9 19 31  2 14 40  1  2  1  2  6  6  1  1 10
```



Phredスコアが*X*の場合「ASCII (*X*+33)」に対応する文字コードを割り当てる

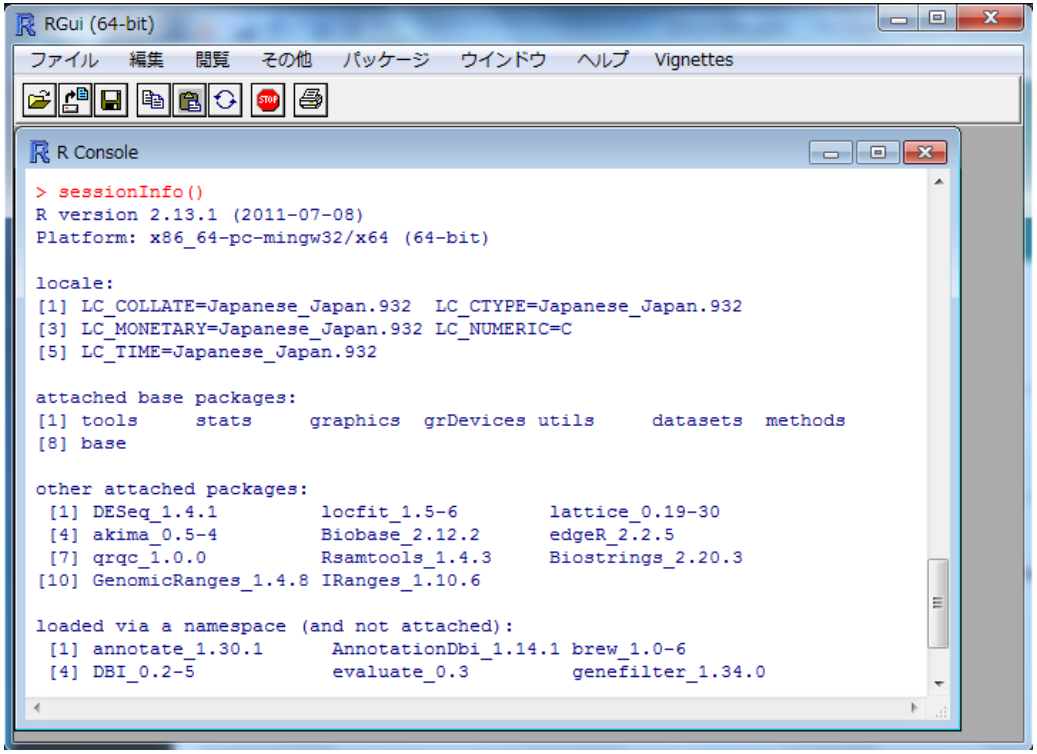
Rの現実…

- 「(Rで)塩基配列解析」のウェブページは常に最新の情報が記載されているわけではない
  - 昔のバージョンだとうまくいくが、最新版だとエラーがでる、こともある
  - Rのバージョンを上げてから昔作ったスクリプトを実行しようとすると、「そんな関数ない」と文句を言われた。よく調べてみると関数名が変わっていた…
    - 例:DESeqパッケージ中のestimateVarianceFunctions → estimateDispersions
  - DEGseqのスクリプトがうまく動かなくなっている…

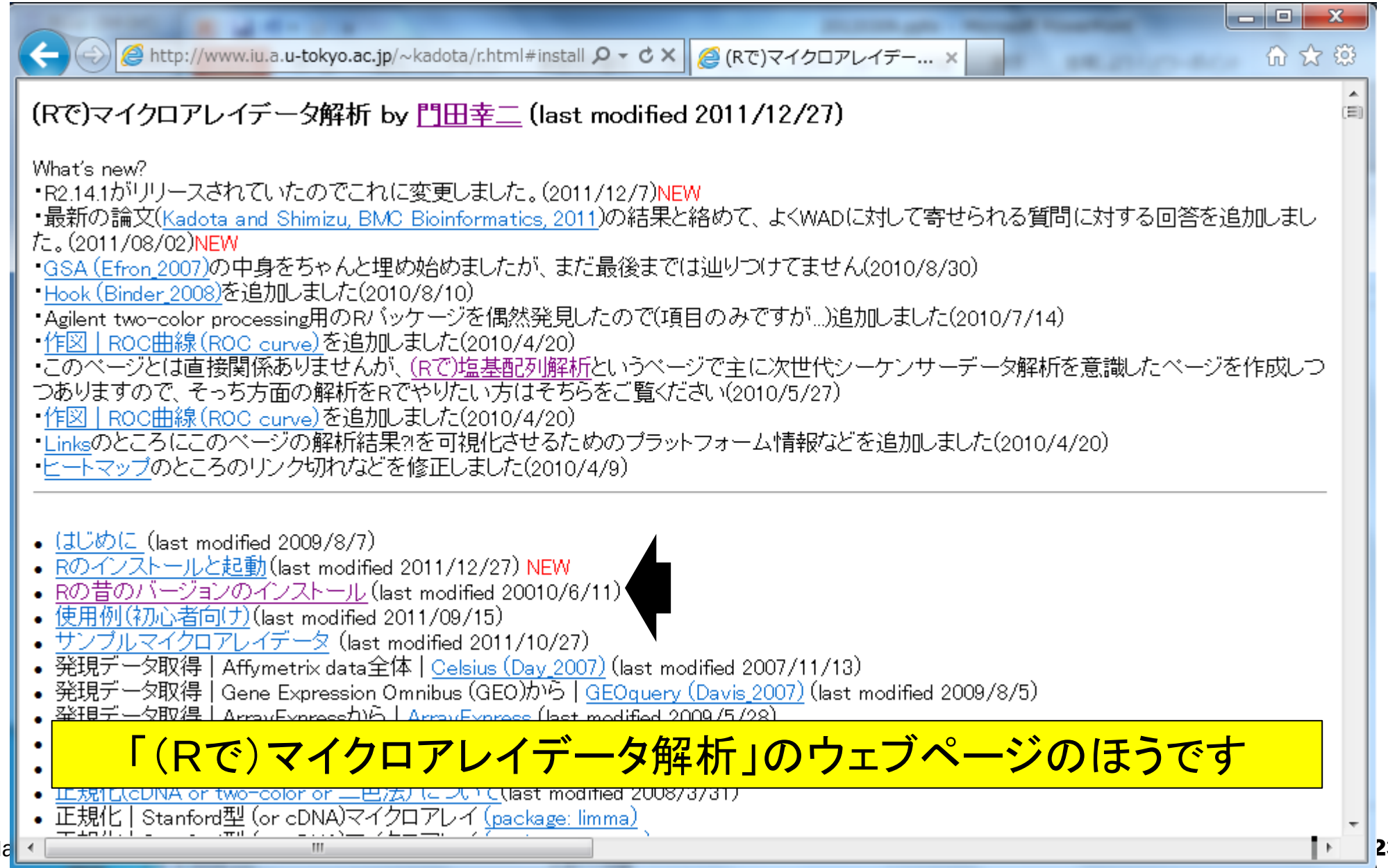
R 2.13.1 (2011-07-08) → R 2.14.1 (2011-12-22)での体験談

Rの現実…と対処法

- ぐぐる（「qrqc error」などで検索）
- Rのバージョンを最新版（または古いもの）に変更
  - 「sessionInfo()」で現在利用しているRのバージョンを確認
- 手元にある印刷物のマニュアルは古いものでは?!



Rの昔のバージョンのインストール



比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

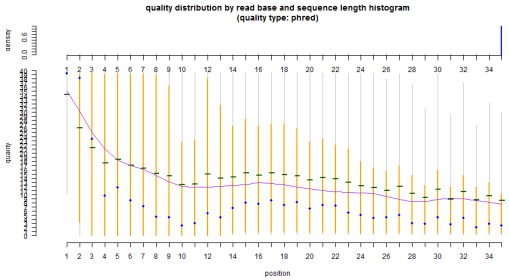
↓

マッピング

↓

データ解析

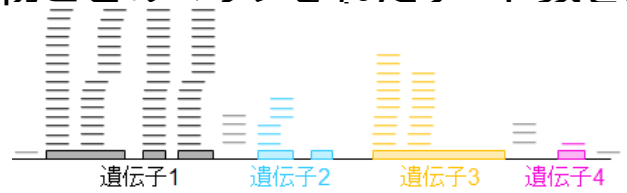
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

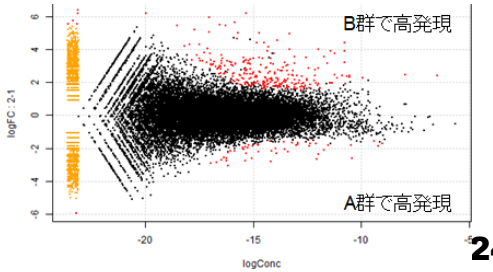
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



リファレンス配列について

http://hgdownload.cse.ucsc.edu/downloads.html

- Q:マッピングに使うリファレンス配列は？
  - A:好きなものを使ってください。ゲノム配列でもトランスクリプトーム配列でも結構です。
- Q:どこから取得できるんですか？
  - A:「UCSC Sequence and Annotation Downloads」などから取得できます（アノテーション情報も）。以下はほんの一例
    - ヒト全ゲノム配列の場合
      ftp://genome-ftp.cse.ucsc.edu/goldenPath/hg19/bigZips/hg19.2bit
    - ヒトトランスクリプトーム配列（RefSeq mRNA）の場合
      ftp://genome-ftp.cse.ucsc.edu/goldenPath/hg19/bigZips/refMrna.fa.gz
    - ヒトアノテーション情報の場合
      http://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/refFlat.txt.gz

こんな感じ

ヒトゲノム配列

```
[kadota@hpcs02 hg19]$ head hg19.fasta
>chr1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
[kadota@hpcs02 hg19]$
```

- 1-22番染色体＋X＋Y
- 約6200万行のファイル
- 約3GBのサイズ

ヒトトランスクリプトーム配列

```
[kadota@hpcs02 h_rna]$ head h_rna.fasta
>NM_203348.1
CAGCCCAGGGCCGGGCTGCCCACCGTGCCAGCCCAGTTTCAATGACCCACCTGAGGGTTT
CCATCGTGGGCCAGGGGACCGGCGCAGGCGGGCATCCGGAGCCAGGCAGTGGCCAGCCCA
TCCCGGGCAGGGTCATAGGTGGGGCTGCAGGCCTCGAAGCCGCCATGTGAAGGCAGCAGC
GACCCCAGGCAGGGCGGGCCAGGTTGACCTTGCACCTGCTCTCCCCTCAGCCCCGCGGCC
ATGCCGACCCTGGTCGTGGGCACGCCGCCCACCTGCCTGGGGGACACACCTCAGCCCTGC
CACAAGAACAGCCAGAGGCAGGGCCCCTTCTCCCATGGGGCCCCAGGGAGAGCAGCCGAC
TGGAAGGCTGTTGCCAAGCCCAGGCTCTGCGCACCTGCAGCTGAGGATGACGTGGCAGCC
CTGAGGTGGCCCGGGCCCTCCCAGCAGCCAGACCCACCCTGGGCAGCTCCCCACGTGGTC
GGGTCTGACGACCTCAAGGAACCAGGCCCCTGGGGGAAGGCGTGCAGCCTGCCCATGTGG
[kadota@hpcs02 h_rna]$
```

- 46,XXX mRNA sequences

ftp://ftp.ncbi.nih.gov/refseq/H_sapiens/mRNA_Prot/human.rna.fna.gz

アノテーション情報ファイル（refFlat形式）

symbol（A）、name（B）、染色体名（C）、strand（D）、転写開始位置（E）、転写終結位置（F）、CDS start（G）、CDS end（H）、Exon数（I）

	A	B	C	D	E	F	G	H	I	J	K
1	SNAR-G2	NR_024244	chr19	−	49534925	49535044	49535044	49535044	1	49534925,	49535044,
2	SNAR-D	NR_024243	chr19	−	50643458	50643577	50643577	50643577	1	50643458,	50643577,
3	SNORD113-5	NR_003233	chr14	+	101404523	101404600	101404600	101404600	1	101404523,	101404600,
4	UBL5	NM_024292	chr19	+	9938567	9940797	9939013	9940684	5	9938567,9939002,9939267,9939512,9940640,	9938740,9939069,9

非モデル生物の場合

- 手持ちのRNA-Seqデータのみ
  - 2010年頃から提供されはじめた*de novo* transcriptome assembly用のプログラム（TrinityやTrans-ABySSなど;もちろんLinux用です）を利用すればトランスクリプトームの配列セット（RefSeqのようなイメージ）を得ることができます。

入力：RNA-Seqデータ

```
>read1
GGGGTTCAAAGCAGTATCGATCAAATAGTA
>read2
GTTCAAAGCAGTATCGATCAAATAGTAAAT
>read3
ACGATGCAGCCTTAACGATGGTCCACAATT
>read4
```

出力：コンティグ（≒転写物配列）

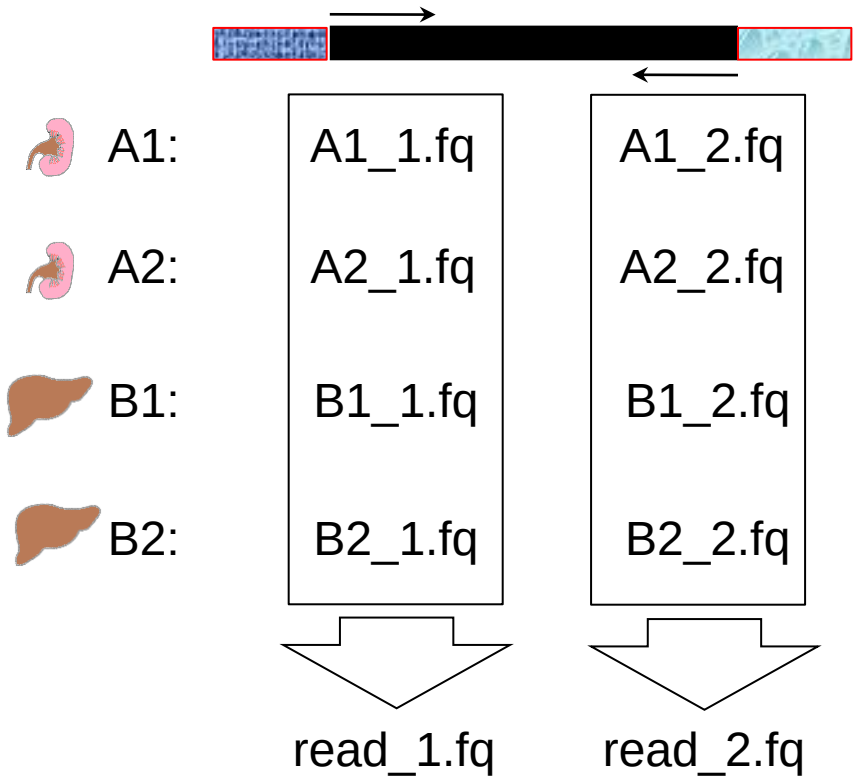
```
>contig1(transcript1)
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAA
CTCACAGTTTGGAGCTTATCAGTCAA…
>contig2(transcript2)
ACGATGCAGCCTTAACGATGGTCCACAATTATCGGGAATCA…
>contig3(transcript3)
…
```

非モデル生物の場合

- 手持ちのRNA-Seqデータのみでアセンブルを行う場合は、paired-endのデータが基本

・ペアードエンド法
断片配列の両末端が数百塩基以内の対の二種類の配列が得られる

・シングルエンド法



二つのファイルを入力として比較対象サンプルを混合したデータのアセンブル
→リファレンストランスクリプトーム配列の取得

非モデル生物の場合

- Trinity (Grabherr et al., *Nat. Biotechnol.*, 2011)実行プログラムの一例

```
nohup Trinity.pl --seqType fq --left read_1.fq --right read_2.fq --
run_butterfly --bflyHeapSpace 180G --CPU 2 --bfly_opts "-V 10 --
stderr" --run_ALLPATHSLG_error_correction --output hoge &
```

アセンブルが終了すると、**hoge**というディレクトリ中にTrinity.fastaというmulti-fastaファイルが得られる

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

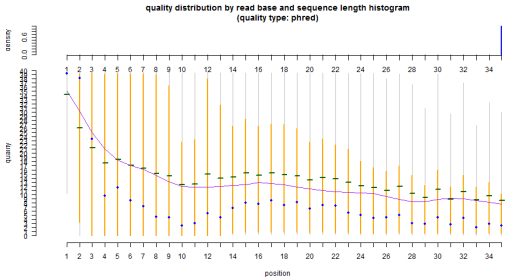
↓

マッピング

↓

データ解析

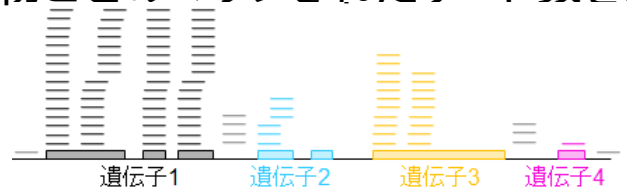
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

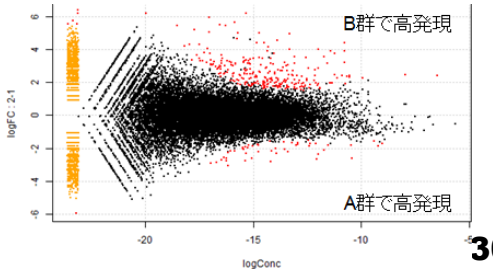
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



multi-fasta形式ファイルからの情報抽出1

• イントロダクション | NGS | アセンブル後のmulti-fastaファイルからN50などの基本情報を取得

Trinityなどのアセンブルプログラムを実行したあとのファイルからを想定して、Total lengthやaverage lengthなど
方を示します。ここでは130MB程度のmult
「ファイル」－「ディレクトリの変更」でファイ

```
------    ここから    ------
in_f <- "h_rna.fasta"
out_f <- "hoge.txt"

library(Biostrings)
reads <- read.DNAStringSet(in_f, format=

Total_length <- sum(width(reads))
Number_of_contigs <- length(reads)
Average_length <- mean(width(reads))
Median_length <- median(width(reads))
Max_length <- max(width(reads))
Min_length <- min(width(reads))

#N50計算のところ
sorted_length <- rev(sort(width(reads)))
N50 <- sorted_length[cumsum(sorted_lengt

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
GC_content <- sum(CG)/sum(ACGT)

#出力用に結果をまとめている
tmp <- NULL
tmp <- rbind(tmp, c("Total length (bp)",
tmp <- rbind(tmp, c("Number of contigs",
tmp <- rbind(tmp, c("Average length", Av
tmp <- rbind(tmp, c("Median length", Mec
tmp <- rbind(tmp, c("Max length", Max_le
tmp <- rbind(tmp, c("Min length", Min_le
tmp <- rbind(tmp, c("N50", N50))
tmp <- rbind(tmp, c("GC content", GC_cor
write.table(tmp, out_f, sep="¥t", appenc

------    ここまで    ------
```

①

RGui (64-bit)

ファイル　編集　閲覧　その他　パッケージ　ウインドウ　ヘルプ

R Console

```
R version 2.13.1 (2011-07-08)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-mingw32/x64 (64-bit)

Rは、自由なソフトウェアであり、「完全に無保証
一定の条件に従えば、自由にこれを再配布す
配布条件の詳細に関しては、'license()'

Rは多くの貢献者による共同プロジェクトです。
詳しくは'contributors()'と入力してくださ
また、RやRのパッケージを出版物で引用する際
'citation()'と入力してください。

'demo()'と入力すればデモをみることができま
'help()'とすればオンラインヘルプが出ます。
'help.start()'でHTMLブラウザによるヘルプ
'q()'と入力すればRを終了します。

> getwd()
[1] "C:/Users/kadota/Desktop/
>
```

②

コピー　Ctrl+C
ペースト　Ctrl+V
コマンドのみペースト
コピー&ペースト　Ctrl+X
ウインドウの消去　Ctrl+L
全て選択
✓ バッファに出力　Ctrl+W
ウィンドウを常にトップに置く

①一連のコマンド群をコピーして

②R Console画面上でペースト

multi-fasta形式ファイルからの情報抽出1

RGui (64-bit)

ファイル 編集 閲覧 その他 パッケージ ウインドウ ヘルプ

R Console

```
> #N50計算のところ
> sorted_length <- rev(sort(width(reads)))                    #長さ
> N50 <- sorted_length[cumsum(sorted_length) >= Total_length/2][1] #「N5
>
> #GC含量(GC content)計算のところ
> count <- alphabetFrequency(reads)                           #A,C,
> CG <- rowSums(count[,2:3])                                  #C,G
> ACGT <- rowSums(count[,1:4])                                #A,C,
> GC_content <- sum(CG)/sum(ACGT)                             #ト
>
> #出力用に結果をまとめている
> tmp <- NULL
> tmp <- rbind(tmp, c("Total length (bp)", Total_length))
> tmp <- rbind(tmp, c("Number of contigs", Number_of_contigs))
> tmp <- rbind(tmp, c("Average length", Average_length))
> tmp <- rbind(tmp, c("Median length", Median_length))
> tmp <- rbind(tmp, c("Max length", Max_length))
> tmp <- rbind(tmp, c("Min length", Min_length))
> tmp <- rbind(tmp, c("N50", N50))
> tmp <- rbind(tmp, c("GC content", GC_content))
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中$
>
>
```

hoge.txt - Microsoft Excel

D11

	A	B
1	V1	V2
2	Total length (bp)	123433471.0
3	Number of contigs	46261.0
4	Average length	2668.2
5	Median length	2128.0
6	Max length	101516.0
7	Min length	33.0
8	N50	3688.0
9	GC content	0.48732425

hoge

コマンド 100%

出力ファイル名として指定したもの（hoge.txt）が「20111221」フォルダ中に作成される（はず）

練習

- 「20111221」中にある**practice1.txt**中の記述を変更して、**Trinity.fasta**ファイルに対して同様の解析を行い、結果を**hoge1.txt**に出力せよ

practice1.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "h_rna.fasta"
out_f <- "hoge.txt"

library(Biostrings)
reads <- read.DNAStringSet(in_f, format="fasta")

Total_length <- sum(width(reads))
Number_of_contigs <- length(reads)
Average_length <- mean(width(reads))
Median_length <- median(width(reads))
Max_length <- max(width(reads))
Min_length <- min(width(reads))

#N50計算のところ
sorted_length <- rev(sort(width(reads)))
N50 <- sorted_length[cumsum(sorted_length) >= Total_length/

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
GC_content <- sum(CG)/sum(ACGT)
```

practice1.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "Trinity.fasta"
out_f <- "hoge1.txt"

library(Biostrings)
reads <- read.DNAStringSet(in_f, format="fasta")

Total_length <- sum(widt
Number_of_contigs <- le
Average_length <- mean(
Median_length <- median
Max_length <- max(width
Min_length <- min(width

#N50計算のところ
sorted_length <- rev(so
N50 <- sorted_length[cu

#GC含量(GC content)計算
count <- alphabetFreque
CG <- rowSums(count[,2:
ACGT <- rowSums(count[,
GC_content <- sum(CG)/su
```

	A	B
1	V1	V2
2	Total length (bp)	2993
3	Number of contigs	4
4	Average length	748.3
5	Median length	784
6	Max length	888
7	Min length	537
8	N50	886
9	GC content	0.524

multi-fasta形式ファイルからの情報抽出2

• 解析 | 一般 | GC含量 (GC contents)

GC含量 (GC contents)の計算の仕方を書きます。ここでは、ファイルの読み込み(FASTA形式)で読み込んだ250 readsからなるtest1.fastaファイルを入力として、250 readsの各配列に対して「description」「C,Gの総数」「A,C,G,Tの総数」「配列長」「%GC含量」をファイルに出力するやり方を例示します。
尚、ここでは%GC含量の計算を「CGの総数/ACGTの総数」で計算していますので、もしGC含量を計算したい配列中にNなどが含まれる場合でNなどを含めた「配列長」を分母にしたい場合にはGC含量を計算をする数式中の「CG/ACGT*100」を「CG/width(reads)*100」に変更してください。
「ファイル」ー「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。

```
------    ここから    ------
in_f <- "test1.fasta"                                   #読み込みたいFASTA形式のファイル名を指定してin_fに格納
out_f <- "hoge.txt"                                     #出力ファイル名を指定
library(Biostrings)                                     #パッケージの読み込み
reads <- read.DNAStringSet(in_f, format="fasta")        #in_fで指定したファイルの読み込み
count <- alphabetFrequency(reads)                       #A,C,G,T,..の数を各配列ごとにカウントした結果をcountに格納
CG <- rowSums(count[,2:3])                              #C,Gの総数を計算してCGに格納
ACGT <- rowSums(count[,1:4])                            #A,C,G,Tの総数を計算してACGTに格納
out <- CG/ACGT*100                                      #%GC含量を計算してoutに格納

tmp <- cbind(names(reads), CG, ACGT, width(reads), out)          #ファイルに出力したい情報を連結してtmpに格納
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報を与えている
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

BioconductorのBiostringsのwebページ

配列ごとのGC含量を計算したいとき

練習

- 「20111221」中にある**practice2.txt**中の記述を変更して、**Trinity.fasta**ファイルに対して同様の解析を行い、結果を**hoge2.txt**に出力せよ

practice2.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "test1.fasta"                                    #読み込みたいFASTA形
out_f <- "hoge.txt"                                      #出力ファイル名を指
library(Biostrings)
reads <- read.DNAStringSet(in_f, format="fasta")
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

tmp <- cbind(names(reads), CG, ACGT, width(reads
colnames(tmp) <- c("description", "CG", "ACGT",
write.table(tmp, out_f, sep="¥t", append=F, quot
```

hoge2.txt - Microsoft Excel

ファイル ホーム 挿入 ページ レイアウト 数式 データ 校閲 表示 活用しよう！エクセル

G7

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	comp59_c0_seq1 le	266	537	537	49.53445065
3	comp371_c0_seq1	577	886	886	65.1241535
4	comp26_c0_seq1 le	289	682	682	42.37536657
5	comp8729_c0_seq1	437	888	888	49.21171171

hoge2

コマンド 100%

multi-fasta形式ファイルからの情報抽出3

• イントロダクション | 一般 | multi-fastaファイルから指定した配列長のもののみ抽出

RefSeqのhuman mRNAのmulti-fasta形式のファイル (h_rna.fasta)が手元にあったとして、任意の配列長(例:>= 200 bp)のもののみ抽出す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。

```
------    ここから    ------
in_f <- "h_rna.fasta"                          #multi-fasta形式のファイルを指定
out_f <- "hoge.fasta"                          #出力ファイル名を指定
param <- 200                                   #配列長の閾値を指定

library(Biostrings)                            #パッケージの読み込み
reads <- read.DNAStringSet(in_f, format="fasta")  #in_fで指定したファイルをFASTA形式で読み込み
reads                                          #今現在のreadsオブジェクトを表示
reads <- reads[width(reads) >= param]          #paramで指定した配列長以上のもののみ抽出してreadsに格納
reads                                          #今現在のreadsオブジェクトを表示
write.XStringSet(reads, file=out_f, format="fasta")  #out_fで指定したファイル名でreadsというオブジェクトをfasta形式で保存
------
```

Bioconduct

practice3.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "h_rna.fasta"                          #multi-fasta形式のファイルを指定
out_f <- "hoge.fasta"                          #出力ファイル名を指定
param <- 200                                   #配列長の閾値を指定

library(Biostrings)                            #パッケージの読み込み
reads <- read.DNAStringSet(in_f, format="fasta")  #in_fで指定したファイルをFASTA形式で読み込み
reads                                          #今現在のreadsオブジェクトを表示
reads <- reads[width(reads) >= param]          #paramで指定した配列長以上のもののみ抽出してreadsに
reads                                          #今現在のreadsオブジェクトを表示
write.XStringSet(reads, file=out_f, format="fasta")  #out_fで指定したファイル名でreadsというオブジェクト
```

Trinity.fastaに対して600bp以上のもののみ抽出してみよう

multi-fasta形式ファイルからの情報抽出4

• **前処理 | Trinity出力ファイルからFPKM値を取得**

2011年10月20日時点で最もお手軽にトランスクリプトーム配列のde novo assembleをしてくれるのはおそらくTrinity(参考文献1)です。アセンブルプログラム(転写物)のところでは、kをいろいろやって...などと書いてますが、Trinityは(trans-ABySSなどと違ってkの値を複数振ってコンティグの和集合を得てから重複を取り除いていくというような作業ではなく)k=25だけでアセンブルを実行しているということもあるのでしょうが、とにかく早いし(こちらの環境で数週間→3日程度)設定ファイルの記述など面倒なことはほとんどありません。
ここでは、手元にTrinityを実行して(どこかでやってもらって)得られたTrinity.fastaファイルがあるという前提で、description部分に記述されているそのコンティグの発現レベル(FPKM値)の情報などを抽出するやり方を示します。
尚、このファイルは基本的なフォーマット部分のみ人工的に作ったものですので、FPKMの記述以外のところは特に気にしないでください。

「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。

```
------    ここから    ------
in_f <- "Trinity.fasta"                                          #読み込みたいファイル名を指定してin_fに格納
out_f <- "output.txt"                                            #出力ファイル名を指定してout_fに格納

library(Biostrings)                                              #パッケージの読み込み
reads <- read.DNAStringSet(in_f, format="fasta")                 #in_fで指定したファイルをFASTA形式で読み込み
hoge <- strsplit(names(reads), " ", fixed=TRUE)                  #names(reads)中の文字列をスペース(" ")で区切った結果をリスト形式
contigID <- unlist(lapply(hoge, "[[", 1))                        #hogeのリスト中の1番目の要素(コンティグID部分に相当)のみ抽出してco
hoge2 <- unlist(lapply(hoge, "[[", 3))                           #hogeのリスト中の3番目の要素(FPKM部分に相当)のみ抽出してhoge2に格納
hoge3 <- strsplit(hoge2, "=", fixed=TRUE)                        #hoge2中の文字列を"="で区切った結果をリスト形式でhoge3に格納
FPKM <- unlist(lapply(hoge3, "[[", 2))                           #hoge3のリスト中の2番目の要素(FPKMの実際の値部分に相当)のみ抽出して
transcript_length <- width(reads)                                #配列長情報をtranscript_lengthに格納
tmp <- cbind(contigID, FPKM, transcript_length)                  #「コンティグID」、「FPKM値」、「配列長」を結合してtmpに格納
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

(参考文献1(Trinityの原著論文; Grabherr et al., Nature Biotechnol, 2011)

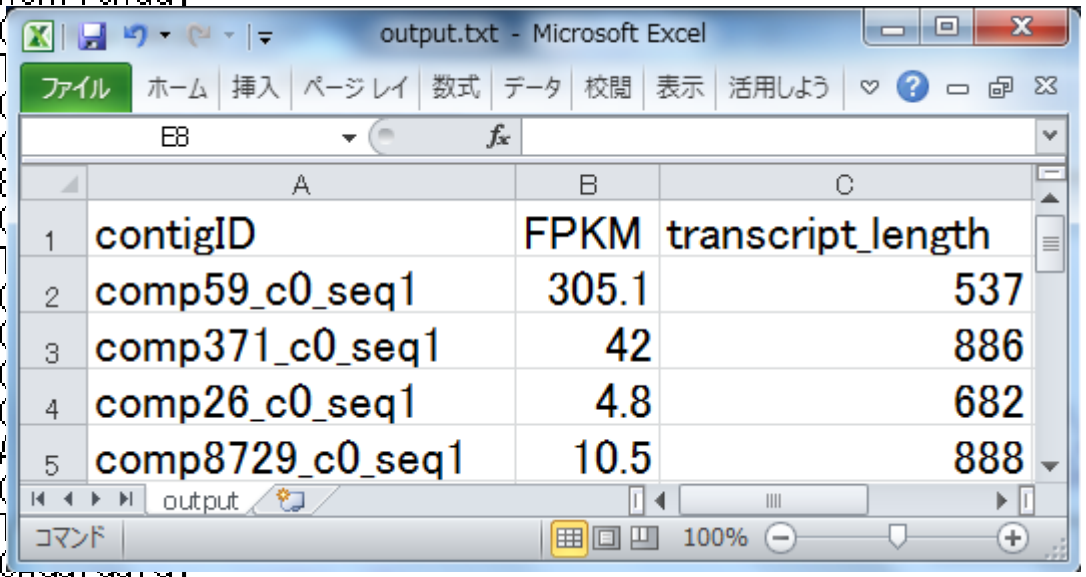
FPKM値：配列長補正ずみの発現量に相当する値

Trinityの最近のバージョンではFPKM値を出力しなくなったようですね...

multi-fasta形式ファイルからの情報抽出4

Trinity.fasta

```
>comp59_c0_seq1 len=537 ~FPKM=305.1 path=[0:0-536]
TCATGCCAAAAGGCAGCAAATAAGTGCCTTTTCTTCCCTTCAGAATACATGGACAATCCA
AAGCTCTATTAGTCTATTATTCAGAATGAAAAGTGTTTACAATATTCGTCCTCTTACTCC
TCAGTATGTGAGACTGTTCCTCGTAGCAGGTAATTTCTTCCGAATTCAAAAACTCCTCAT
GGAAGCATCTGTTTTTGTCATCAAGGAGGGGGCTGTATGTGGAATTGCAAGGCCAAAGAC
ATCTCGGGTCAACTCTTCTCAGGACCAGATCCAAGTCGCCAGTGAGAACACACATTCAGG
CAGCCTCCACCAGCGCCCTGCCTCAGGAGCGAGGCTCCCTGCTTCATGTGG
GCCCTTGTTTTCCCAGCGGGCAGTTGGGGGTCTGGAGCCTAGGAAGCAAAG
CACTCCTGCTTCCTTCCTTCCCTGCAGTTGAGACGGGAGTCTTACTTTGTTC
GGTCTCAAACTCCTGGCTTCAAGCAATCCTTCCACTTTGGCCTTCCAAAGTC
>comp371_c0_seq1 len=886 ~FPKM=42 path=[27:0-88 53:8
GCTTCAGTCCAGCACCTTTCTCGGGTCACGGCCTCCTCCTGGCTCCCAGGAC
AGGCAGAGGCAGGCCTTCCTACACCCTACTCCCTGTGCCTCCAGGCTCGACT
GCACTCGACGACTGAGTCTCTGAGGTCACTTCACCGTGGTCTCCGCCTCACC
TGGACCAGTGAGAGGAGAGGGCTGGGGCGCTCCGCTGAGCCACTCCTGCGCC
CTTGTCTACCTCTTGCCCCCCGAAGGGTTAGTGTCGAGCTCACTCCAGCATC
TCCTGGTGGCCTTGCCGCCCCCACAACCCCGAGGTTTAAAGCCAGGTACACC
GACACACCAAGGATGGAGATGTTCCAGGGGCTGCTGCTGTTGCTGCTGCTGA
GGGACATGGGCATCCAAGGAGCCGCTTCGGCCACGGTGCCGCCCCATCAATC
GCTGTGGAGAAGGAGGGCTGCCCCGTGTGCATCACCGTCAACACCACCATCT
TACTGCCCCACCATGACCCGCGTGCTGCAGGGGGTCCTGCCGGCCCTGCCTC
TAGCATTCACCAAGGATGAACTGAAAGCAGGATCTGTCTCACCATACACTGAGAACTGTA
```



	A	B	C
1	contigID	FPKM	transcript_length
2	comp59_c0_seq1	305.1	537
3	comp371_c0_seq1	42	886
4	comp26_c0_seq1	4.8	682
5	comp8729_c0_seq1	10.5	888

FPKM値をもとにサンプル内の転写物間の発現レベルの大小を議論可能

サンプル間の比較には使えない（といわれている）

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

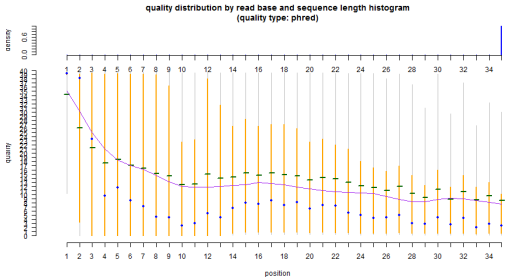
↓

マッピング

↓

データ解析

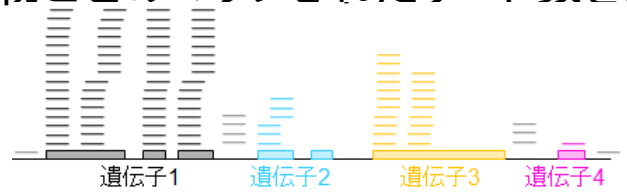
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

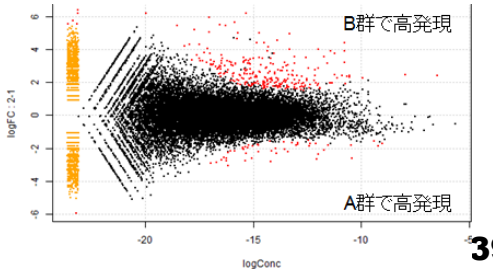
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



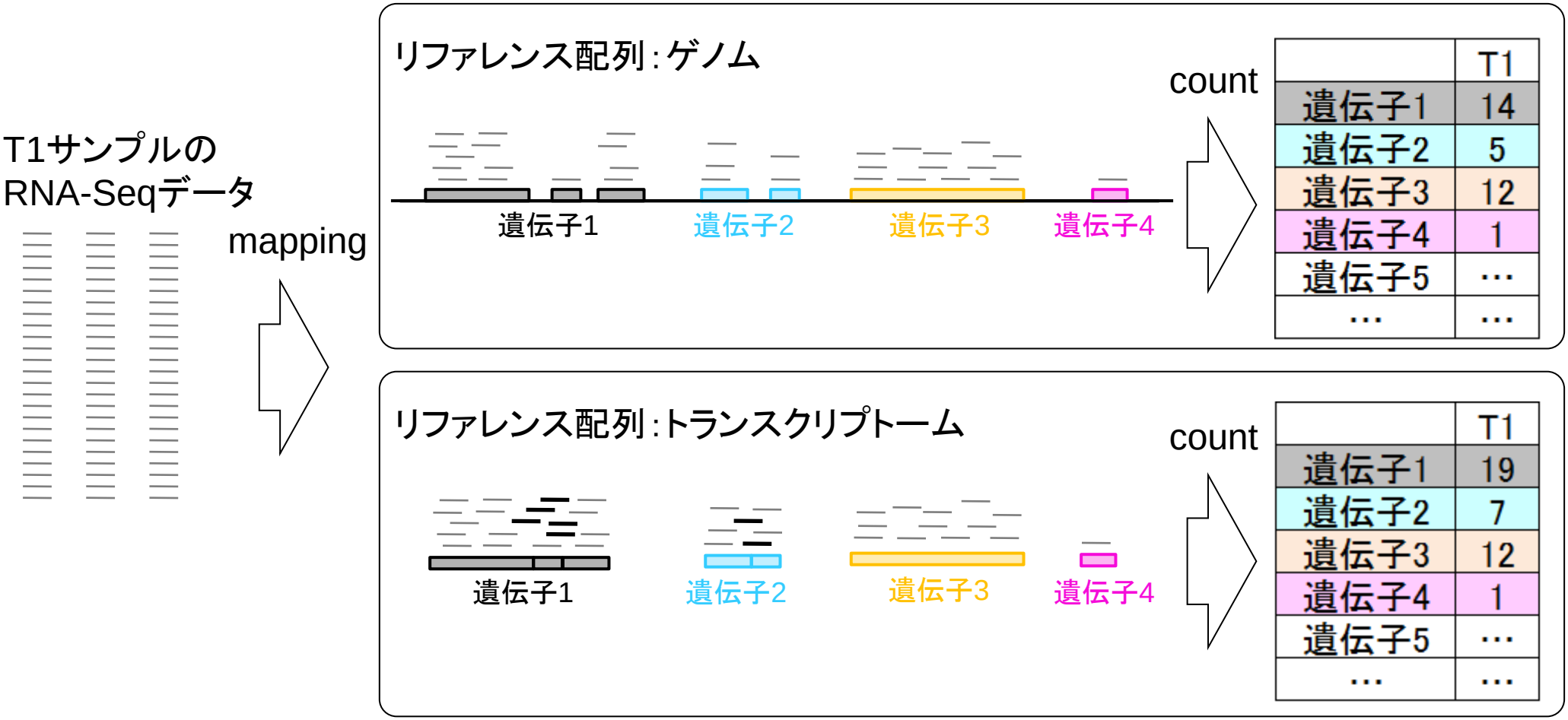
遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



マッピングの基本的なイメージ

- 基本的なマッピングプログラム（bowtieなど）を用いた場合



解析の目的に応じてプログラムを使い分け

- ゲノム配列既知で遺伝子構造を知りたいような場合
  - Cufflinks (Trapnell et al., *Nat. Biotechnol*., 2010)
  - ARTADE2 (Kawaguchi et al., *Bioinformatics*, 2012)
  - …
- (ゲノム配列の有無に関わらず)RefSeqのようなトランスクリプトーム配列にマッピングして比較トランスクリプトーム解析を行いたい場合
  - Bowtie (Langmead et al., *Genome Biol*., 2009)
  - BWA (Li and Durbin, *Bioinformatics*, 2009)
  - …

Cufflinks周辺は専門外です

マッピング結果の出力ファイル形式

- (ゲノム配列の場合)どの染色体上のどの位置に(どのリードが)マッピングされたか、あるいは(トランスクリプトーム配列の場合)どの転写物配列上のどの位置に(どのリードが)マッピングされたかを表すファイル形式(フォーマット)は複数あります:
  - **BED** (Browser Extensible Data) format
    - BEDtools (Quinlan et al., *Bioinformatics*, **26**: 841-842, 2010)
  - GFF (General Feature Format) format
  - **SAM** (Sequence Alignment/Map) format
    - SAMtools (Li et al., *Bioinformatics*, **25**: 2078-2079, 2009)
  - …

マッピング結果ファイルから、どうやって転写物ごとのマップされたリード数をカウントするのか？

http://genome.ucsc.edu/FAQ/FAQformat.html#format1

BED形式

- イントロダクション | NGS | マッピング | (short) readの出力形式について

マッピング | (short) readを眺めると、いろいろな出力形式があることがわかります。
注目すべきは、Sequence Alignment/Map (SAM) formatです。この形式は国際共同研究の1000人のゲノムを解析するという1000 Genomes Projectで採用された(開発された)フォーマットで、("@"から始まる)header sectionと(そうでない)alignment sectionから構成されています。このヒトの目で解読可能な形式がSAMフォーマットで、このバイナリ版がBinary Alignment/Map (BAM)フォーマットというものです。今後SAM/BAM formatという記述を良く見かけるようになることでしょう

代表的な出力ファイル形式

- BED format
- ELAND format
- GFF (General Feature Format)
- GFF3 (General Feature Format)
- SAM (Sequence Alignment/Map)
- SOAP format
- ZOOM format

UCSC Genome Bioinformatics

Home - Genomes - Blat - Tables - Gene Sorter - PCR - Proteome - Help

Frequently Asked Questions: Data File Formats

- BED format
- bigBed format
- BED (detail) format
- PSL format
- GFF format
- GTF format

BED format

BED format provides a flexible way to define the data lines that are displayed in an annotation track. BED lines have three r
and nine additional optional fields. The number of fields per line must be consistent throughout any single set of data in an ann
The order of the optional fields is binding: lower-numbered fields must always be populated if higher-numbered fields are use

The first three required BED fields are:

1. **chrom** - The name of the chromosome (e.g. chr3, chrY, chr2_random) or scaffold (e.g. scaffold10671).
2. **chromStart** - The starting position of the feature in the chromosome or scaffold. The first base in a chromosome is nu
3. **chromEnd** - The ending position of the feature in the chromosome or scaffold. The *chromEnd* base is not included in of the feature. For example, the first 100 bases of a chromosome are defined as *chromStart=0, chromEnd=100*, an bases numbered 0-99.

The 9 additional optional BED fields are:

BED形式

- あるトランスクリプトーム配列（RefSeq）にマップした結果

NM_001190702.1	235	271	U0	0	+
NM_024408.3	7718	7754	U0	0	+
NM_000110.3	2390	2426	U0	0	-
NR_002819.2	2753	2789	U0	0	+
NR_002819.2	2322	2358	U0	0	-
NR_003286.2	1359	1395	U0	0	-
NM_001190470.1	91	127	U0	0	+
NM_014918.4	1389	1425	U0	0	-
NM_002046.3	275	311	U0	0	-
NM_001419.2	1424	1460	U0	0	+

転写物ID　　Start　　End

転写物IDごとの出現数＝マップされたリード数

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

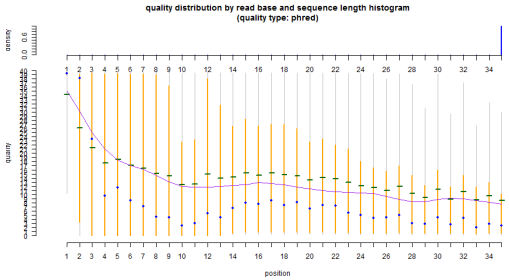
↓

マッピング

↓

データ解析

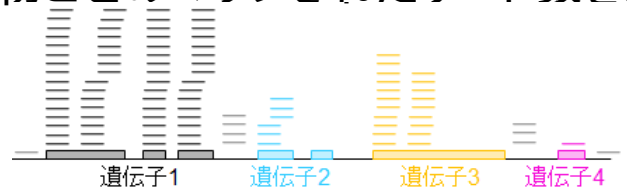
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

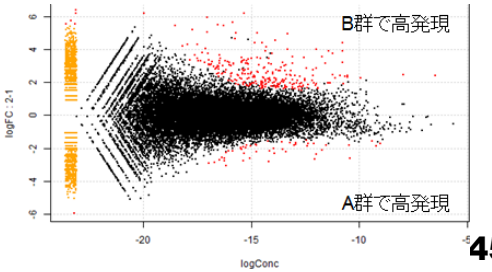
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



• 前処理 | トランスクリプトーム配列へのマップ後のファイルからマップされたリード数をカウント(BED形式ファイル)

手元に「RefSeqのhuman mRNAのmulti-fasta形式のファイル (h_rna.fasta; マップされる側の配列) に対して、参考文献1のKidney 1.5pMのサンプルに対して得られたNGSデータ(SRA ID: SRR002324; マップする側の配列)をBowtieプログラムを用いてマップした結果得られたBED形式ファイル(SRR002324_t.bed)」があるとします。
ここでは、各RefSeq IDに対してマップされたリード数をカウントした結果をファイルに保存す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. SRR002324_t.bed中にあるIDのもののみ、アルファベット順にソートして出力する場合：

```
------    ここから    ------
in_f <- "SRR002324_t.bed"                                   #読み込みたいBED形式ファイル名を指定してin_fに格納
out_f <- "output1.txt"                                      #出力ファイル名を指定してout_fに格納

data <- read.table(in_f)                                    #in_fで指定したファイルを読み込んでdataに格納
ID_list <- as.vector(data[,1])                              #行列dataから1列目の情報のみ抽出してベクトル形式にした
out <- rle(sort(ID_list))                                   #どのIDがいくつあったかをカウントした結果をoutに格納

write.table(out, out_f, sep="¥t", quote=FALSE, row.names=FALSE, col.names=FALSE)    #out_fで指定したファイル名で保存

------    ここまで    ------
```

3. マップされる側の配列 (h_rna.fasta) 中のIDの並びで出力する場合 (複数サンプルの比較の際に便利) ：

```
------    ここから    ------
in_f1 <- "SRR002324_t.bed"                                  #読み込みたいBED形式ファイル名を指定してin_f1に格納
in_f2 <- "h_rna.fasta"                                      #マップに用いたリファレンス配列のファイル名を指定してi
out_f <- "output3.txt"                                      #出力ファイル名を指定してout_fに格納

#BEDファイル中に出現するIDのみ、アルファベット順にソートして生リード数をカウント
data <- read.table(in_f1)                                   #in_f1で指定したファイルを読み込んでdataに格納
ID_list <- as.vector(data[,1])                              #行列dataから1列目の情報のみ抽出してベクトル形式にした
out <- rle(sort(ID_list))                                   #どのIDがいくつあったかをカウントした結果をoutに格納

#マップされる側のファイルのID情報を抽出
library(ShortRead)                                          #パッケージの読み込み
reads <- read.DNAStringSet(in_f2, format="fasta")           #in_f2で指定したファイルを読み込んでreadsに格納

#本番
rawcount <- rep(0, length(reads))                           #最終的に欲しいrawcount値の初期値を0にしておく
names(rawcount) <- sort(names(reads))                       #sort(names(reads))をnames(rawcount)に格納
hoge <- out$lengths                                         #outオブジェクト中のカウント情報をhogeに格納
names(hoge) <- out$values                                   #outオブジェクト中のID情報をnames(hoge)に格納
obj <- is.element(names(rawcount), names(hoge))             #names(rawcount)の並び順でnames(hoge)中のIDが存在する位
rawcount[obj] <- hoge                                       #objがTRUEの位置に生リードカウントを代入

#出力 (この段階でin_f2で読み込んだファイルのIDの並びに変更している)
tmp <- cbind(names(reads), rawcount[names(reads)])          #結果ファイルの1列目がID，2列目がcount情報になるように
write.table(tmp, out_f, sep="¥t", quote=FALSE, row.names=FALSE, col.names=FALSE)    #out_fで指定したファイル名で保

------    ここまで    ------
```

NM_203348.1	3
NM_001008737.1	19
NM_001037228.1	7
NM_033183.2	0
NM_138368.3	56
NM_152833.2	85
NM_001100111.1	0
NM_001102659.1	0
NM_001104548.1	3
NM_001101330.1	5
NR_003083.2	3
NM_003530.3	0
NM_033142.1	0
NM_032030.2	0
NM_001129895.1	0
NR_024114.1	0
NR_024147.1	0
NR_024148.1	1
NR_003545.1	8
NM_020931.2	19
NM_001077637.1	2
NM_198947.3	0
NR_024621.1	0
NM_032576.4	38
NM_001144064.1	0

output3.txt

Rを用いてコピペでマップされたリード数情報を得ることができます

データ解析の前に…

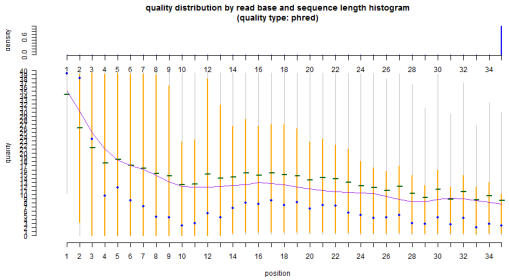
- 研究目的(と手持ちのデータ)をおさらい
  - 一つのサンプル内でどの転写物(or 遺伝子)の発現レベルが高いか低いかを調べたい場合
    - RPKMやFPKMなどの「転写物の長さを考慮して正規化されたデータ」で解析
    - トータルのリード数を補正する必要はないがやってもよい
      - 遺伝子間の発現レベルの大小関係を調べたいだけなので、解析データを定数倍したところで何ら影響を与えないから…
  - サンプル間比較(sample A vs. Bなど)で、発現変動遺伝子(Differentially Expressed Genes; DEGs)を調べたい場合
    - 「トータルのリード数を補正したデータ」で解析
      - 正確には、「サンプル間で発現変動していない遺伝子(non-DEGs)ができるだけ発現変動していないと判定されるように正規化したデータ」
    - 既存のRパッケージを用いて解析を行う場合には、「(整数値のみからなる)生のリードカウントデータ」を入力とし、内部的に上記正規化を行う。

研究目的によってやっていい正規化とやってはいけない(と言われている)正規化がある

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

クオリティチェック



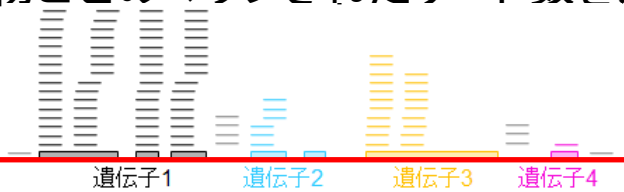
リファレンス配列の作成

アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング

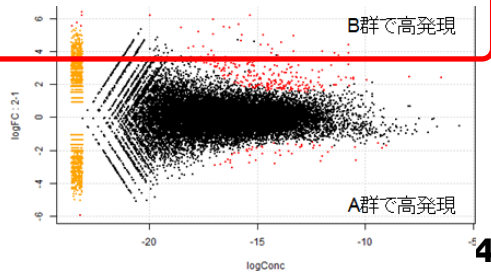
マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

データ解析

発現変動遺伝子のリストアップや、作図など



二群間比較用Rパッケージ

- *DEGseq* (Wang et al., *Bioinformatics*, **26**: 136-138, 2010)
  - ポワソン分布（variance = mean）を仮定しているためばらつきを過少評価
- *edgeR* (Robinson et al., *Bioinformatics*, **26**: 139-140, 2010)
  - 正規化法：TMM法
  - 負の二項分布（variance > mean）を仮定。meanのみのパラメータを用いて現実のばらつきを表現
- *DESeq* (Anders and Huber, *Genome Biol*., **11**: R106, 2010)
  - 正規化法：RLE法(relative log expression)
  - edgeRのモデルをさらに拡張（しているらしい）
- *baySeq* (Hardcastle and Kelly, *BMC Bioinformatics*, **11**:422, 2010)
  - 正規化法：RPM (たぶん)
  - 配列の長さ情報を与えるオプションがあ
  - **データセット中に占めるDEGの割合（P_{DEG}）を一意に返す**
- *NBPSeq* (Di et al., *SAGMB*, **10**:24, 2011)

入力：生のリードカウントからなる遺伝子発現行列
出力：遺伝子ごとの発現変動の度合い（p値など）

理想的な実験デザイン(二群間比較)

- サンプルA vs. Bの比較(Kidney vs. Liver; 腎臓 vs. 肝臓)
  - 生のリードカウントのデータ(整数値)

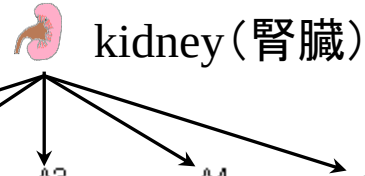
Gene ID	A1	A2	A3	A4	…	B1	B2	B3	B4	…
Gene1										
Gene2										
Gene3										
Gene4										
Gene5										
Gene6										
Gene7										
…										

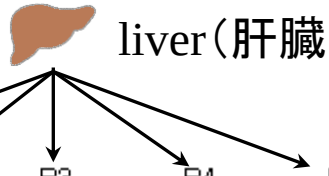
A1: ある生物の**腎臓**
A2: 同じ生物種の別個体の**腎臓**
A3: 同じ生物種のさらに別個体の**腎臓**
…
B1: ある生物の**肝臓**
B2: 同じ生物種の別個体の**肝臓**
…

Biological replicatesのデータ
生物学的なばらつき(個体間の違い)を考慮すべし

分布の話

- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ（の一部）





	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5
EnsemblGeneID	R1 L1 Kidney	R1 L3 Kidney	R1 L7 Kidney	R2 L2 Kidney	R2 L6 Kidney	R1 L2 Liver	R1 L4 Liver	R1 L6 Liver	R1 L8 Liver	R2 L3 Liver
ENSG00000146556	0	0	0	0	0	0	0	0	0	0
ENSG00000197194										
ENSG00000197490										
ENSG00000205292										
ENSG00000177693										
ENSG00000209338										
ENSG00000196573										
ENSG00000177799										
ENSG00000209341										
ENSG00000209342										
ENSG00000209343										
ENSG00000209344										
ENSG00000209346										
ENSG00000209349										
ENSG00000209350	4	7	3	6	7	35	32	31	29	34
ENSG00000209351	0	0	0	0	0	0	0	0	0	0
ENSG00000209352	0	0	1	1	0	2	0	0	0	0
ENSG00000212679	110	131	149	112	118	177	135	141	148	145
ENSG00000212678	12685	13204	12403	13031	13268	9246	9312	8746	8496	9070
ENSG00000185097	0	0	0	0	0	0	0	0	0	0

Technical replicatesのデータ

サンプル内の技術的なばらつき（例：レーン間の違い）の度合いを調べるためのデータであり、このようなデータで二群間比較し、**発現変動遺伝子がどの程度あるかといった数に関する議論は無意味**

解析例：アリエナイ?!数（50%とか）が発現変動遺伝子として検出される

理由：Biological variation > Technical variation

分布の話

- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ（の一部）

kidney（腎臓）

EnsemblGeneID	A1	A2	A3	A4	A5
ENSG00000146556	0	0	0	0	0
ENSG00000197194	0	0	0	0	0
ENSG00000197490	0	0	0	0	0
ENSG00000205292	0	0	0	0	0
ENSG00000177693	0	0	0	0	0
ENSG00000209338	0	0	0	0	0
ENSG00000196573	0	0	0	0	0
ENSG00000177799	0	0	0	0	0
ENSG00000209341	0	0	0	0	0
ENSG00000209342	0	0	2	4	3
ENSG00000209343	0	0	0	0	0
ENSG00000209344	0	0	0	0	0
ENSG00000209346	0	0	0	0	0
ENSG00000209349	0	0	0	0	0
ENSG00000209350	4	7	3	6	7
ENSG00000209351	0	0	0	0	0
ENSG00000209352	0	0	1	1	0
ENSG00000212679	110	131	149	112	118
ENSG00000212678	12685	13204	12403	13031	13268
ENSG00000185097	0	0	0	0	0
…	…	…	…	…	…
総リード数	1804977	1855190	1742426	1927517	1963420

RPM
正規化

EnsemblGeneID	A1	A2	A3	A4	A5
ENSG00000209342	0.0	0.0	1.1	2.1	1.5
ENSG00000209350	2.2	3.8	1.7	3.1	3.6
ENSG00000209352	0.0	0.0	0.6	0.5	0.0
ENSG00000212679	60.9	70.6	85.5	58.1	60.1
ENSG00000212678	7027.8	7117.3	7118.2	6760.5	6757.6
ENSG00000197049	0.0	0.0	0.0	0.5	0.0
ENSG00000177757	1.1	0.0	1.1	0.5	1.5
ENSG00000177750	0.6	2.2	1.7	1.6	3.6
ENSG00000177741	0.6	0.5	0.0	3.1	0.0
ENSG00000198907	3.3	0.0	3.4	1.0	0.0
ENSG00000187634	27.1	23.2	23.5	21.8	23.9
ENSG00000188976	40.4	41.5	39.0	36.3	41.8
ENSG00000187961	8.3	8.1	7.5	6.2	7.6
ENSG00000187583	0.6	0.5	1.7	0.0	1.5
ENSG00000187642	2.2	2.7	6.9	4.7	4.6
ENSG00000188290	5.0	5.4	6.9	5.2	6.6
ENSG00000187608	6.6	5.9	4.0	8.3	6.6
ENSG00000188157	227.1	223.2	200.9	239.7	240.4
ENSG00000131591	5.5	4.9	4.0	6.2	8.1
ENSG00000215916	5.5	4.9	4.6	6.7	8.7
…	…	…	…	…	…
総リード数	1000000	1000000	1000000	1000000	1000000

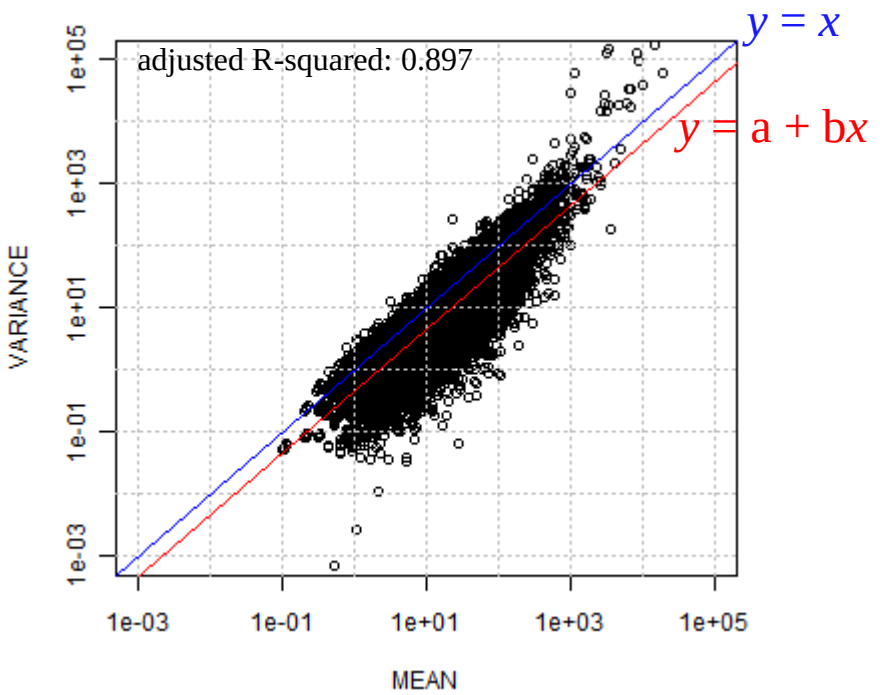
$$12{,}685 \times \frac{1{,}000{,}000}{1{,}804{,}977} = 7027.8$$

分布の話

- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ（の一部）

kidney（腎臓）

EnsemblGeneID	A1	A2	A3	A4	A5	MEAN	VARIANCE
ENSG00000209342	0.0	0.0	1.1	2.1	1.5	1.0	0.9
ENSG00000209350	2.2	3.8	1.7	3.1	3.6	2.9	0.8
ENSG00000209352	0.0	0.0	0.6	0.5	0.0	0.2	0.1
ENSG00000212679	60.9	70.6	85.5	58.1	60.1	67.1	129.8
ENSG00000212678	7027.8	7117.3	7118.2	6760.5	6757.6	6956.3	33770.4
ENSG00000197049	0.0	0.0	0.0	0.5	0.0	0.1	0.1
ENSG00000177757	1.1	0.0	1.1	0.5	1.5	0.9	0.4
ENSG00000177750	0.6	2.2	1.7	1.6	3.6	1.9	1.2
ENSG00000177741	0.6	0.5	0.0	3.1	0.0	0.8	1.7
ENSG00000198907	3.3	0.0	3.4	1.0	0.0	1.6	2.9
ENSG00000187634	27.1	23.2	23.5	21.8	23.9	23.9	3.9
ENSG00000188976	40.4	41.5	39.0	36.3	41.8	39.8	5.0
ENSG00000187961	8.3	8.1	7.5	6.2	7.6	7.5	0.7
ENSG00000187583	0.6	0.5	1.7	0.0	1.5	0.9	0.5
ENSG00000187642	2.2	2.7	6.9	4.7	4.6	4.2	3.4
ENSG00000188290	5.0	5.4	6.9	5.2	6.6	5.8	0.8
ENSG00000187608	6.6	5.9	4.0	8.3	6.6	6.3	2.4
ENSG00000188157	227.1	223.2	200.9	239.7	240.4	226.3	258.8
ENSG00000131591	5.5	4.9	4.0	6.2	8.1	5.8	2.5
ENSG00000215916	5.5	4.9	4.6	6.7	8.7	6.1	2.8
…	…	…	…	…	…	…	…
総リード数	1000000	1000000	1000000	1000000	1000000		

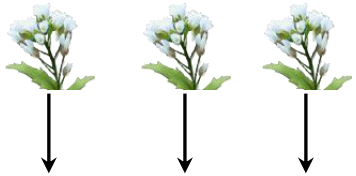


Technical replicatesのデータは：
・（遺伝子の）VARIANCEはそのMEANで説明可能である
・VARIANCE ≒ MEAN
・ポアソン分布に従う
・ポアソンモデルが適用可能

分布の話

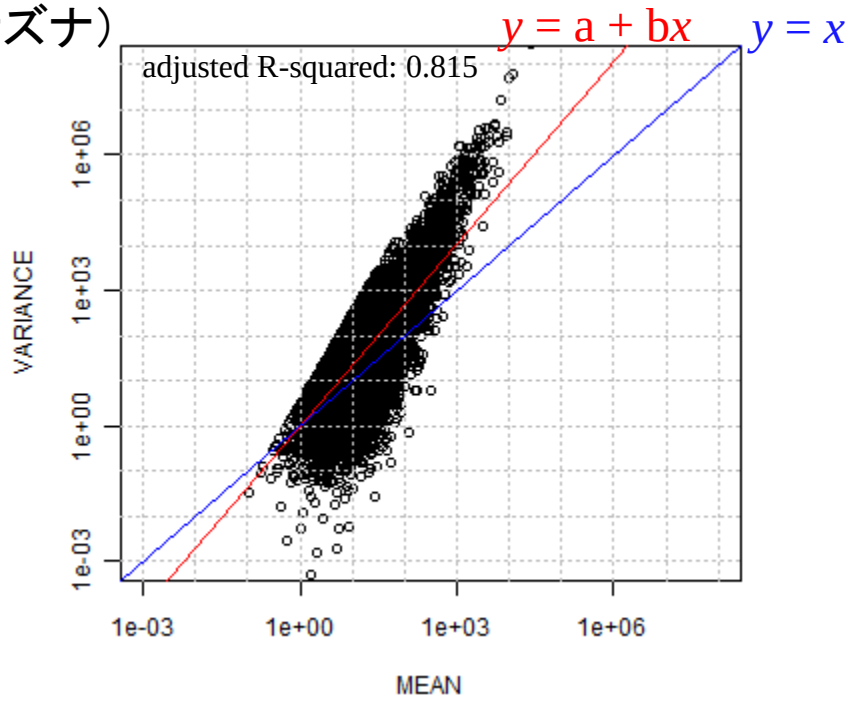
生物アイコン(http://biosciencedbc.jp/taxonomy_icon/taxonomy_icon.cgi)

- 例題:Cumbie et al., *PLoS ONE*, **6**: e25279, 2011のデータ(の一部)



Arabidopsis(シロイヌナズナ)

	mock1	mock2	mock3	MEAN	VARIANCE
AT1G01010	18.4	39.8	12.3	23.5	209.1
AT1G01020	22.6	23.3	9.8	18.6	57.5
AT1G01030	8.4	12.4	8.0	9.6	6.0
AT1G01040	37.9	22.2	19.6	26.6	97.1
AT1G01050	25.8	40.3	27.6	31.2	62.9
AT1G01060	0.0	7.8	0.6	2.8	18.6
AT1G01070	8.4	17.6	1.8	9.3	62.5
AT1G01080	89.4	98.8	117.2	101.8	200.2
AT1G01090	153.0	178.9	172.7	168.2	183.1
AT1G01100	59.4	64.6	75.5	66.5	67.1
AT1G01110	0.0	0.5	0.3	0.3	0.1
AT1G01120	119.9	97.7	82.8	100.1	347.3
AT1G01130	4.7	5.7	0.3	3.6	8.2
AT1G01140	95.2	62.0	43.6	66.9	683.3
…	…	…	…	…	…
総リード数	1000000	1000000	1000000		



Biological replicatesのデータは:
- **VARIANCE > MEAN**
- 負の二項(NB)分布に従う
- NBモデルが適用可能

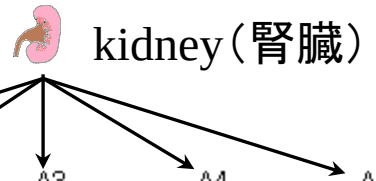
なぜ沢山の方法が存在しうるのか？

- *DEGseq* (Wang et al., *Bioinformatics*, **26**: 136-138, 2010) $\mathrm{VAR} = \mu$
  - ポワソン分布（variance = mean）を仮定しているためばらつきを過少評価
- ***edgeR*** (Robinson et al., *Bioinformatics*, **26**: 139-140, 2010) $\mathrm{VAR} = \mu(1+\phi\mu)$
  - 正規化法：TMM法
  - 負の二項分布（variance > mean）を仮定。
- *DESeq* (Anders and Huber, *Genome Biol.*, **11**: R106, 2010) $\mathrm{VAR} = \mu(1+\phi_\mu\mu)$
  - 正規化法：RLE法(relative log expression)
  - edgeRのモデルをさらに拡張（しているらしい）
- ***baySeq*** (Hardcastle and Kelly, *BMC Bioinformatics*, **11**:422, 2010)
  - 正規化法：RPM (たぶん)
  - 配列の長さ情報を与えるオプションがある
  - データセット中に占めるDEGの割合（P_{DEG}）を一意に返す
- *NBPSeq* (Di et al., *SAGMB*, **10**:24, 2011) $\mathrm{VAR} = \mu(1+\phi\mu^{\alpha-1})$

Ans. VarianceとMeanの関係を表現する手段が沢山あるから

edgeRを使ってみる

- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ



liver（肝臓）

	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5
EnsemblGeneID	R1 L1 Kidney	R1 L3 Kidney	R1 L7 Kidney	R2 L2 Kidney	R2 L6 Kidney	R1 L2 Liver	R1 L4 Liver	R1 L6 Liver	R1 L8 Liver	R2 L3 Liver
ENSG00000146556	0	0	0	0	0	0	0	0	0	0
ENSG00000197194	0	0	0	0	0	0	0	0	0	0
ENSG00000197490	0	0	0	0	0	0	0	0	0	0
ENSG00000205292	0	0	0	0	0	0	0	0	0	0
ENSG00000177693	0	0	0	0	0	0	0	0	0	0
ENSG00000209338	0	0	0	0	0	0	0	0	0	0
ENSG00000196573	0	0	0	0	0	0	0	0	0	0
ENSG00000177799	0	0	0	0	0	0	0	0	0	0
ENSG00000209341	0	0	0	0	0	0	0	0	0	0
ENSG00000209342	0	0	2	4	3	0	0	0	1	0
ENSG00000209343	0	0	0	0	0	0	0	0	0	0
ENSG00000209344	0	0	0	0	0	0	0	0	0	0
ENSG00000209346	0	0	0	0	0	0	0	0	0	0
ENSG00000209349	0	0	0	0	0	0	0	0	0	0
ENSG00000209350	4	7	3	6	7	35	32	31	29	34
ENSG00000209351	0	0	0	0	0	0	0	0	0	0
ENSG00000209352	0	0	1	1	0	2	0	0	0	0
ENSG00000212679	110	131	149	112	118	177	135	141	148	145
ENSG00000212678	12685	13204	12403	13031	13268	9246	9312	8746	8496	9070
ENSG00000185097									0	0

ファイル名：**SupplementaryTable2_changed.txt**
内容：A群が最初の**5**列、B群が残りの**5**列のデータ
解析結果を**hoge2.txt**という名前でファイルに出力したい

edgeRを使ってみる

• 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | edgeR (Robinson_2010)

2011/12/26にDispersionの計算方法をcommon dispersionからtag-wise dispersionに変更しました。ご注意ください。
参考文献1のedgeRパッケージを用いて解析を行
empirical Bayes)を実装したものです。(おそらくe
ケージの使用例中ではTMM法で得られた正規化
した位置に組み込めばいいです。また、参考文献
ますが、このような極端な例でなくても常にTMM法
法で得られた正規化係数は1に近い値となるので
入力ファイルは、"遺伝子発現行列"形式のもので
ここでは、サンプルデータ2 (つまりSupplementary

「ファイル」－「ディレクトリの変更」で解析したいフ

```
1. 基本形
------    ここから    ------
in_f <- "SupplementaryTable2_changed.txt"
out_f <- "hoge2.txt"
param1 <- 5
param2 <- 5

library(DEGseq)
library(edgeR)
data <- read.table(in_f, header=TRUE, row.names
data <- as.matrix(data)

data.cl <- c(rep(1, param1), rep(2, param2))
d <- DGEList(counts=data, group=data.cl)
d <- calcNormFactors(d)
d <- estimateCommonDisp(d)
d <- estimateTagwiseDisp(d, prop.used=0.5, grid
out <- exactTest(d)
fdr <- p.adjust(out$table$p.value
tmp <- cbind(rownames(data), data
write.table(tmp, out_f, sep="¥t"

------    ここまで    ------
```

RGui (64-bit)

ファイル　編集　閲覧　その他　パッケージ　ウインドウ　ヘルプ

R Console

```
R version 2.14.1 (2011-12-22)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-mingw32/x64 (64-bit)

Rは、自由なソフトウェアであり、「完全に無保証」です。
一定の条件に従えば、自由にこれを再配布することができます。
配布条件の詳細に関しては、'license()'あるいは'licence()'と入力してください。

Rは多くの貢献者による共同プロジェクトです。
詳しくは'contributors()'と入力してください。
また、RやRのパッケージを出版物で引用する際の形式については
'citation()'と入力してください。

'demo()'と入力すればデモをみることができます。
'help()'とすればオンラインヘルプが出ます。
'help.start()'でHTMLブラウザによるヘルプがみられます。
'q()'と入力すればRを終了します。
```

ファイル名：SupplementaryTable2_changed.txt
内容：A群が最初の5列、B群が残りの5列のデータ
解析結果をhoge2.txtという名前でファイルに出力したい

edgeRを使ってみる

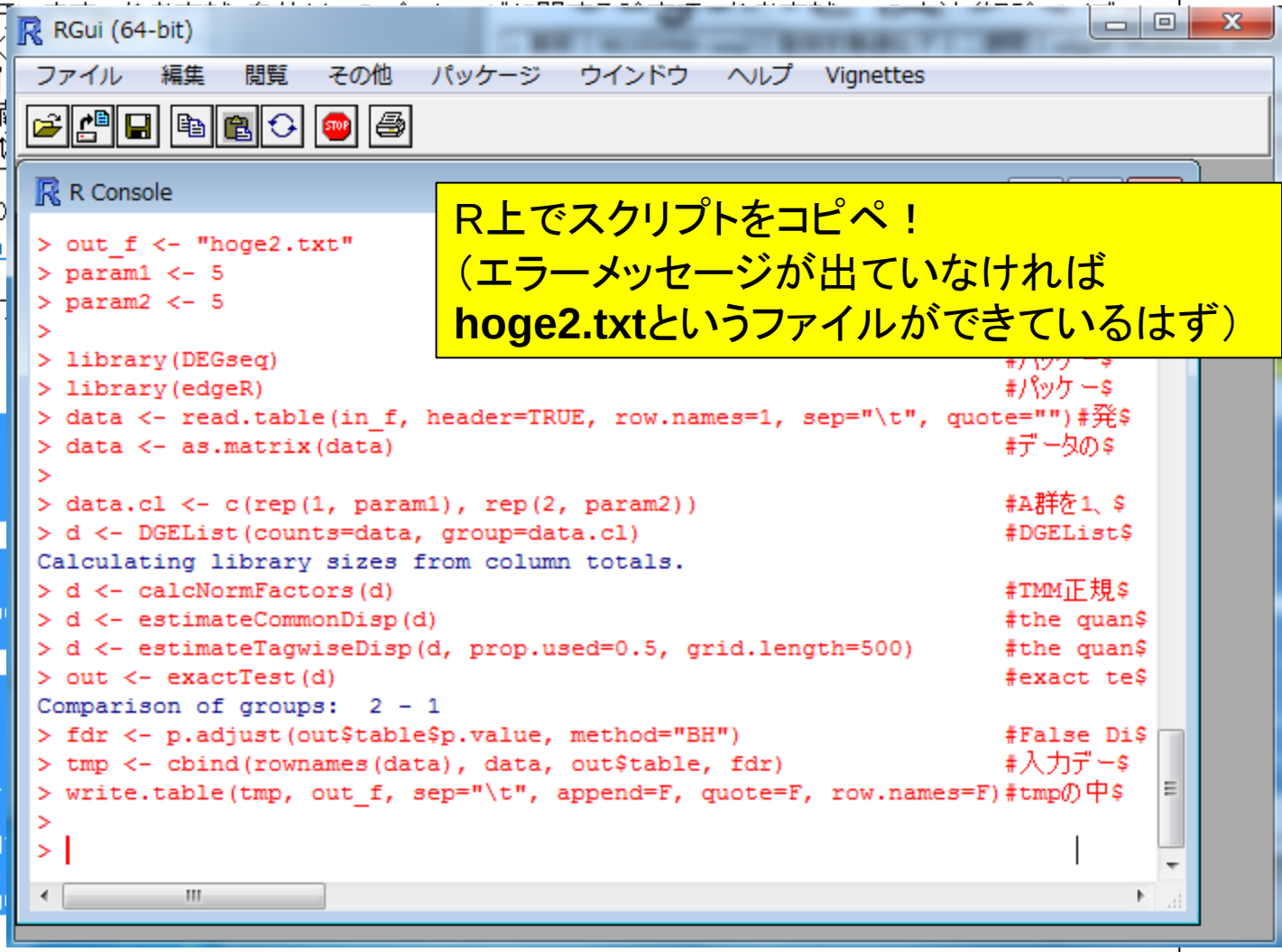
• 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | edgeR (Robinson_2010)

2011/12/26にDispersionの計算方法をcommon dispersionからtag-wise dispersionに変更しました。ご注意ください。
参考文献1のedgeRパッケージを用いて解析を行
empirical Bayes)を実装したものです。(おそらく
ケージの使用例中ではTMM法で得られた正規化
した位置に組み込めばいいです。また、参考文献
ますが、このような極端な例でなくても常にTMM
法で得られた正規化係数は1に近い値となるの
入力ファイルは、"遺伝子発現行列"形式のもの
ここでは、サンプルデータ2 (つまりSupplementa

「ファイル」-「ディレクトリの変更」で解析したい

1. 基本形

```
------    ここから    ------
in_f <- "SupplementaryTable2_changed.txt"
out_f <- "hoge2.txt"
param1 <- 5
param2 <- 5

library(DEGseq)
library(edgeR)
data <- read.table(in_f, header=TRUE, row.nam
data <- as.matrix(data)

data.cl <- c(rep(1, param1), rep(2, param2))
d <- DGEList(counts=data, group=data.cl)
d <- calcNormFactors(d)
d <- estimateCommonDisp(d)
d <- estimateTagwiseDisp(d, prop.used=0.5, gr
out <- exactTest(d)
fdr <- p.adjust(out$table$p.value, method="BH
tmp <- cbind(rownames(data), data, out$table,
write.table(tmp, out_f, sep="¥t", append=F, q

------    ここまで    ------
```



R上でスクリプトをコピペ！
（エラーメッセージが出ていなければ
hoge2.txtというファイルができているはず）

edgeRを使ってみる

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
rownames(data)	R1 L1 Kidr	R1 L3 Kidr	R1 L7 Kidr	R2 L2 Kidr	R2 L6 Kidr	R1 L2 Live	R1 L4 Live	R1 L6 Live	R1 L8 Live	R2 L3 Live	logConc	logFC	p.value	fdr
ENSG00000146556	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000197194	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000197490	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000205292	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000177693	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209338	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000196573	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000177799	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209341	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209342	0	0	2	4	3	0	0	0	1	0	−21.4867	−2.44627	0.179885	0.242167
ENSG00000209343	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209344	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209346	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209349	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209350	4	7	3	6	7	35	32	31	29	34	−17.0288	3.299663	1.78E−40	5.60E−40
ENSG00000209351	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209352	0	0	1	1	0	2	0	0	0	0	−22.0717	0.72357	1	1
ENSG00000212679	110	131	149	112	118	177	135	141	148	145	−13.6603	0.98883	4.88E−22	1.37E−21
ENSG00000212678	12685	13204	12403	13031	13268	9246	9312	8746	8496	9070	−7.35443	0.197539	6.28E−11	1.52E−10
ENSG00000185097	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104
ENSG00000209353	0	0	0	0	0	0	0	0	0	0	−50.0161	0	2.29E−104	8.09E−104

一番右側の数値がFalse Discovery Rate (FDR)
この列（O列）で昇順にソートすれば任意の閾値を満たす遺伝子数がわかる
・19,785個がFDR < 0.01を満たす
・21,291個がFDR < 0.05を満たす

edgeRを使ってみる

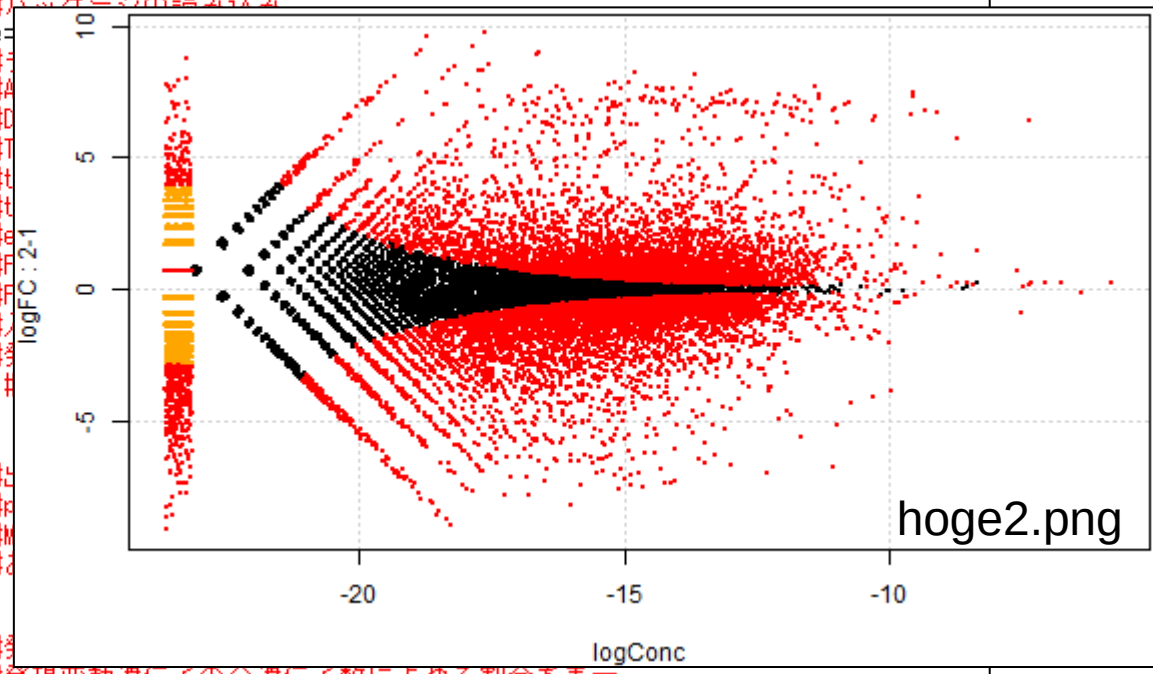
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
rownames(data)	R1 L1 Kidr	R1 L3 Kidr	R1 L7 Kidr	R2 L2 Kidr	R2 L6 Kidr	R1 L2 Live	R1 L4 Live	R1 L6 Live	R1 L8 Live	R2 L3 Live	logConc	logFC	p.value	fdr
ENSG00000116285	115	144	115	143	153	1669	1753	1710	1675	1794	−11.8424	4.407151	0	0
ENSG00000049239	183	232	179	207	199	838	822	814	773	895	−12.0808	2.773867	0	0
ENSG00000186510	515	564	516	568	590	6	1	1	3	2	−15.5081	−7.00256	0	0
ENSG00000184908	484	486	463	573	512	4	2	5	4	3	−15.3378	−6.40426	0	0
ENSG00000142949	332	320	312	350	354	732	772	716	711	808	−11.7855	1.888007	0	0
ENSG00000117472	572	614	603	624	688	14	17	15	13	16	−14.1581	−4.64598	0	0
ENSG00000162366	730	782	720	832	866	4	8	7	7	6	−14.6019	−6.21592	0	0
ENSG00000121310	229	223	247	228	239	1832	1805	1812	1693	1954	−11.4022	3.686564	0	0
ENSG00000116171	542	568	545	548	601	1777	1800	1817	1663	1845	−10.7847	2.390384	0	0
ENSG00000162391	435	444	414	455	450	5	2	5	6	7	−15.1986	−5.73479	0	0
ENSG00000116133	632	681	622	733	702	3534	3396	3178	3196	3657	−10.1878	3.054915	0	0
ENSG00000169174	10	8	8	7	13	223	230	221	173	219	−15.281	5.257754	0	0
ENSG00000157131	14	11	13	7	14	1352	1405	1400	1345	1402	−13.7532	7.59514	0	0
ENSG00000021852	10	12	11	4	20	968	1002	969	982	982	−14.0249	7.151354	0	0
ENSG00000132855	82	96	86	76	90	822	874	823	821	885	−12.6749	4.020325	0	0
ENSG00000134243	919	875	849	883	937	86	93	77	75	94	−12.6438	−2.66983	0	0
ENSG00000163399	7334	7494	6959	7702	7744	284	272	272	250	243	−10.2953	−4.0931	0	0
ENSG00000134240	170	189	180	191	199	2161	2229	2166	2019	2393	−11.4316	4.284652	0	0
ENSG00000168509	9	10	7	8	8	696	710	736	666	711	−14.4847	7.112899	0	0
ENSG00000143384	582	626	568	626	618	1164	1236	1126	1134	1234	−11.0288	1.688325	0	0
ENSG00000197956	942	961	886	1071	995	64	56	58	62	60	−12.8347	−3.29241	0	0

Top-ranked geneの生リードカウントを眺めても確かに発現変動 (Kidney << Liver)していることが分かる

edgeRを使ってみる

- M-A plotを描画（FDR < 0.01を満たすものを赤色で表示）

```
7. MA-plotも描く場合(FDR < 0.01を満たすものを赤色で示したMA-plotをファイルに保存)
------    ここから    ------
in_f <- "SupplementaryTable2_changed.txt"                    #読み込みたい発現データファイルを指定してin_fに格納
out_f1 <- "hoge2.txt"                                        #出力ファイル名を指定
out_f2 <- "hoge2.png"                                        #出力ファイル名を指定
param1 <- 5                                                  #A群のサンプル数を指定
param2 <- 5                                                  #B群のサンプル数を指定
param3 <- 0.01                                               #MA-plot描画時のFDRの閾値を指定
param4 <- c(600, 400)                                        #MA-plotのファイル出力時の横幅(単位はピクセル)と縦幅を指定

library(DEGseq)                                              #パッケージの読み込み
library(edgeR)                                               #パッケージの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote=
data <- as.matrix(data)                                      #
data.cl <- c(rep(1, param1), rep(2, param2))                 #A
d <- DGEList(counts=data, group=data.cl)                     #D
d <- calcNormFactors(d)                                      #T
d <- estimateCommonDisp(d)                                   #t
d <- estimateTagwiseDisp(d, prop.used=0.5, grid.length=500)  #t
out <- exactTest(d)                                          #e
fdr <- p.adjust(out$table$p.value, method="BH")              #F
rank_edgeR <- rank(fdr)                                      #F
hoge <- cbind(rownames(data), data, out$table, fdr, rank_edgeR)  #
tmp <- hoge[order(rank_edgeR),]                              #
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)#

#MA-plotを描画
png(out_f2, width=param4[1], height=param4[2])               #
obj <- rownames(data)[fdr < param3]                          #
plotSmear(d, de.tags=obj)                                    #M
dev.off()                                                    #d

#おまけ
length(obj)                                                  #
length(obj)/nrow(data)                                       #発現変動遺伝子の全遺伝子数に占める割合を表示

------    ここまで    ------
```

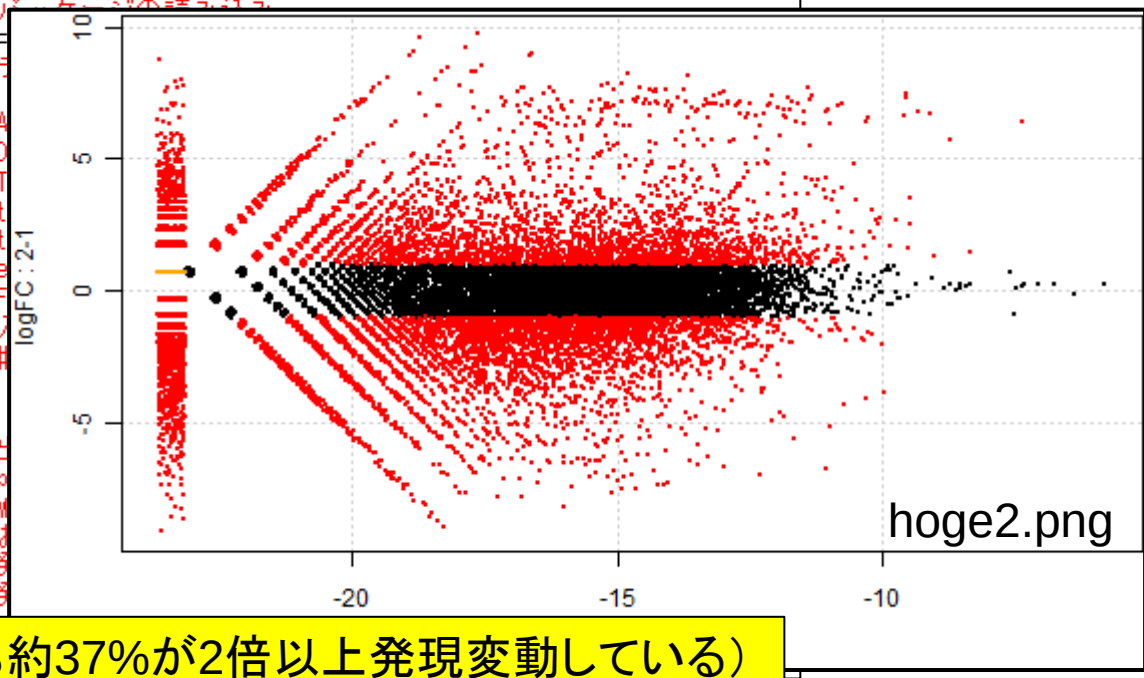


19,794個（全遺伝子数のうち約62%がFDR < 0.01を満たす）

edgeRを使ってみる

- M-A plotを描画（2倍以上発現変動しているものを赤色で表示）

```
6. MA-plotも描く場合(5.のMA-plotで大きさを指定してpng形式ファイルに保存したいとき)
------    ここから    ------
in_f <- "SupplementaryTable2_changed.txt"                    #読み込みたい発現データファイルを指定してin_fに格納
out_f1 <- "hoge2.txt"                                        #出力ファイル名を指定
out_f2 <- "hoge2.png"                                        #出力ファイル名を指定
param1 <- 5                                                  #A群のサンプル数を指定
param2 <- 5                                                  #B群のサンプル数を指定
param3 <- 2                                                  #MA-plot描画時の倍率変化の閾値を指定
param4 <- c(600, 400)                                        #MA-plotのファイル出力時の横幅(単位はピクセル)と縦幅を指定

library(DEGseq)                                              #パッケージの読み込み
library(edgeR)
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote=
data <- as.matrix(data)

data.cl <- c(rep(1, param1), rep(2, param2))
d <- DGEList(counts=data, group=data.cl)
d <- calcNormFactors(d)
d <- estimateCommonDisp(d)
d <- estimateTagwiseDisp(d, prop.used=0.5, grid.length=500)
out <- exactTest(d)
fdr <- p.adjust(out$table$p.value, method="BH")
tmp <- cbind(rownames(data), data, out$table, fdr)
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)

#MA-plotを描画
png(out_f2, width=param4[1], height=param4[2])
obj <- rownames(data)[abs(out$table$logFC) >= log2(param3)]
plotSmear(d, de.tags=obj)
dev.off()
length(obj)
length(obj)/nrow(data)

------    ここまで    --
```



11787個（全遺伝子数のうち約37%が2倍以上発現変動している）
このやり方はダメなんです

倍率変化がだめな理由をデモ

- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ

	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5
EnsemblGeneID	R1 L1 Kidney	R1 L3 Kidney	R1 L7 Kidney	R2 L2 Kidney	R2 L6 Kidney	R1 L2 Liver	R1 L4 Liver	R1 L6 Liver	R1 L8 Liver	R2 L3 Liver
ENSG00000146556	0	0	0	0	0	0	0	0	0	0
ENSG00000197194	0	0	0	0	0	0	0	0	0	0
ENSG00000197490	0	0	0	0	0	0	0	0	0	0
ENSG00000205292	0	0	0	0	0	0	0	0	0	0
ENSG00000177693	0	0	0	0	0	0	0	0	0	0
ENSG00000209338	0	0	0	0	0	0	0	0	0	0
ENSG00000196573	0	0	0	0	0	0	0	0	0	0
ENSG00000177799	0	0	0	0	0	0	0	0	0	0
ENSG00000209341	0	0	0	0	0	0	0	0	0	0
ENSG00000209342	0	0	2	4	3	0	0	0	1	0
ENSG00000209343	0	0	0	0	0	0	0	0	0	0
ENSG00000209344	0	0								
ENSG00000209346	0	0								
ENSG00000209349	0	0	0	0	0	0	0	0	0	0
ENSG00000209350	4	7	3	6	7	35	32	31	29	34
ENSG00000209351	0	0	0	0	0	0	0	0	0	0
ENSG00000209352	0	0	1	1	0	2	0	0	0	0
ENSG00000212679	110	131	149	112	118	177	135	141	148	145
ENSG00000212678	12685	13204	12403	13031	13268	9246	9312	8746	8496	9070
ENSG00000185097	0	0	0	0	0	0	0	0	0	0

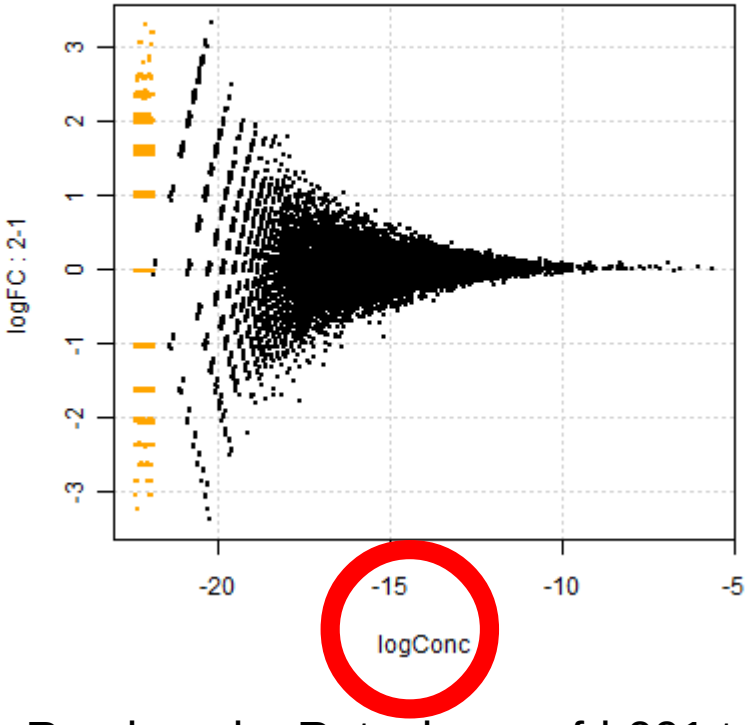
発現変動遺伝子がないデータで二群間比較をしてみる

A群　　B群

倍率変化がだめな理由をデモ

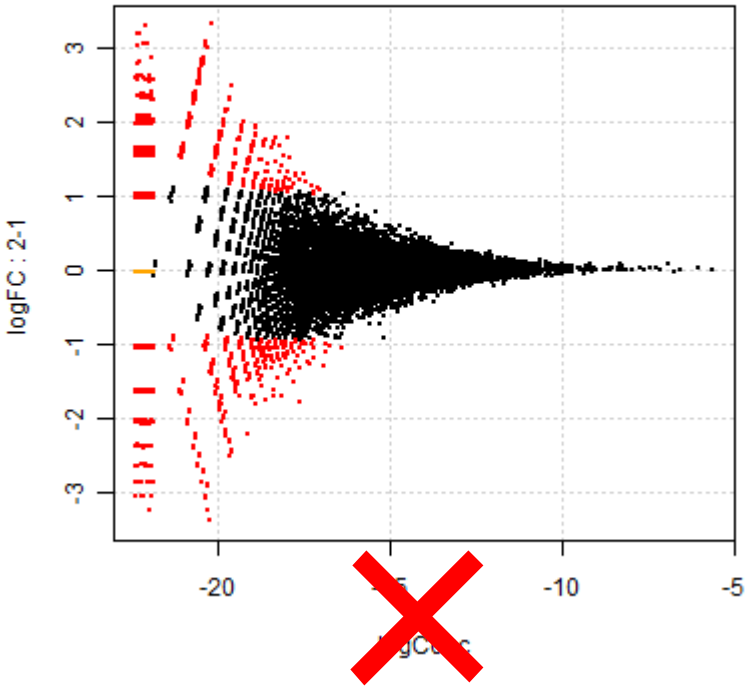
- 例題：Marioni et al., *Genome Res.*, **18**: 1509-1517, 2008のデータ（の一部）
  - (A1, A2) vs. (A3, A4)の二群間比較結果

*edgeR*でFDR < 0.01を満たすものは0個



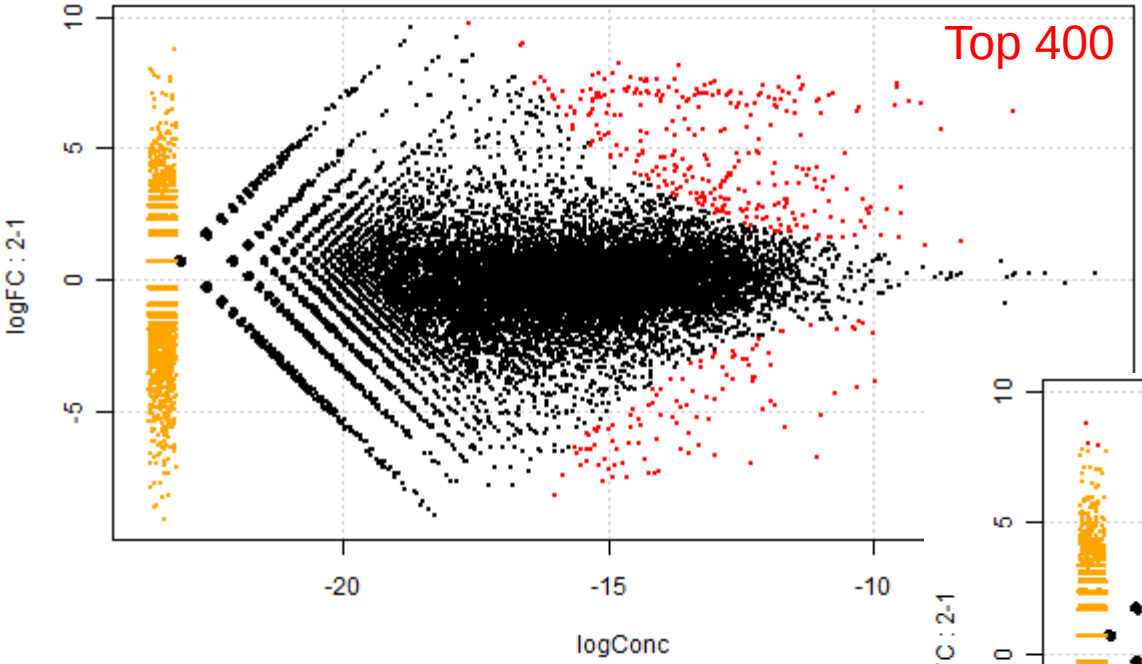
Rcode_edgeR_tech_rep_fdr001.txt

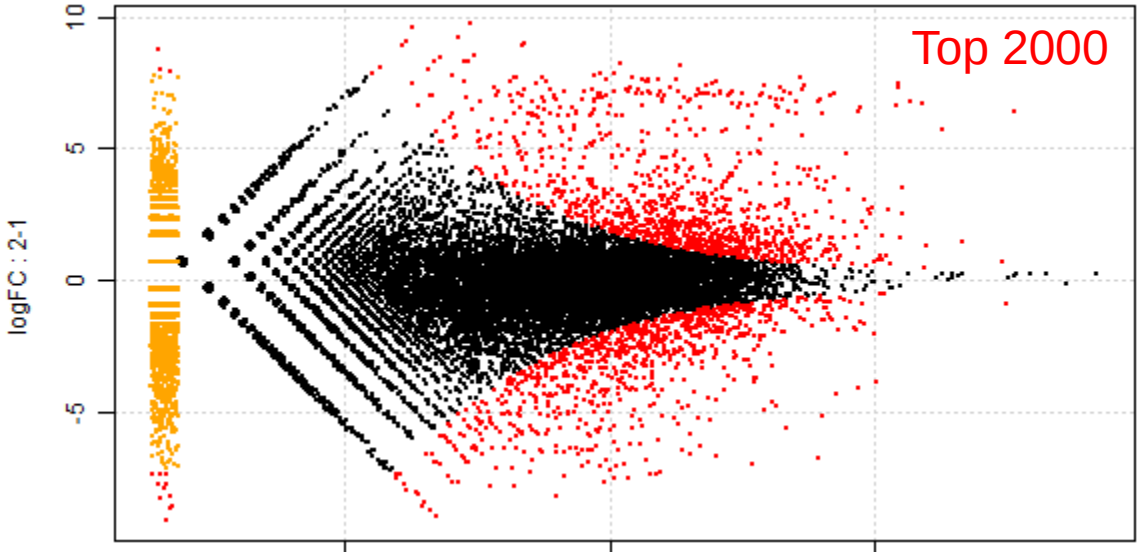
（*edgeR*で）2倍以上発現変動しているものは3814個



Rcode_edgeR_tech_rep_fc2.txt

低発現領域でlog比が大きくなる現象をうまくモデル化することが重要



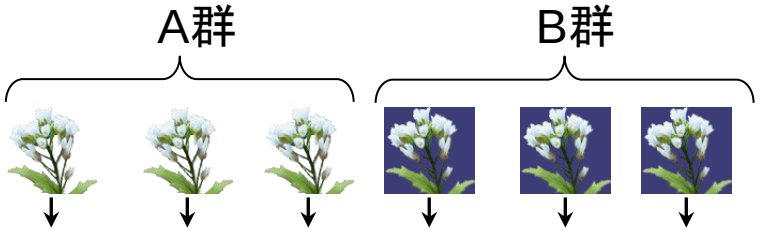


低い ← 全体的な発現レベル → 高い

こんな感じでランキングすることが重要です

Biological replicatesの3 vs. 3サンプル

- 例題：Cumbie et al., *PLoS ONE*, **6**: e25279, 2011のArabidopsisデータ



26,221 genes

identifier	mock1	mock2	mock3	hrcc1	hrcc2	hrcc3
AT1G01010	35	77	40	46	64	60
AT1G01020	43	45	32	43	39	49
AT1G01030	16	24	26	27	35	20
AT1G01040	72	43	64	66	25	90
AT1G01050	49	78	90	67	45	60
AT1G01060	0	15	2	0	21	8
AT1G01070	16	34	6	9	20	1
AT1G01080	170	191	382	127	98	184
AT1G01090	291	346	563	171	116	453
AT1G01100	113	125	246	78	27	361
AT1G01110	0	1	1	0	0	0

data_arab.txt

オリジナルは” AT4G32850”のものが重複して存在していたため19520行目のデータを予め除去している

*edgeR*をdefaultの手順(edgeR/default)で実行

Rcode_edgeR_default.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "data_arab.txt"                                                    #読み込みたい発現データファイルを指定してin_fに格納
out_f1 <- "result_edgeR_default.txt"                                       #出力ファイル名を指定
out_f2 <- "result_edgeR_default.png"                                       #出力ファイル名を指定
param1 <- 3                                                                #A群のサンプル数を指定
param2 <- 3                                                                #B群のサンプル数を指定
param3 <- 0.05                                                             #MA-plot描画時のFDRの閾値を指定
param4 <- c(600, 400)                                                      #MA-plotのファイル出力時の横幅(単位はピクセル)と縦幅を指定

library(DEGseq)                                                            #パッケージの読み込み
library(edgeR)                                                             #パッケージの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#発現データファイルの読み込み
data <- as.matrix(data)                                                    #データの型をmatrixにしている
data.cl <- c(rep(1, param1), rep(2, param2))                               #A群を1、B群を2としたベクトルdata.clを作成
d <- DGEList(counts=data, group=data.cl)                                   #DGEListオブジェクトを作成してdに格納
d <- calcNormFactors(d)                                                    #TMM正規化(参考文献5)を実行
d <- estimateCommonDisp(d)                                                 #the quantile-adjusted conditional maximum likelihood (qCML)
d <- estimateTagwiseDisp(d, prop.used=0.5, grid.length=500)                #the quantile-adjusted conditional maximum likelihood (qCML)
out <- exactTest(d)                                                        #exact test (正確確率検定)で発現変動遺伝子を計算した結果をou
fdr <- p.adjust(out$table$p.value, method="BH")                            #False Discovery Rate (FDR)を計算し、結果をfdrに格納
rank_edgeR <- rank(fdr)                                                    #FDR値でランキングした結果をrank_edgeRに格納
hoge <- cbind(rownames(data), data, out$table, fdr, rank_edgeR)            #入力データの右側に、「logConc (M-A plotのAに相当するもの；
tmp <- hoge[order(rank_edgeR),]                                            #発現変動の度合いでソートした結果をtmpに格納
write.table(tmp, out_f1, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身をout_f1で指定したファイル名で保存。

#MA-plotを描画
png(out_f2, width=param4[1], height=param4[2])                             #出力ファイルの各種パラメータを指定
obj <- rownames(data)[fdr < param3]                                        #param3で指定したFDRの閾値を満たす遺伝子名情報をobjに格納
plotSmear(d, de.tags=obj)                                                  #MA-plotの基本形に加え、発現変動遺伝子に相当する
dev.off()                                                                  #おまじない
length(obj)                                                                #発現変動遺伝子数を表示
length(obj)/nrow(data)                                                     #発現変動遺伝子の全遺伝子数に占める割合を表示
```

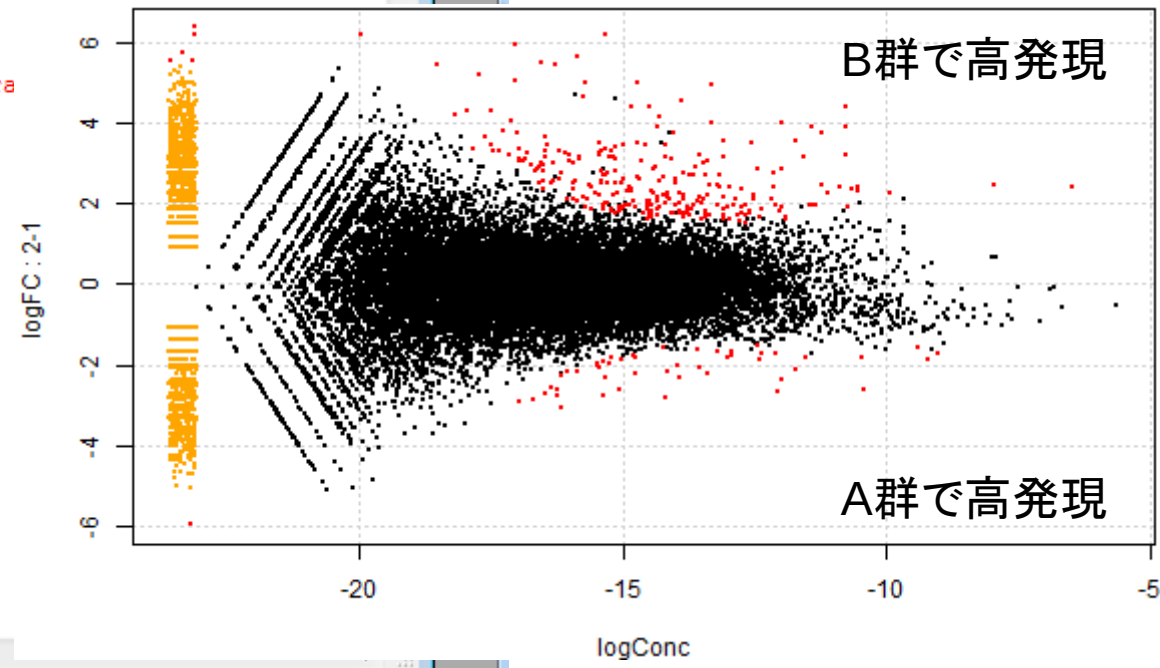
*edgeR*をdefaultの手順(edgeR/default)で実行

RGui (64-bit)

ファイル 編集 閲覧 その他 パッケージ ウインドウ ヘルプ Vignettes

R Console

```
> out <- exactTest(d)                                    #exact tes$
Comparison of groups:  2 - 1
> fdr <- p.adjust(out$table$p.value, method="BH")
> rank_edgeR <- rank(fdr)
> hoge <- cbind(rownames(data), data, out$table, fdr, ra
> tmp <- hoge[order(rank_edgeR),]
> write.table(tmp, out_f1, sep="\t", append=F, quote=F,
>
> #MA-plotを描画
> png(out_f2, width=param4[1], height=param4[2])
> obj <- rownames(data)[fdr < param3]
> plotSmear(d, de.tags=obj)
> dev.off()
null device
          1
> length(obj)
[1] 318
> length(obj)/nrow(data)
[1] 0.01212768
>
>
```



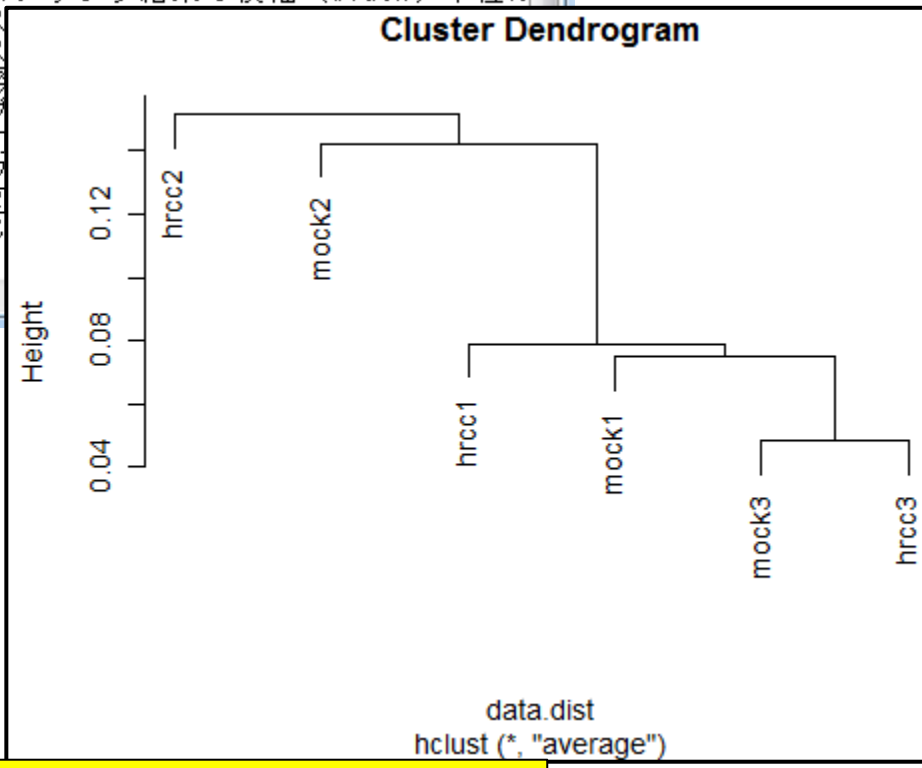
サンプル間クラスタリングも重要です

サンプル間クラスタリングも重要です

Rcode_clustering.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f <- "data_arab.txt"                                    #入力ファイル名(発現データファイル)を指
param2 <- "average"                                        #方法(method)を指定
out_f <- "result_cluster.png"                              #出力ファイル名(クラスタリング結果ファイ
param3 <- 500                                              #クラスタリング結果の横幅 (width; 単位は
param4 <- 400                                              #クラス
param5 <- 14                                               #クラス
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#
data.dist <- as.dist(1 - cor(data, method="spearman"))     #サンプ
out <- hclust(data.dist, method=param2)                    #階層的
png(out_f, pointsize=param5, width=param3, height=param4)  #出力フ
plot(out)                                                  #樹形図
dev.off()                                                  #おまじ
```



データ中に発現変動遺伝子がありそうかどうかは
クラスタリング結果を眺めるだけでかなりわかる

東大生以外の方も受講可能です（来年度もやります）

謝辞

共同研究者

清水 謙多郎　先生（東京大学）
嶋田 透　先生（東京大学）
西山 智明　先生（金沢大学）
勝間 進　先生（東京大学）
河岡 慎平　博士（東京大学）
末次 克行　先生（農業生物資源研究所）
上樂 明也　先生（農業生物資源研究所）

グラント

- 若手研究(B)（H21-23年度）:「マイクロアレイ解析の再現性・感度・特異度を飛躍的に向上させるデータ解析手法の開発」(代表)
- 新学術領域研究(研究領域提案型)(H22年度-):「非モデル生物におけるゲノム解析法の確立」(分担；研究代表者：西山智明)