

# RNA-Seqによる 大量発現データ解析の基礎・応用

農研機構 基盤技術研究本部

高度分析研究センター

ゲノム情報大規模解析ユニット

川原 善浩



y.kawahara@affrc.go.jp

9時00分-12時00分

「RNA-Seqによる大量発現データ解析の基礎」



13時00分-15時00分

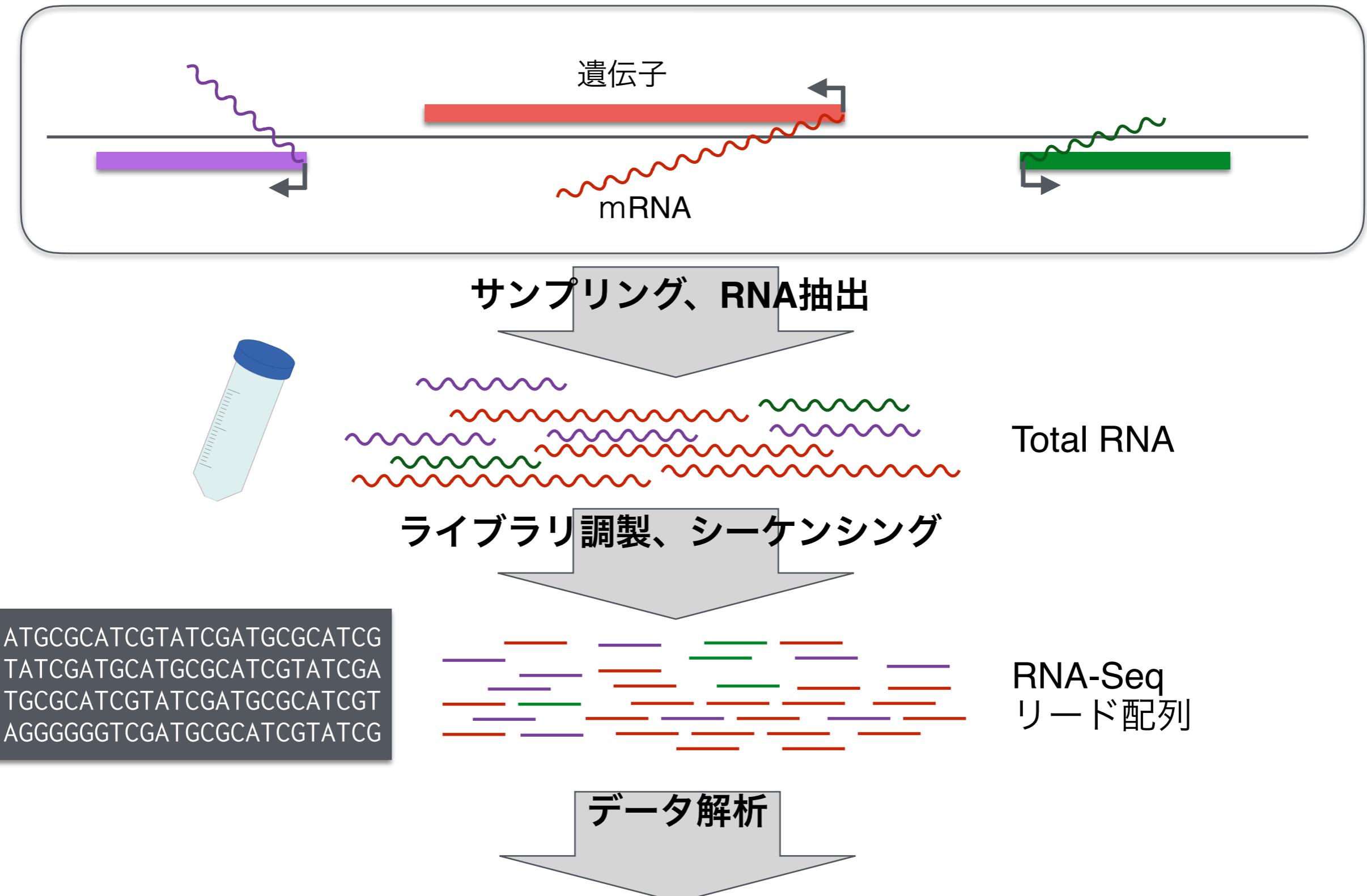
「RNA-Seqによる大量発現データ解析の応用」

適当に質問タイム、休憩を挟みながら・・・

- リファレンス情報を用いたRNA-Seq解析
- RNA-Seqデータを用いた多型検出

の演習をおこないます。

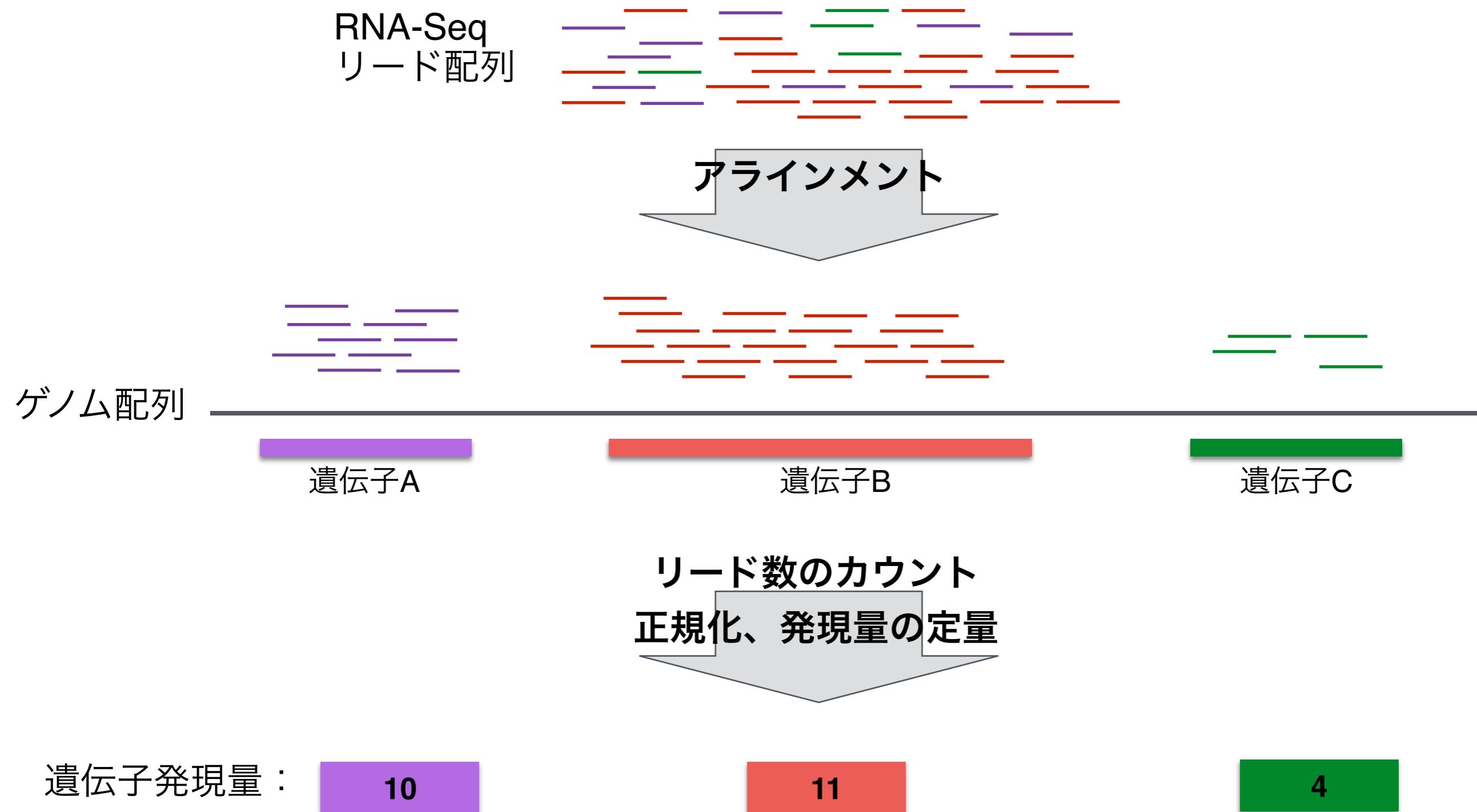
# RNA-Seqとは



# 手法1：ゲノム配列や遺伝子アノテーションを用いた遺伝子発現解析



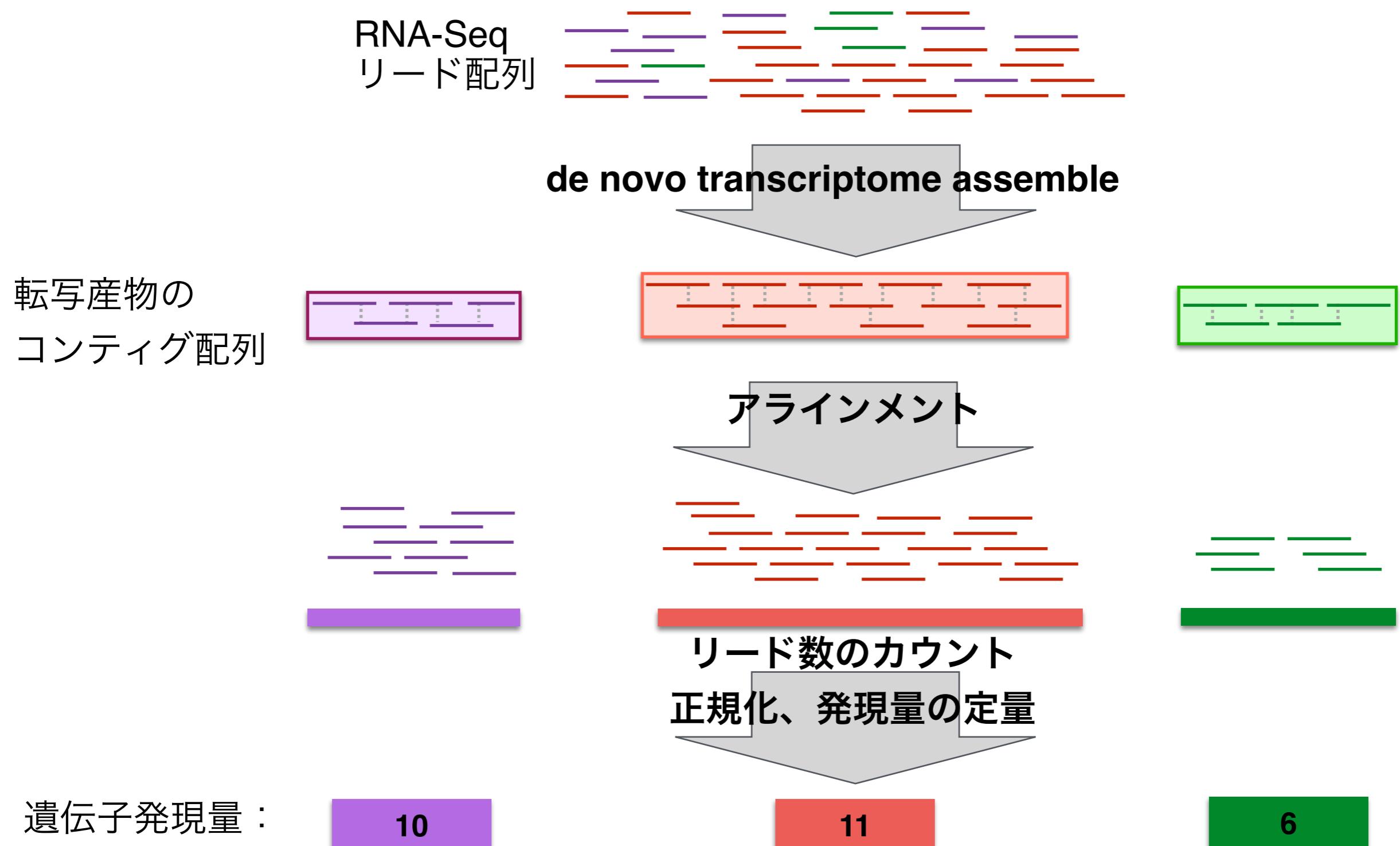
RNA-Seqリードをゲノム配列にアラインメント（マッピング）し、  
遺伝子ごとにマップされたリードをカウントし、遺伝子発現を定量する。



## 手法2：de novo transcriptome assemble法による遺伝子発現解析

まず始めに、RNA-Seqリード配列から転写産物配列を再構築する。

アセンブルされた転写産物配列上にRNA-Seqリードをアラインメントし、転写産物ごとにリード数をカウントして遺伝子発現を定量する。



過去の講義資料を公開しています

[https://github.com/YoshiKawahara/NGS\\_HandsOnWorkshop](https://github.com/YoshiKawahara/NGS_HandsOnWorkshop)

YoshiKawahara / NGS\_HandsOnWorkshop Public

Code Issues Pull requests Actions Projects Security Insights

master NGS\_HandsOnWorkshop / RNA-Seq\_analysis\_2019 / NGS\_Workshop\_RNA-Seq\_2019\_public.pdf

YoshiKawahara Add files via upload Latest commit f2d2bb1 on Sep 25, 2019 History

1 contributor

9.82 MB Download

 農研機構

第221回農林交流センターワークショップ  
「次世代シーケンサーのデータ解析技術」  
2019年9月26日（木）@電農館

# RNA-Seqによる 大量発現データ解析の基礎・応用

国立研究開発法人 農業・食品産業技術総合研究機構  
次世代作物開発研究センター  
(併任：高度解析センター)

川原 善浩

 [y.kawahara@affrc.go.jp](mailto:y.kawahara@affrc.go.jp)  [@YoshiKawahara](https://twitter.com/YoshiKawahara)

## 本日の流れ

9時00分-12時00分

「RNA-Seqによる大量発現データ解析の基礎」



13時00分-15時00分

「RNA-Seqによる大量発現データ解析の応用」

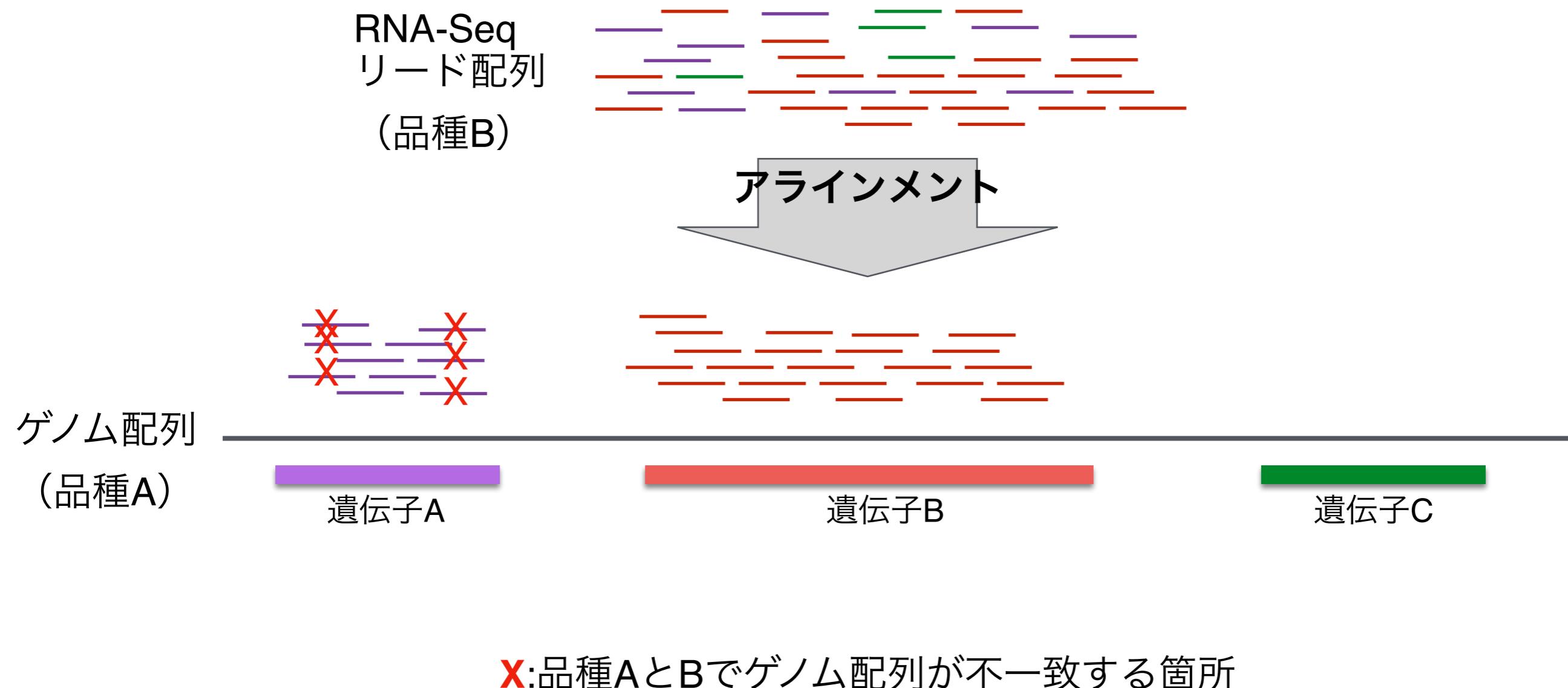
適時に質問タイム、休憩を挟みながら・・・

- リファレンス情報を用いたRNA-Seq解析
- De novoトランスクリプトーム解析
- RNA-Seqデータを用いた多型検出

の演習をおこないます。

# RNA-Seqデータからは発現量以外にも得られる情報はある

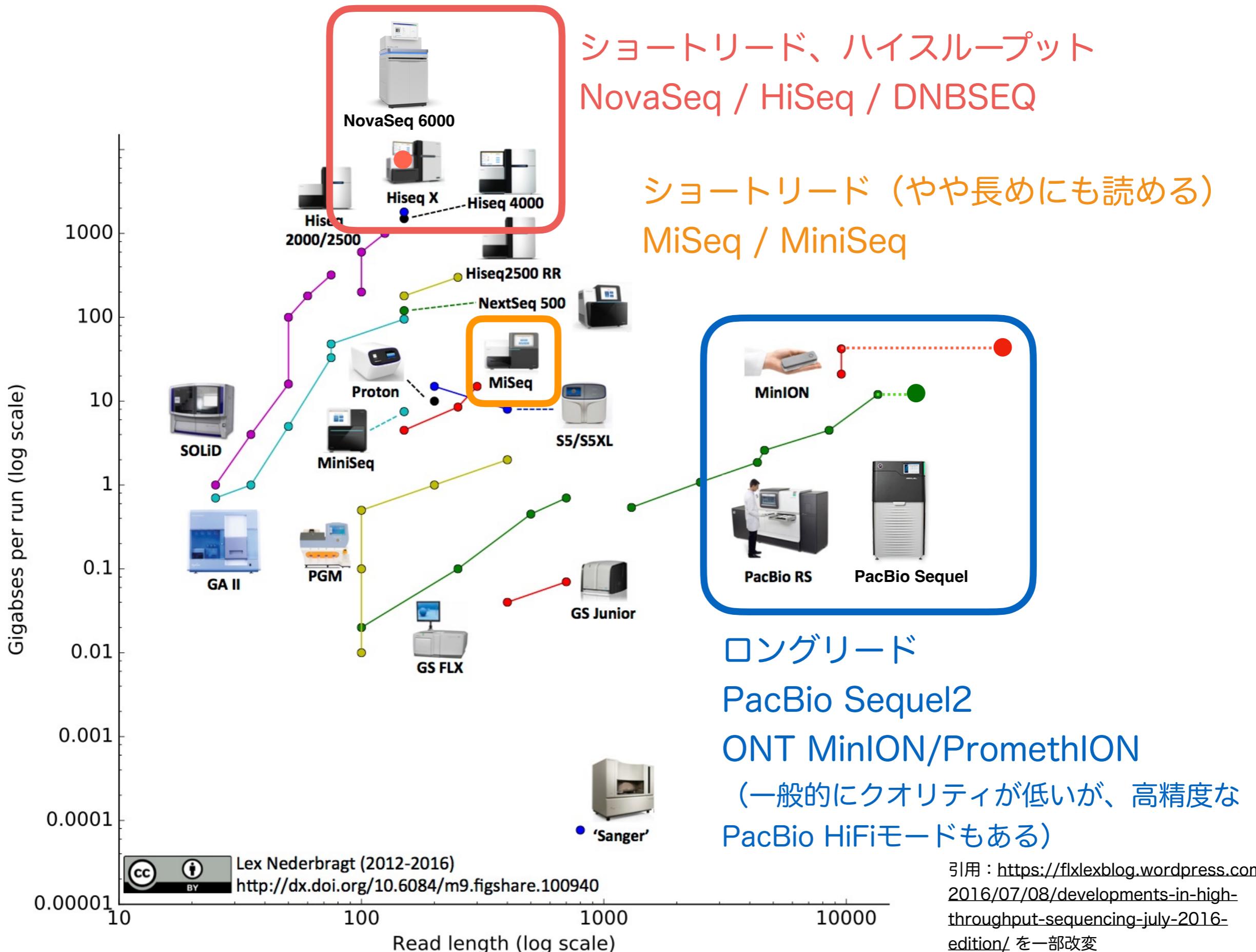
リファレンスとは異なる種や品種由来のRNA-Seqリードをリファレンスゲノム配列にアラインメントし、ゲノムとリード配列の違い（種間や品種間の転写領域上の多型情報）を得る。



目的やデータ解析手法を考え、  
適切なライブラリ調製やシークエンシング方法を  
選びましょう

「データが出てから考える」  
では手遅れ！（ということもある）

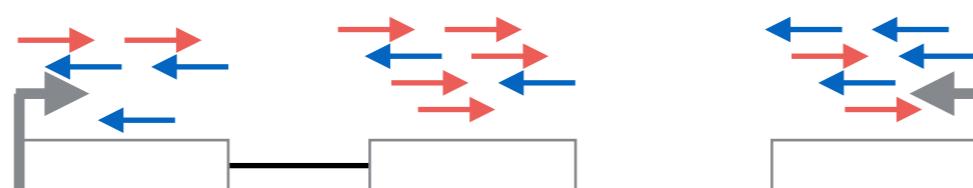
# シーケンサーの種類はいろいろある



# ライブラリ調製やシークエンシング方法にもいろいろある

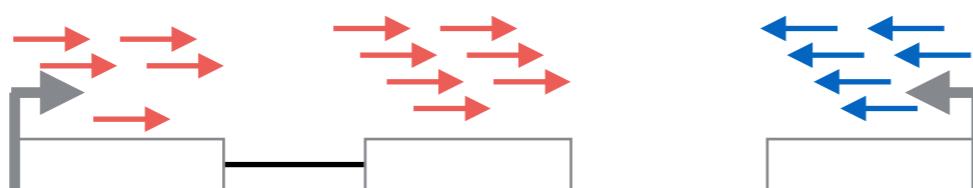


普通の (standard) RNA-Seq



転写の向きとリードの向きは無関係

strand-specific (stranded) RNA-Seq



転写の向きとリードの向きに対応関係があり、アセンブルや発現解析の精度が向上する。

cDNA断片の片側だけを読むか両端を読むかの違い。ペアリード間の距離 (cDNA断片長) の情報を用いることで、遺伝子構造予測やアセンブル精度が向上する。

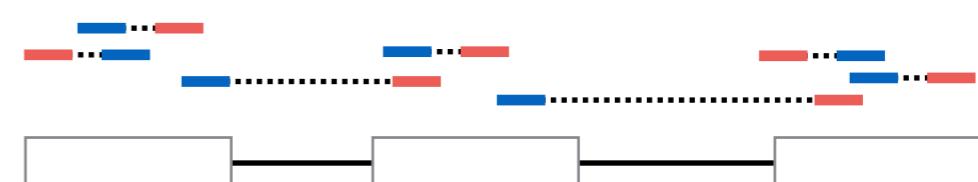
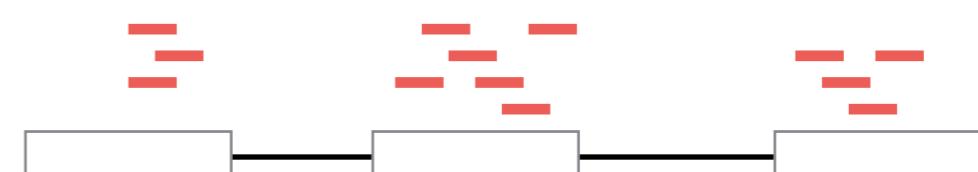
シークエンシング

single read  
(SR/SE)



アラインメント

paired-end read  
(PE)



# 目的にあったシークエンシング手法やライブラリ調製法を選ぶ



リファレンス情報を用いた  
遺伝子発現解析

- ・リファレンス情報なし  
の遺伝子発現解析
- ・多型検出

遺伝子カタログ作成

ゲノムや遺伝子情報は充分  
とにかく発現量が知りたい

とにかく安価に多くの  
リードを読む

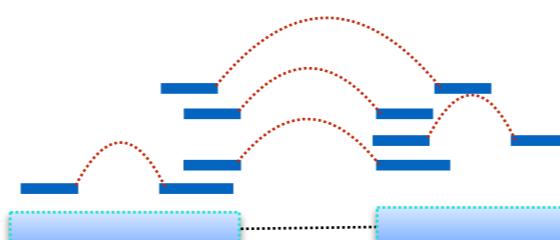
HiSeq, single-read, 50bp



遺伝子構造も発現量も  
どちらも知りたい

リード数と転写構造の情  
報をバランスよく得る

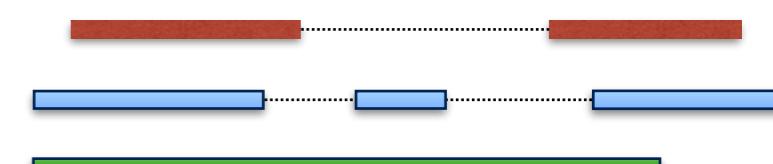
HiSeq, paired-end, 100/150bp



発現量よりもまずはどん  
な遺伝子が発現している  
か知りたい

リード数は少ないが、  
とにかく長いリードで  
転写産物の全長配列を得る

PacBio, ONT



- ▶ リファレンス情報を使ったRNA-Seq解析
- ▶ RNA-Seqデータを用いた多型検出

# 演習用データの準備

以下の要領で演習用データ（/work/NGSworkshop\_22/RNA-Seq.tar.gz）を各自のホームディレクトリにコピー、解凍、展開して中身を確認する。アカウント名（guest20）は各人で異なります。

```
$ cd ..... ホームディレクトリへ移動
$ pwd ..... カレントディレクトリを表示
/home/guest20
$ cp /work/NGSworkshop_22/RNA-Seq.tar.gz ./ ..... 解析用データをカレントディレクトリにコピー
$ ls -lh
...
-rw-rw-r-- 1 guest20 guest20 237M 9月 3 14:17 RNA-Seq.tar.gz
...
$ tar xfz RNA-Seq.tar.gz ..... 解析用データを解凍、展開
$ ls -lh
...
drwxrwxr-x 6 guest20 guest20 105 9月 3 14:11 RNA-Seq
-rw-rw-r-- 1 guest20 guest20 237M 9月 3 14:17 RNA-Seq.tar.gz
...
$ ls RNA-Seq ..... 解析用ディレクトリの中身を確認
data  install_tools.sh  noaln  tool  useref  varcall
```

## RNA-Seqディレクトリ中の各ファイル/ディレクトリの説明

- data : 解析に用いるデータ（ゲノム配列、遺伝子アノテーション、RNA-Seqリードなど）
- noaln : アラインメントフリーのRNA-Seq解析用のディレクトリ
- useref : リファレンス情報を用いたRNA-Seq解析用のディレクトリ
- varcall : RNA-Seqデータを用いた多型検出解析用のディレクトリ
- tool : インストールされた解析ツール群が置かれたディレクトリ
- install\_tools.sh : 解析ツール群をインストールするためのシェルスクリプト **(事前に実行済み)**

## リファレンスゲノム情報を用いたRNA-Seq解析

1. FastQC
2. Trimmomatic
3. HISAT2
4. StringTie
5. Samtools
6. Kallisto

## データの可視化、発現変動解析

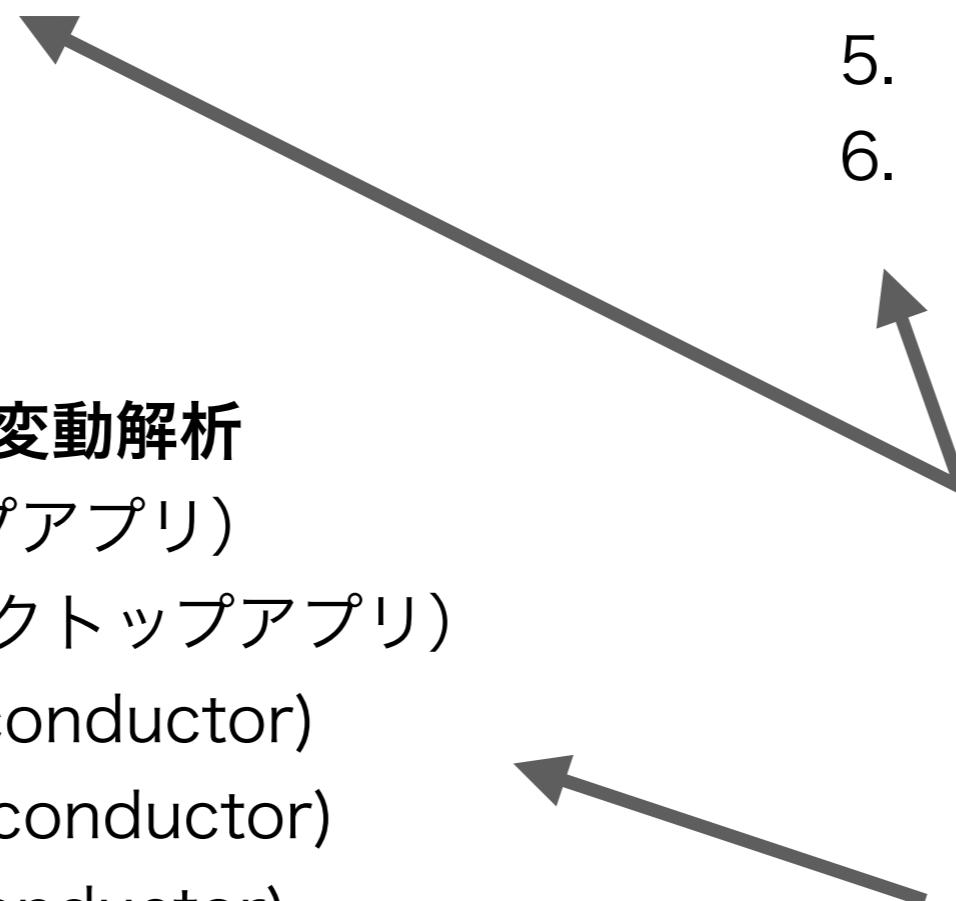
1. IGV (デスクトップアプリ)
2. R/RStudio (デスクトップアプリ)
3. Ballgown (R/Bioconductor)
4. genefilter (R/Bioconductor)
5. DESeq2 (R/Bioconductor)
6. tidyverse (R)

## RNA-Seqデータを用いた多型検出

1. FastQC
2. Trimmomatic
3. Samtools
4. STAR2
5. Picard
6. GATK

解析サーバ (Linux) に  
インストール

ローカル環境 (Win/Mac)  
にインストール

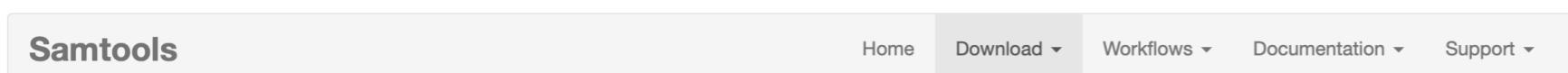


# 解析ツールの取得、インストール

ウェブサイトから最新版のツールのバイナリ、またはソースファイルをダウンロード、インストールする（資料末尾に各ツールのウェブサイト等の情報あり）。

biocondaやhomebrew（Mac）などのパッケージマネージャ、コンテナ型仮想環境「Docker」を用いてインストールできるものもある。

## Samtoolsの場合 (<https://www.htslib.org/download/>)



### Current releases

SAMtools and BCFtools are distributed as individual packages. The code uses HTSlib internally, but these source packages contain their own copies of HTSlib so they can be built independently.

HTSlib is also distributed as a separate package which can be installed if you are writing your own programs against the HTSlib API. HTSlib also provides the **bgzip**, **htsfile**, and **tabix** utilities, so you may also want to build and install HTSlib to get these utilities, or see the additional instructions in [INSTALL](#) to install them from a samtools or bcf tools source package.

Download current source releases: [!\[\]\(8706f9f9febc74216a91030d11f10ce7\_img.jpg\) samtools-1.13](#) [!\[\]\(989df8a2617dc59cec0d761ee1f5d011\_img.jpg\) bcf tools-1.13](#) [!\[\]\(46944fcbcb2f2fc5190c2123640734aa\_img.jpg\) htslib-1.13](#)

See also release notes for [samtools](#), [bcf tools](#), and [htslib](#).

New releases are announced on the [samtools mailing lists](#) and by [@htslib](#) on Twitter. Previous releases are available from the [samtools GitHub organisation](#) (see [samtools](#), [bcf tools](#), or [htslib](#) releases) or from the [samtools Sourceforge project](#).

**インストール方法の説明**

### Building and installing

Building each desired package from source is very simple:

```
cd samtools-1.x    # and similarly for bcf tools and htslib
./configure --prefix=/where/to/install
make
make install
```

See [INSTALL](#) in each of the source directories for further details.

The executable programs will be installed to a **bin** subdirectory under your specified prefix, so you may wish to add this directory to your \$PATH:

```
export PATH=/where/to/install/bin:$PATH    # for sh or bash users
```

```
setenv PATH /where/to/install/bin:$PATH    # for csh users
```

# 解析ツールのダウンロードとインストール

全解析ツールをまとめてインストールするためのシェルスクリプト

```
$ less install_tools.sh
```

- install\_tools.shの先頭部分 -

- less、more、catなどのコマンドをつかってスクリプトの中身を見ることができる。
- lessやmoreを終了するためには「q」キーを打つ。

```
mkdir tool                                ツール類を置くためのディレクトリの作成と移動
cd tool
ToolDir=`pwd`                               ツールディレクトリのパスの取得

echo "START:" `date`                      ツールインストールの開始時間を出力
### Tools for "useref" analysis           #で始まるのはコメント行
echo "installing FastQC..."               FastQCのダウンロード
wget https://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0.11.9.zip
unzip fastqc_v0.11.9.zip                  解凍・展開
chmod +x FastQC/fastqc                   fastqcプログラムに実行権限を与える
rm fastqc_v0.11.9.zip                    不要なダウンロードファイルは削除

echo "installing Trimmomatic..."          Trimmomaticのダウンロード
wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.39.zip
unzip Trimmomatic-0.39.zip                解凍・展開
rm Trimmomatic-0.39.zip                  不要なダウンロードファイルは削除
```

実行ファイル（バイナリファイル）やJavaプログラムのインストールは、  
基本的にダウンロード（wget や curl）して、解凍・展開（unzip や tar）するだけ。

# 解析ツールのダウンロードとインストール

## - install\_tools.shのつづき1 -

```
echo "installing HISAT2..."  
wget https://cloud.biohpc.swmed.edu/index.php/s/hisat2-220-Linux_x86_64/download \  
-O hisat2-2.2.0-Linux_x86_64.zip  
unzip hisat2-2.2.0-Linux_x86_64.zip ..... 解凍・展開  
rm hisat2-2.2.0-Linux_x86_64.zip ..... 不要なダウンロードファイルは削除  
  
echo "installing StringTie..."  
wget http://ccb.jhu.edu/software/stringtie/dl/stringtie-2.1.6.Linux_x86_64.tar.gz  
tar xfz stringtie-2.1.6.Linux_x86_64.tar.gz ..... 解凍・展開  
rm stringtie-2.1.6.Linux_x86_64.tar.gz ..... 不要なダウンロードファイルは削除
```

\ (バックスラッシュ) は実行時には無視され、ひと続きのコマンドとして実行される。可読性を上げるために使用。

HISAT2のダウンロード

StringTieのダウンロード

- \*.tar.gz や \*.tar.bz2 の解凍・展開は、tarコマンドにオプションをつけて実行する (\*.tar.gz の場合は 「tar xfz」 、 \*.tar.bz2 の場合は 「tar xfj」 ) 。
- HISAT2の最新版 (hisat2-2.2.1) は演習で使うサーバ環境ではうまく動作しなかったため、1つ古いバージョン (hisat2-2.2.0) を使用します。

# 解析ツールのダウンロードとインストール



- install\_tools.shのつづき2 -

```
echo "installing Samtools..."  
wget https://github.com/samtools/samtools/releases/download/1.13/samtools-1.13.tar.bz2  
tar xfj samtools-1.13.tar.bz2 ..... 解凍・展開  
Samtoolsのダウンロード  
rm samtools-1.13.tar.bz2 ..... 不要なダウンロードファイルは削除  
cd samtools-1.13 ..... Samtoolsディレクトリに移動  
make ..... コンパイル  
make prefix=`pwd` install ..... prefix (インストール先) を指定してインストール  
cd .. ..... toolディレクトリに戻る  
  
echo "installing Kallisto..."  
wget https://github.com/pachterlab/kallisto/releases/download/v0.46.1/kallisto_linux- \  
v0.46.1.tar.gz  
tar xfz kallisto_linux-v0.46.1.tar.gz ..... 解凍・展開  
Kallistoのダウンロード  
rm kallisto_linux-v0.46.1.tar.gz ..... 不要なダウンロードファイルは削除  
  
echo "END:" `date` ..... ツールインストールの終了時間を出力
```

- Samtoolsについてはソースファイルをダウンロードし、コンパイル (make) 、インストール (make install) を行なう。

# 解析ツールのダウンロードとインストール



## - install\_tools.shのつづき3 -

```
### Tools for "varcall" analysis
echo "installing STAR..."
wget https://github.com/alexdobin/STAR/releases/download/2.7.10b/STAR_2.7.10b.zip
unzip STAR_2.7.10b.zip
rm STAR_2.7.10b.zip

echo "installing Picard..."
mkdir picard-2.27.5
cd picard-2.27.5/
wget https://github.com/broadinstitute/picard/releases/download/2.27.5/picard.jar
cd ..

echo "installing GATK..."
wget https://github.com/broadinstitute/gatk/releases/download/4.2.6.1/gatk-4.2.6.1.zip
unzip gatk-4.2.6.1.zip
rm gatk-4.2.6.1.zip

echo "END:" `date`
```

- ・バイナリファイルをダウンロード、展開、不要なファイルの削除

# 解析ツールのダウンロードとインストール

事前に実行済みですので、演習中は実行しないでください

## ツールのインストールをするためのシェルスクリプトの実行 (実行時間：約2分)

- 普通にbashコマンドで実行すると、スクリプトの標準出力と標準エラー出力をターミナル画面に表示する。

```
$ bash ./install_tools.sh
```

- 以下のようなコマンドを実行すると、標準出力と標準エラー出力を合わせてターミナル画面に表示しながら、ファイルにも書き出してくれる。こうしておくとあとでログを見直すことができて便利。

```
$ bash ./install_tools.sh 2>&1 | tee install_tools.log
```

- 一連のコマンドが書かれたシェルスクリプトを作成すると、コマンドを連続して実行することができる。スクリプトとして残しておくと、あとで実行コマンドを確認する際にも役立つ。
- ツールによっては、解析結果の統計情報などを標準出力や標準エラー出力として出力するので、それらもファイルに保存しておくとよい。

# インストールされた解析ツールを確認

全てのツールはtoolディレクトリにインストールされる。

```
$ ls tool
FastQC          Trimmomatic-0.39  hisat2-2.2.0  picard-2.27.5  stringtie-2.1.6.Linux_x86_64
STAR_2.7.10b    gatk-4.2.6.1     kallisto      samtools-1.13
```

例えば、HISAT2は「tool/hisat2-2.2.0」以下にインストールされている。

```
$ ls tool/hisat2-2.2.0
AUTHORS          hisat2-align-s-debug        hisat2_extract_snps_haplotypes_UCSC.py
LICENSE          hisat2-build              hisat2_extract_snps_haplotypes_VCF.py
MANUAL           hisat2-build-l            hisat2_extract_splice_sites.py
MANUAL.markdown  hisat2-build-l-debug       hisat2_read_statistics.py
NEWS             hisat2-build-s            hisat2_simulate_reads.py
TUTORIAL         hisat2-build-s-debug      hisatgenotype.py
VERSION          hisat2-inspect            hisatgenotype_build_genome.py
example          hisat2-inspect-l          hisatgenotype_extract_reads.py
extract_exons.py hisat2-inspect-l-debug    hisatgenotype_extract_vars.py
extract_splice_sites.py hisat2-inspect-s      hisatgenotype_hla_cyp.py
hisat2           hisat2-inspect-s-debug    hisatgenotype_locus.py
hisat2-align-l   hisat2-repeat             hisatgenotype_modules
hisat2-align-l-debug hisat2-repeat-debug     hisatgenotype_scripts
hisat2-align-s   hisat2_extract_exons.py  scripts
```

- 様々なドキュメントやプログラム、スクリプトが存在している。
- ターミナルによっては実行形式のファイルやディレクトリ名等に色がついていて、見やすくなっていることがあるが、設定によって異なる。

# インストールされた解析ツールを確認

多くのツールは「`--help`」や「`-h`」オプションをつけて実行することで簡単な使い方や指定できるパラメータなどを確認できる。

```
$ ./tool/hisat2-2.2.0/hisat2 -h ..... -hオプションをつけてHISAT2を実行
HISAT2 version 2.2.0 by Daehwan Kim (infphilo@gmail.com, wwwccb.jhu.edu/people/infphilo)
Usage:
hisat2 [options]* -x <ht2-idx> {-1 <m1> -2 <m2> | -U <r>} [-S <sam>]

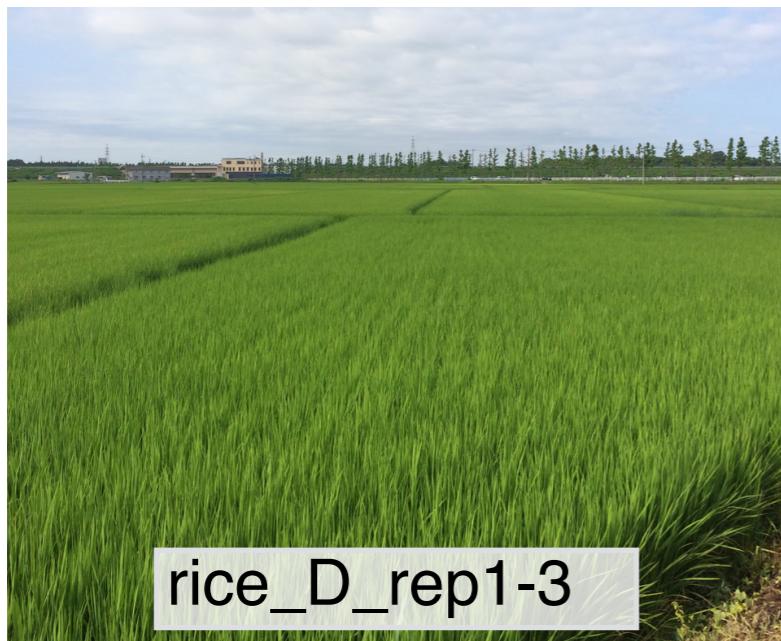
<ht2-idx> Index filename prefix (minus trailing .X.ht2).
<m1> Files with #1 mates, paired with files in <m2>.
      Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<m2> Files with #2 mates, paired with files in <m1>.
      Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<r> Files with unpaired reads.
      Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
<sam> File for SAM output (default: stdout)

<m1>, <m2>, <r> can be comma-separated lists (no whitespace) and can be
specified many times. E.g. '-U file1.fq,file2.fq -U file3.fq'.
| (省略)
--version          print version information and quit
-h/--help         print this usage message
```

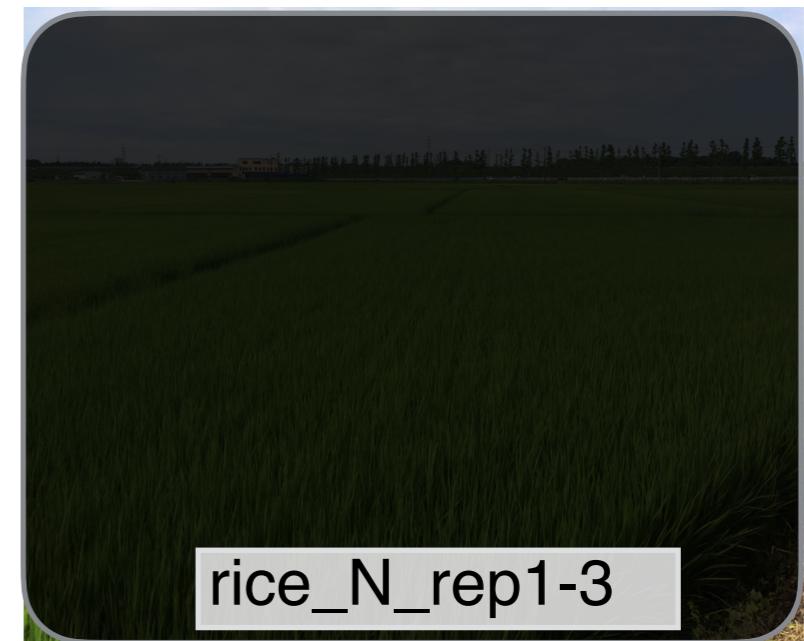
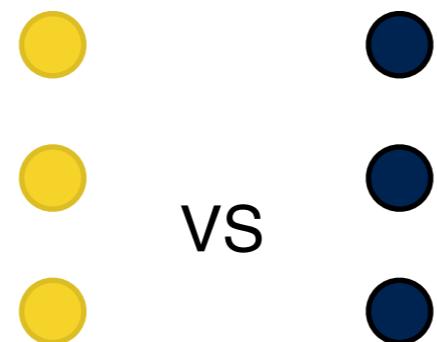
ここでエラーが出なければ、正しくインストールされている（たぶん）

# 目的と使用するデータ

目的：リファレンス情報を用いて、  
昼と夜でのイネの葉の遺伝子発現の違いを調べる



12:00 PM



00:00 AM

- 田んぼで栽培する、イネ（コシヒカリ）の葉身のサンプル。
- 3つの異なる生育ステージ（**3反復**）において、昼（12時）と夜（0時）にサンプリング（**2条件**）。
- Illumina社の**Stranded mRNA-Seq法**でライブラリ調製。
- Illumina社のHiSeq2000による、**101bpのPaired-endシーケンシング**。

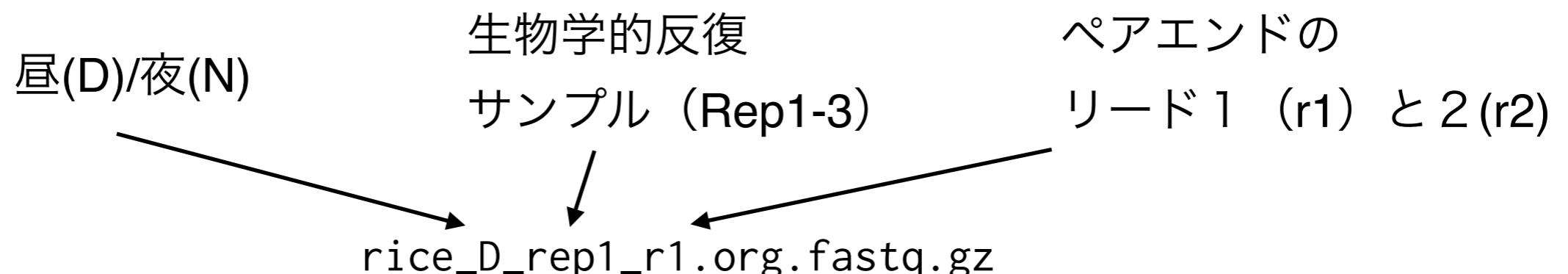
\*全データでは解析に時間がかかるため、2番染色体の特定の領域（2Mbp）にアラインメントされるリードだけを事前に抽出しており、演習ではそれを利用する。

# 目的と使用するデータ

```
$ cd ..... ホームディレクトリに移動
$ cd RNA-Seq ..... RNA-Seqディレクトリに移動
$ ls data ..... dataディレクトリの中身を確認
annotation.gtf
genome.fa
rice_D_rep1_r1.org.fastq.gz
rice_D_rep1_r2.org.fastq.gz
rice_D_rep2_r1.org.fastq.gz
rice_D_rep2_r2.org.fastq.gz
rice_D_rep3_r1.org.fastq.gz
rice_D_rep3_r2.org.fastq.gz
rice_N_rep1_r1.org.fastq.gz
rice_N_rep1_r2.org.fastq.gz
rice_N_rep2_r1.org.fastq.gz
rice_N_rep2_r2.org.fastq.gz
rice_N_rep3_r1.org.fastq.gz
rice_N_rep3_r2.org.fastq.gz
transcript.fasta
```

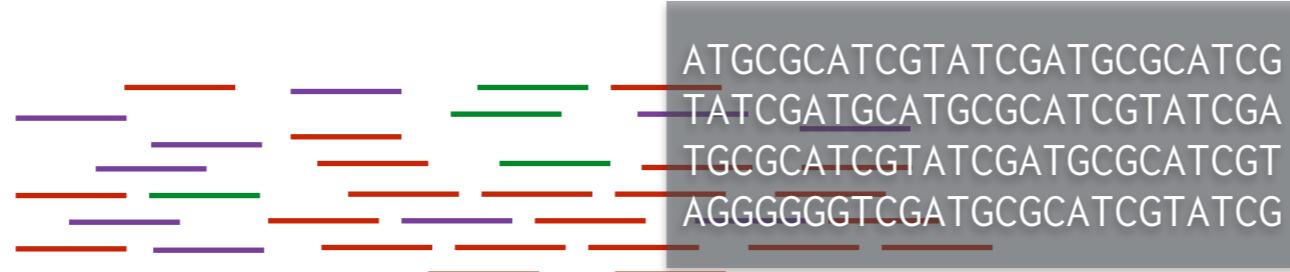
## dataディレクトリ中の各ファイルの説明

- ・ リファレンスゲノム配列 (genome.fa)
- ・ 遺伝子アノテーションファイル (annotation.gtf)
- ・ イネ遺伝子の塩基配列 (transcript.fasta)
- ・ mRNA-Seqリード配列 (\*.fastq.gz)



# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

アラインメント

HISAT2

リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

統計解析、可視化

IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

# リファレンス情報を用いたRNA-Seq解析のためのツール

**HISAT2**  
graph-based alignment of next generation sequencing reads to a population of genomes

JOHNS HOPKINS UNIVERSITY  
CENTER FOR COMPUTATIONAL BIOLOGY  
CCB

HISAT2 is a fast and sensitive alignment program for mapping next-generation sequencing reads (both DNA and RNA) to a population of human genomes (as well as against a single reference genome). Based on an extension of BWT for graphs [Sirén et al. 2014], we designed and implemented a graph FM index (GFM), an original approach and its first implementation to the best of our knowledge. In addition to using one global GFM index that represents a population of human genomes, HISAT2 also supports multiple local GFM indexes (called LGMs) for each transcript, which greatly speeds up the search process.

**HISAT2**  
splice-awareなリードアラインメント

HISAT2 2.0.4 Windows binary available [here](#), thanks to André Osório Falcão  
5/24/2016

HISAT2 2.0.4 release 5/18/2016

Version 2.0.4 is a minor release with the following changes.  
◦ Improved template length estimation (the 9th column of the SAM format) of RNA-seq

<https://ccb.jhu.edu/software/hisat2/index.shtml>

**StringTie**  
Transcript assembly and quantification for RNA-Seq

JOHNS HOPKINS UNIVERSITY  
CENTER FOR COMPUTATIONAL BIOLOGY  
CCB

Home Manual Examples CCB » Software » StringTie

▪ Overview  
▪ News  
▪ Obtaining and installing  
▪ Licensing and contacting  
▪ Publications

**StringTie**  
発現量の定量  
遺伝子構造のアセンブル

StringTie is a fast and highly accurate transcript assembly and quantification algorithm as well as an optional *de novo* assembly step to assemble and quantitate full-length transcripts representing multiple splice variants for each gene locus. Its input can include not only the alignments of raw reads used by other transcript assemblers, but also alignments longer sequences that have been assembled from those reads. In order to identify differentially expressed genes between experiments, StringTie's output can be processed by specialized software like Ballgown, Cuffdiff or other programs (DESeq2, edgeR, etc.).

a novel network flow

<https://ccb.jhu.edu/software/stringtie/>

## Tuxedo suite tools

(TopHat, Cufflinksの後継プログラム)

Home » Bioconductor 3.9 » Software Packages » ballgown

**ballgown**

platforms all rank 131 / 1741 post 0 / 0 / in Bioc 5 years  
build ok updated before release dependencies 6 8

DOI: [10.18129/B9.bioc.ballgown](https://doi.org/10.18129/B9.bioc.ballgown) [f](#) [t](#)

Flexible, isoform-level differential expression analysis

Bioconductor version: Release 3.9  
Tools for statistical analysis and visualization of transcript abundance data, including differential expression analysis, and gene annotation.

**Ballgown**  
発現解析と可視化

Author: Jack Fu [aut], Alyssa C. Frazee [aut, cre], Leonardo Collado-Torres [aut], Andrew E. Jaffe [aut], Jeffrey T. Leek [aut, ths]

Maintainer: Jack Fu <jmfu at jhsph.edu>

Citation (from within R, enter `citation("ballgown")`):

Fu J, Frazee AC, Collado-Torres L, Jaffe AE, Leek JT (2019). *ballgown: Flexible, isoform-level differential expression analysis*. R package version 2.16.0.

<http://bioconductor.org/packages/release/bioc/html/ballgown.html>

# Tuxedo suite toolsによるRNA-Seq解析プロトコル

Pertea et al. *Nature Protocol* 2016 11(9)

## PROTOCOL

### Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown

Mihaela Pertea<sup>1,2</sup>, Daehwan Kim<sup>1</sup>, Geo M Pertea<sup>1</sup>, Jeffrey T Leek<sup>3</sup> & Steven L Salzberg<sup>1–4</sup>

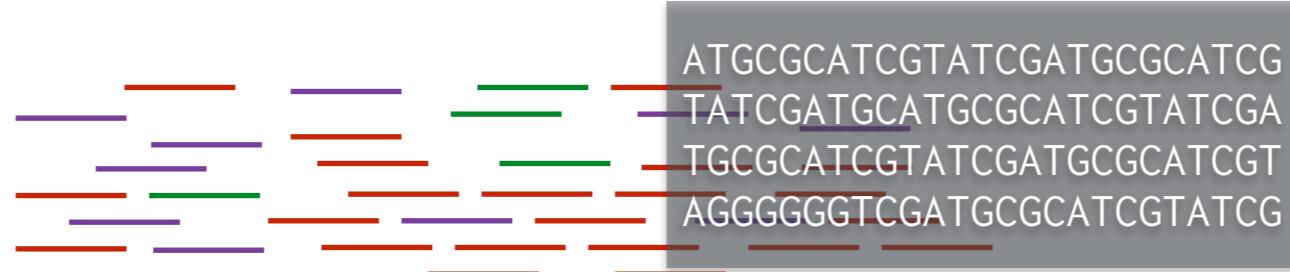
<sup>1</sup>Center for Computational Biology, McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins School of Medicine, Baltimore, Maryland, USA. <sup>2</sup>Department of Computer Science, Whiting School of Engineering, Johns Hopkins University, Baltimore, Maryland, USA. <sup>3</sup>Department of Biostatistics, Bloomberg School of Public Health, Johns Hopkins University, Baltimore, Maryland, USA. <sup>4</sup>Department of Biomedical Engineering, Johns Hopkins University, Baltimore, Maryland, USA.  
Correspondence should be addressed to S.L.S. ([salzberg@jhu.edu](mailto:salzberg@jhu.edu)).

- 6年前に公開されたプロトコルのため、最新版のツールではオプション名等が異なる部分もある。
- ツールは更新され続けているため、論文や書籍の情報はどんどん古くなっていくので要注意。ウェブサイトの最新情報を必ず参照しましょう。

Figure 1 from Pertea et. al. Nat. Protoc. 2016 11(9)

# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

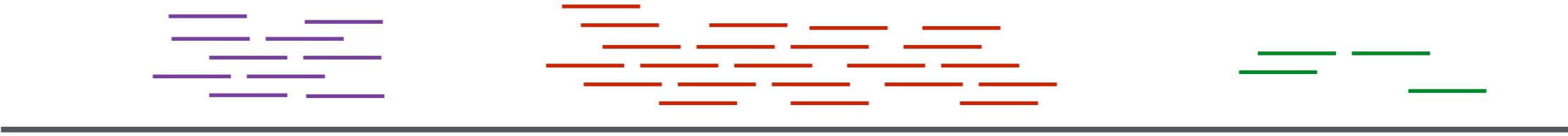
アラインメント

HISAT2

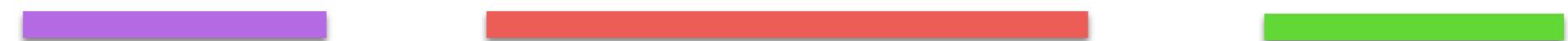
リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

統計解析、可視化

IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

# Step 1. 前処理 - 配列データのQC

悪いデータからは良い結果は得られない。

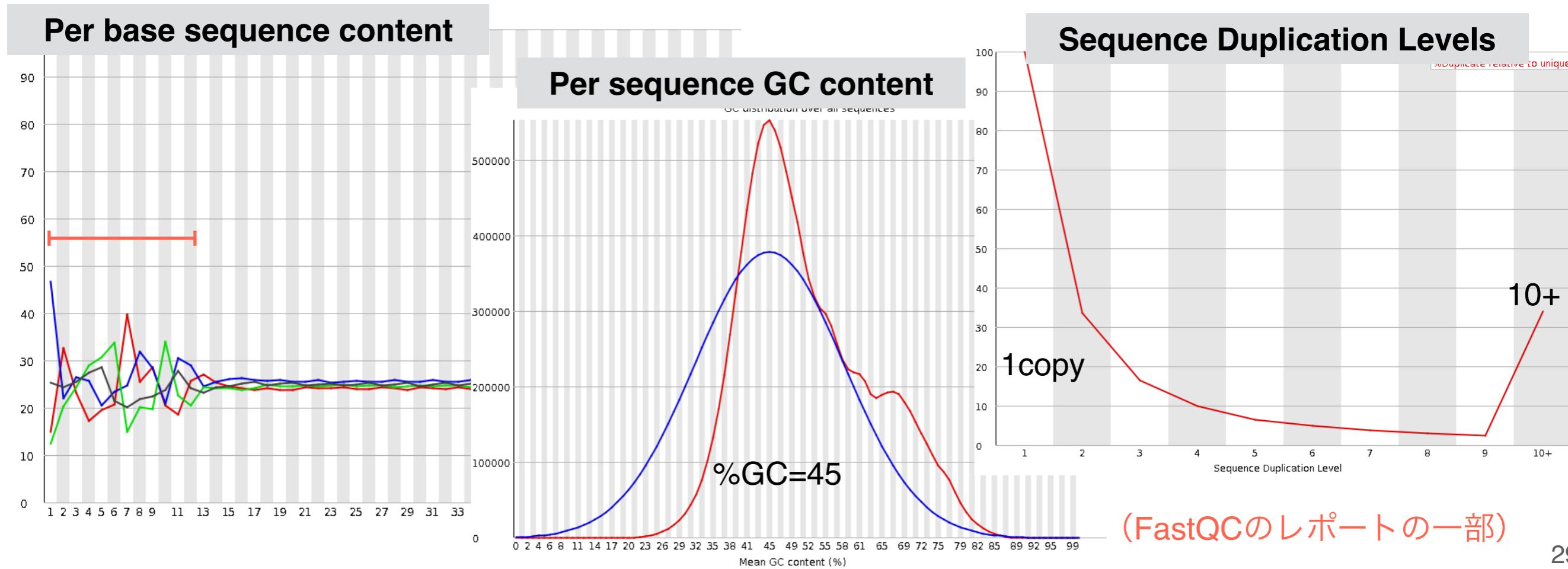
FastQCなどのQCツールを使い、リード数や解読塩基量、クオリティ、塩基出現頻度の偏り、配列長などを事前によくチェックする！

## RNA-Seqデータの特徴

- ・先頭部分の塩基頻度のバイアス
- ・GCバイアス
- ・高発現遺伝子によると思われる

Duplication levelの高さ

これらの特徴はゲノムシーケンスデータとはやや異なるので注意。



# Step 1. 前処理 - 配列データのクリーニング



## 1. 低クオリティ塩基やアダプター配列の除去

TrimmomaticやFASTX-Toolkitなどのツールを用いる。

(例) Trimmomaticで「ILLUMINACLIP:[adapter file]:2:30:10 LEADING:15 TRAILING:15 SLIDINGWINDOW:10:15 MINLEN:50」などと指定して実行する。

## 2. 実験で除ききれなかったrRNA由来のリードの除去 (本演習ではスキップ)

既知のrRNA配列ライブラリに対してBowtie2等でアラインメントし、そのアンマップリード（つまり、rRNA由来ではないリード）を以降の解析で用いる。

(例) bowtie2 -x [rRNA bowti2 index file] --un-conc-gz [unmap\_read.fastq]  
-1 [read1.fastq] -2 [read2.fastq] -S [mapped\_read.sam]

これらの処理は必須ではないが、塩基やリードの除去率やrRNAへのマップ率は、ライブラリ調製やシーケンシングに何か問題がないかを確認するための指標になる。

# Step 1. 前処理 - 配列データのQCとクリーニング

リファレンス情報を用いたRNA-Seq解析用ディレクトリ「useref」に移動

```
$ cd ~/RNA-Seq/useref
```

解析用ディレクトリ内のファイルを確認

```
$ ls
step1_preprocessing.sh  step4_StringTie-abundance_estimation.sh
step2_HISAT2-build.sh   step5_make_count_matrix.sh
step3_HISAT2.sh
```

シェルスクリプトが5つ用意されており、順番にシェルスクリプトを実行することで解析が進められるようになっている。

```
$ bash ./step1_preprocessing.sh
$ bash ./step2_HISAT2-build.sh
$ bash ./step3_HISAT2.sh
$ bash ./step4_StringTie-abundance_estimation.sh
$ bash ./step5_make_count_matrix.sh
```

演習では各ステップを説明しながら進めていきます。

# Step 1. 前処理 - 配列データのQCとクリーニング

リードの前処理をするためのシェルスクリプト

```
$ less ./step1_preprocessing.sh
```

- step1\_preprocessing.shの前半 -

```
DataDir=$HOME/RNA-Seq/data  
ToolDir=$HOME/RNA-Seq/tool  
FastQC_bin=$ToolDir/FastQC  
Trimmomatic_jar=$ToolDir/Trimmomatic-0.39/trimmomatic-0.39.jar  
Trimmomatic_adapters=$ToolDir/Trimmomatic-0.39/adapters
```

データや各ツールのコマンドが置かれているディレクトリを変数に格納

```
export PATH=$FastQC_bin:$PATH
```

exportコマンドで各ツールのディレクトリをPATHに追加。  
パスを指定しなくともコマンド実行が可能になる。

```
## Step1. Preprocessing by FastQC and Trimmomatic  
FASTQC_OUTDIR_BEFORE=FastQC_before_preprocess  
FASTQC_OUTDIR_AFTER=FastQC_after_preprocess  
mkdir $FASTQC_OUTDIR_BEFORE $FASTQC_OUTDIR_AFTER
```

FastQCの出力ディレクトリを作成

何度も出てくるパス等はスペルミスを避けたり、スクリプトの可読性を上げるために、変数に格納したり、exportコマンドでPATHに設定するとよい。

# Step 1. 前処理 - 配列データのQCとクリーニング

- step1\_preprocessing.shのつづき -

for関数による繰り返し。変数DATASETに  
in以降のサンプル名が順番に格納されて実行される。

```

for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_N_rep1 rice_N_rep2 rice_N_rep3
do ..... 繰り返し開始
    # FastQC before Trimmomatic ..... 前処理前のFastQCの実行 (r*でリード1と2を合せて指定)
    fastqc --threads 1 --nogroup --outdir $FASTQC_OUTDIR_BEFORE --format fastq
    $DataDir/${DATASET}_r*.org.fastq.gz
    # Trimmomatic ..... Trimmomaticの実行
    java -Xmx4G -Xms2G -jar $Trimmomatic_jar PE \
        -threads 1 -phred33 -trimlog Trimmomatic_${DATASET}.log \
        $DataDir/${DATASET}_r1.org.fastq.gz $DataDir/${DATASET}_r2.org.fastq.gz \
        ${DATASET}_r1.pe.fastq.gz ${DATASET}_r1.unpe.fastq.gz \
        ${DATASET}_r2.pe.fastq.gz ${DATASET}_r2.unpe.fastq.gz \
        ILLUMINACLIP:${Trimmomatic_adapters}/TruSeq3-PE-2.fa:2:30:10 \
        LEADING:15 TRAILING:15 SLIDINGWINDOW:10:15 MINLEN:50
    # FastQC after Trimmomatic ..... 前処理後のFastQCの実行
    fastqc --threads 1 --nogroup --outdir $FASTQC_OUTDIR_AFTER --format fastq ${DATASET}
    _r*.pe.fastq.gz
done ..... 繰り返し終了

```

サンプル数が6つあるため、「前処理前後のFastQCとTrimmomaticによる前処理」を各サンプルごとにfor文を使って繰り返し実行している。

# Step 1. 前処理 - 配列データのQCとクリーニング

シェルスクリプトの実行 (実行時間：約3分)

```
$ bash ./step1_preprocessing.sh 2>&1 | tee step1_preprocessing.log
```

出力ファイルの確認

```
$ ls
FastQC_after_preprocess      rice_D_rep2_r1.unpe.fastq.gz  rice_N_rep2_r2.pe.fastq.gz
FastQC_before_preprocess     rice_D_rep2_r2.pe.fastq.gz  rice_N_rep2_r2.unpe.fastq.gz
Trimmomatic_rice_D_rep1.log  rice_D_rep2_r2.unpe.fastq.gz rice_N_rep3_r1.pe.fastq.gz
Trimmomatic_rice_D_rep2.log  rice_D_rep3_r1.pe.fastq.gz  rice_N_rep3_r1.unpe.fastq.gz
Trimmomatic_rice_D_rep3.log  rice_D_rep3_r1.unpe.fastq.gz rice_N_rep3_r2.pe.fastq.gz
Trimmomatic_rice_N_rep1.log  rice_D_rep3_r2.pe.fastq.gz  rice_N_rep3_r2.unpe.fastq.gz
Trimmomatic_rice_N_rep2.log  rice_D_rep3_r2.unpe.fastq.gz step1_preprocessing.log
Trimmomatic_rice_N_rep3.log  rice_N_rep1_r1.pe.fastq.gz  step1_preprocessing.sh
rice_D_rep1_r1.pe.fastq.gz   rice_N_rep1_r1.unpe.fastq.gz step2_HISAT2-build.sh
rice_D_rep1_r1.unpe.fastq.gz rice_N_rep1_r2.pe.fastq.gz  step3_HISAT2.sh
rice_D_rep1_r2.pe.fastq.gz   rice_N_rep1_r2.unpe.fastq.gz step4_StringTie-abundance_estimation.sh
rice_D_rep1_r2.unpe.fastq.gz rice_N_rep2_r1.pe.fastq.gz  step5_make_count_matrix.sh
rice_D_rep2_r1.pe.fastq.gz   rice_N_rep2_r1.unpe.fastq.gz
```

- FastQC\_after\_preprocess: 前処理後のリードのFastQCの結果
- FastQC\_before\_preprocess: 前処理前のリードのFastQCの結果
- Trimmomatic\_rice\_\*.log: Trimmomaticのログ (リードごとのトリミング情報)
- rice\_\*.pe.fastq.gz: 前処理後のリード配列ファイル (ペアを維持)
- rice\_\*.unpe.fastq.gz: 前処理後のリード配列ファイル (ペアの相方が捨てられたもの)

# 前処理結果の確認

Trimmomaticによる前処理結果の統計情報は標準エラーとして出力される。標準エラー出力をファイルに書き出していくれば、以下のように確認できる。

```
$ less step1_preprocessing.log
...
Input Read Pairs: 18407642 Both Surviving: 16992068 (92.31%) Forward Only
Surviving: 1173984 (6.38%) Reverse Only Surviving: 159615 (0.87%) Dropped:
81975 (0.45%)
...
```

- ・ 前処理の結果、16,992,068 ペア（92.31%）が残り、このあとの解析に利用される。
- ・ ペアの片方しか残っていない (\* Only Surviving) 、もしくはペアの両方が失われてしまった（Dropped）割合があまりに多い場合は、トリミングのパラメータを緩めるなど再検討する。あまりに酷い場合は再シーケンシングも検討。

\*この例は演習用データではなく、その元となった全ゲノムデータによる結果。  
演習用データは前処理後にゲノムにアラインメントできたリードのみ抽出しているため、Both Survivingが100%になっているはず。

## 1. 低クオリティ塩基が多い

シークエンシング時の問題であることが多いので読み直す。

## 2. アダプター配列の混入率が高い

RNAの濃度が薄い、解読リード長に対してインサート長が短いなど。ライブラリ調製からやり直す。

## 3. rRNAの混入率が高い

ポリA精製やrRNAの除去がうまくいっていない。ライブラリ調製からやり直す。

## 4. ゲノム、転写産物配列へのマップ率が低い

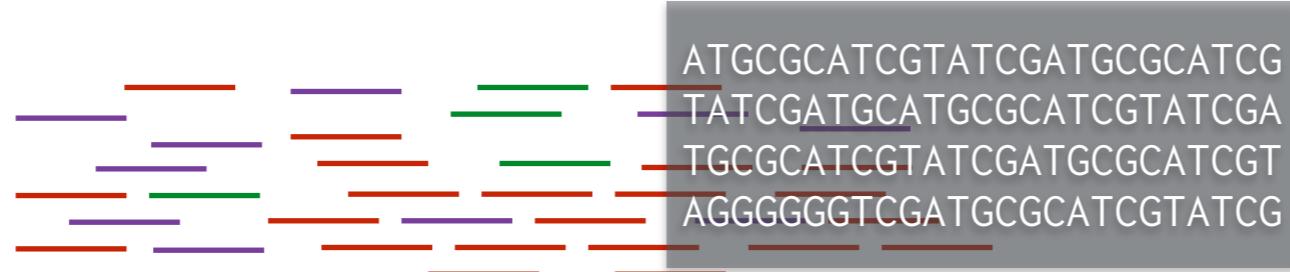
コンタミ等が疑われる。サンプリングからやり直した方が良い場合もある。

## 5. 有効なリード数が少ない

シークエンシングを追加する。生物種や対象組織、目的等によっても必要なリード数は異なるが、個人的にはイネの葉であれば1反復当たり1-2千万（10-20 million）ペアがあれば十分だと思う。

# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

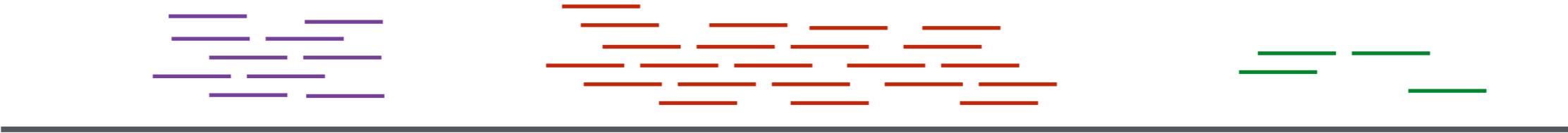
アラインメント

HISAT2

リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

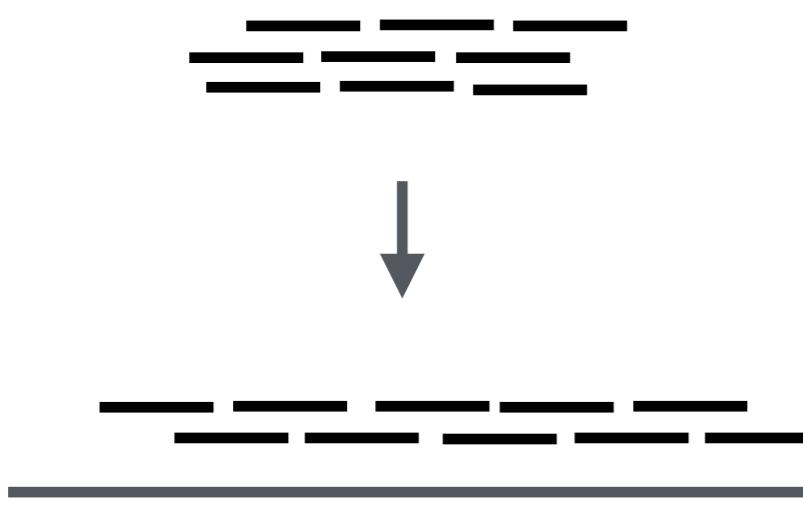
統計解析、可視化

IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

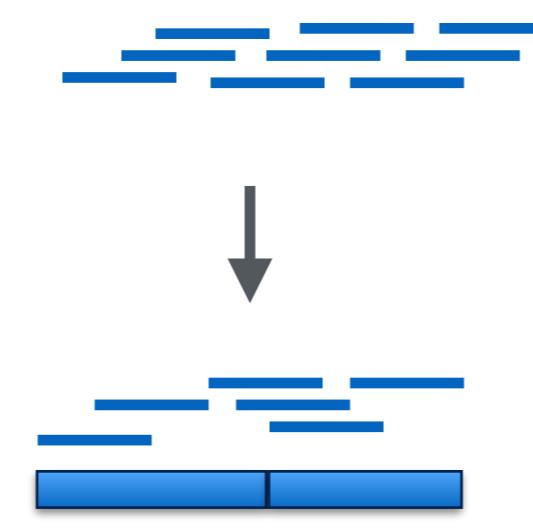
# RNA-Seqリードとゲノム配列のアラインメントの難しさ

ゲノムリシークエンスリード



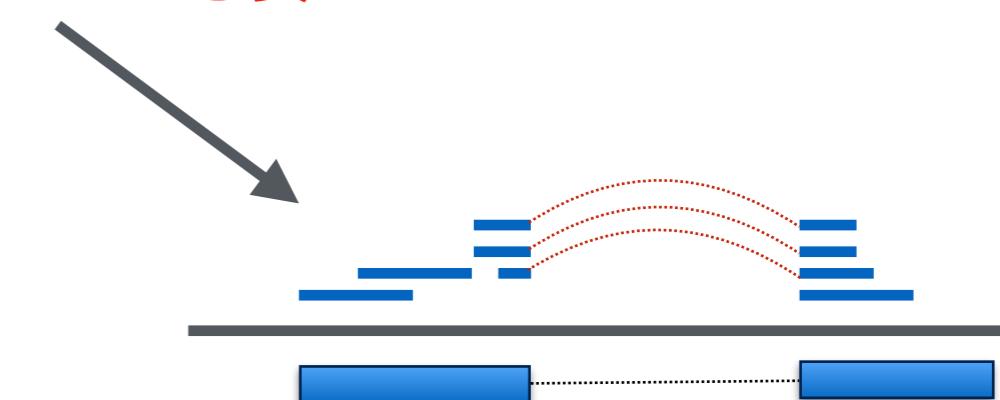
ゲノム配列

RNA-Seqリード



転写産物配列

長いギャップ（イントロン）を含むアラインメントを考慮する必要がある



ゲノム配列

- ・ ゲノムリシークエンス解析でよく使われるBWAやBowtie2などはRNA-Seqデータ解析には向かない。
- ・ RNA-Seqリードアラインメントに特化した様々なツール(HISAT2、STAR2、GSNAPなど)が開発されている。

# Step 2. HISAT2のためのインデックスの作成

HISAT2のためのインデックスを作成するシェルスクリプト

```
$ less ./step2_HISAT2-build.sh
```

- step2\_HISAT2-build.sh -

```
DataDir=$HOME/RNA-Seq/data
ToolDir=$HOME/RNA-Seq/tool
HISAT2_bin=$ToolDir/hisat2-2.2.0
Samtools_bin=$ToolDir/samtools-1.13

export PATH=$HISAT2_bin:$Samtools_bin:$PATH

### Step2. Build index of the reference genome sequence by hisat2-build and samtools
# make splice site and exon position info
python $HISAT2_bin/hisat2_extract_splice_sites.py $DataDir/annotation.gtf > ss.tab
python $HISAT2_bin/hisat2_extract_exons.py $DataDir/annotation.gtf > exon.tab
# build index for HISAT2
hisat2-build --ss ss.tab --exon exon.tab $DataDir/genome.fa genome ..... HISAT2のための
# build index for IGV etc. ..... インデックスの作成
samtools faidx $DataDir/genome.fa ..... IGVによる可視化時に必要なゲノム配列のインデックス作成
```

HISAT2用のhierarchical index作成時に、既知のスプライスサイト(ss.tab)やExon(exon.tab)の位置情報を指定している。これによりアラインメントの精度と効率が向上する。

# Step 3. HISAT2のためのインデックスの作成



シェルスクリプトの実行 (実行時間：約1分)

```
$ bash ./step2_HISAT2-build.sh 2>&1 | tee step2_HISAT2-build.log
```

作成されたインデックスファイルの確認

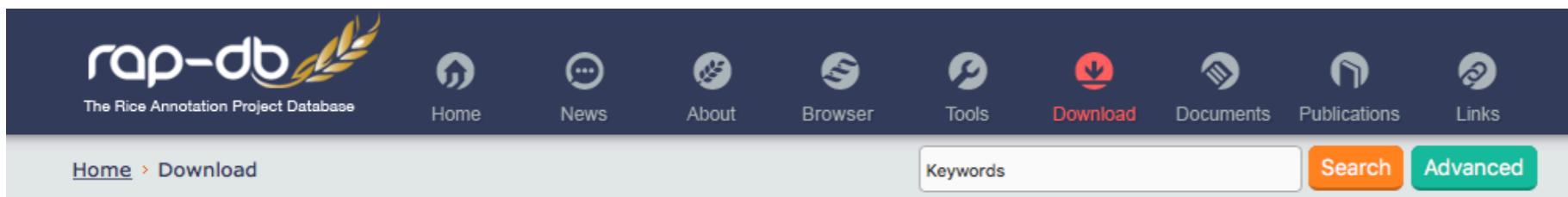
```
$ ls genome.*  
genome.1.ht2  genome.3.ht2  genome.5.ht2  genome.7.ht2  
genome.2.ht2  genome.4.ht2  genome.6.ht2  genome.8.ht2  
$ ls ../data/genome.fa*  
../data/genome.fa  ../data/genome.fa.fai
```

- genome.\*.ht2ファイルはHISAT2のためのインデックス
- genome.fa.faiは、IGVなどの可視化に必要なインデックス

# 遺伝子アノテーションファイル (GTF2/GFF3)

ゲノム上のどこにどんな構造の遺伝子が存在するのかを記載した情報。イネであればRAP-DBから代表転写産物のアノテーション情報 (GFF/GTF) がダウンロードできる。

<http://rapdb.dna.affrc.go.jp/download/irgsp1.html>



The screenshot shows the RAP-DB website interface. At the top, there is a navigation bar with icons for Home, News, About, Browser, Tools, Download (highlighted in red), Documents, Publications, and Links. Below the navigation bar, there is a breadcrumb trail "Home > Download" and a search bar with "Keywords" and "Search" buttons. The main content area is titled "Annotation data on Os-Nipponbare-Reference-IRGSP-1.0". Under the "Genome sequences" section, there are links for "Genome assemblies (12 chromosomes)\*" and "Unanchored sequences\*". A note says "(\*) Sequences are masked by Censor with MIPS and MSU repeat data. The masked regions are replaced by lowercase letters." Under the "Gene set (genes supported by FL-cDNAs, ESTs or proteins)" section, there are four items listed, with the first two highlighted by a red box.

## Annotation data on Os-Nipponbare-Reference-IRGSP-1.0

### Genome sequences

- Genome sequence (Os-Nipponbare-Reference-IRGSP-1.0)
  - Genome assemblies (12 chromosomes)\*** [\[DOWNLOAD\]](#) (gz file, 116MB, [MD5 checksum](#))
  - Unanchored sequences\*** [\[DOWNLOAD\]](#) (gz file, 356KB, [MD5 checksum](#))

(\*) Sequences are masked by [Censor](#) with [MIPS](#) and [MSU](#) repeat data. The masked regions are replaced by lowercase letters.
- Chromosome sequences of the aus rice cultivar 'Kasalath'.
  - [\[DOWNLOAD\]](#) (gz file, 199MB)

### Gene set (genes supported by FL-cDNAs, ESTs or proteins)

- Gene structure and function information in GFF format.
  - [\[DOWNLOAD\]](#) (gz file, 15MB)
- Gene structure (only exon) information in GTF format.
  - [\[DOWNLOAD\]](#) (gz file, 2.1MB)
- Gene annotation information in tab-delimited text format.
  - [\[DOWNLOAD\]](#) (gz file, 2.6MB)
- Gene sequences (CDS + UTRs + introns) in FASTA format.
  - [\[DOWNLOAD\]](#) (gz file, 39MB)

- 様々なデータベースから遺伝子の位置や機能情報が提供されているが、書式や記載方法は必ずしも統一されていない。
- HISAT2やStringTieで正しく扱えるような形式に整形しておくと解析に便利。RAP-DBのGTFファイルはHISAT2に対応。

### Illumina社 iGenomes

[https://support.illumina.com/sequencing/sequencing\\_software/igenome.html](https://support.illumina.com/sequencing/sequencing_software/igenome.html)

### EnsemblPlants

<https://plants.ensembl.org/index.html>

### Phytozome

<https://phytozome-next.jgi.doe.gov>

# HISAT2やStringTieで使用可能なGTFファイルの書式



遺伝子の位置やID、アノテーション情報などが記載されたタブ区切りのテキストファイル

```
$ less ../data/annotation.gtf
...
chr02    irgsp1_rep    transcript    30094300      30099072      .          +          .
gene_id "Os02g0724000"; transcript_id "Os02t0724000-01"; gene_name "DTH2"; note "CONSTANS-like
protein, Heading promotion under long-day condition";
chr02    irgsp1_rep    exon        30094300      30094498      .          +          .          gene_id
"Os02g0724000"; transcript_id "Os02t0724000-01";
chr02    irgsp1_rep    exon        30096198      30096893      .          +          .          gene_id
"Os02g0724000"; transcript_id "Os02t0724000-01";
chr02    irgsp1_rep    exon        30097390      30097590      .          +          .          gene_id
"Os02g0724000"; transcript_id "Os02t0724000-01";
chr02    irgsp1_rep    exon        30097861      30098165      .          +          .          gene_id
"Os02g0724000"; transcript_id "Os02t0724000-01";
chr02    irgsp1_rep    exon        30098451      30099072      .          +          .          gene_id
"Os02g0724000"; transcript_id "Os02t0724000-01";
...
...
```

- 上の例では6行で1つの転写産物（5 exons）の構造とアノテーション情報を示す。
- 「gene\_id」（遺伝子座ID）や「transcript\_id」（転写産物ID）は、遺伝子座や転写産物を対象にした発現解析を行うために必須。
- 「gene\_name」（遺伝子名やシンボル）を付けておくと、解析結果に含まれるので便利。

# Step 3. HISAT2によるRNA-Seqリードのアラインメント

6サンプル（2条件、3反復）を1つずつ順番にHISAT2でアラインメントするためのシェルスクリプト。

```
$ less step3_HISAT2.sh
```

- step3\_HISAT2.sh -

```
DataDir=$HOME/RNA-Seq/data
ToolDir=$HOME/RNA-Seq/tool
HISAT2_bin=$ToolDir/hisat2-2.2.0
Samtools_bin=$ToolDir/samtools-1.13
```

```
export PATH=$HISAT2_bin:$Samtools_bin:$PATH
```

共通パラメータ

```
### Step3. Align reads to the reference genome by HISAT2
```

```
HISAT2_COMMON_PARAM="--threads 1 --min-intronlen 20 --max-intronlen 10000 --dta --rna-strandness RF -x genome"
```

```
for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_N_rep1 rice_N_rep2 rice_N_rep3
do
```

```
    hisat2 $HISAT2_COMMON_PARAM -1 ${DATASET}_r1.pe.fastq.gz -2 ${DATASET}_r2.pe.fastq.gz
```

```
-S ${DATASET}.sam
```

```
    samtools sort -o ${DATASET}.bam ${DATASET}.sam
```

```
    samtools index ${DATASET}.bam
```

```
    rm ${DATASET}.sam
```

共通パラメータ、リードファイル、

出力ファイルを指定してHISAT2を実行

SAM形式で出力されたアラインメントをソート、BAM形式へ変換したのち、BAMインデックスを作成し、SAMを削除

```
done
```

# Step 3. HISAT2によるRNA-Seqリードのアラインメント



シェルスクリプトの実行  
(実行時間：約1分)

```
$ bash ./step3_HISAT2.sh 2>&1 | tee step3_HISAT2.log
```

作成されたアラインメントファイルとそのインデックスの確認

```
$ ls *.bam*
rice_D_rep1.bam      rice_D_rep3.bam      rice_N_rep2.bam
rice_D_rep1.bam.bai  rice_D_rep3.bam.bai  rice_N_rep2.bam.bai
rice_D_rep2.bam      rice_N_rep1.bam      rice_N_rep3.bam
rice_D_rep2.bam.bai  rice_N_rep1.bam.bai  rice_N_rep3.bam.bai
```

- \*.bamファイルはHISAT2によるアラインメントファイル。IGVなどで読み込み可能。
- \*.baiファイルはアラインメント情報のインデックス

# HISAT2によるRNA-Seqリードのアラインメント結果の確認



HISAT2によるアラインメントの統計情報は標準エラー出力に書き出される。

```
$ less step3_HISAT2.log
```

```
16992068 reads; of these:
```

```
 16992068 (100.00%) were paired; of these:
```

```
    491486 (2.89%) aligned concordantly 0 times
```

```
    16005810 (94.20%) aligned concordantly exactly 1 time
```

```
    494772 (2.91%) aligned concordantly >1 times
```

```
----
```

```
491486 pairs aligned concordantly 0 times; of these:
```

```
    240097 (48.85%) aligned discordantly 1 time
```

```
----
```

```
251389 pairs aligned 0 times concordantly or discordantly; of these:
```

```
    502778 mates make up the pairs; of these:
```

```
        344266 (68.47%) aligned 0 times
```

```
        140870 (28.02%) aligned exactly 1 time
```

```
        17642 (3.51%) aligned >1 times
```

```
98.99% overall alignment rate
```

\*この例は演習用データではなく、  
全ゲノムデータによる結果。

- ・ 全16,992,068リードのうち、98.99%がアラインメントされている。
- ・ 温帯ジャポニカのリードを日本晴リファレンスゲノムにアラインメントした場合、特に問題がなければ90%台後半のアラインメント率を示すのが一般的。

# HISAT2によるRNA-Seqリードのアラインメント結果の確認



samtoolsを使い、個々のリードのアラインメント情報を見ることができる。

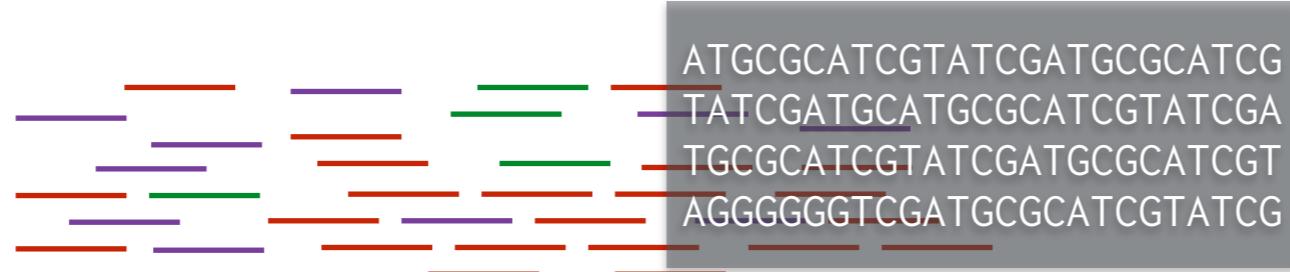
```
$ ./tool/samtools-1.13/bin/samtools view ./rice_D_rep1.bam | less
```

リードID	クエリ位置	ゲノム位置	スコア
MG00HS14:530:C6M7YACXX:8:2208:4438:63887	161	chr02 14245740	60
67M = 31527665 17282026			
TGAACGCTGGCGGCATGCTAACACATGCAAGTCGAACGGGAAGTGGTGTTCAGTGGCGAACGGG			
@@@A7DDDD811@FAFGFB9?BAD38B>FDFDFFFTEA@BFB>F==@@:?=>AADCABABBBB'05B			AS:i:0 XN:i:0
XM:i:0 XO:i:0		DP XS:A:+ NH:i:1	
MG00HS14:530:101M = CATCCCTGTCCCACACTGGCCGTA DCBB;@(;@CCC XG:i:0 NM:i:0 MD:Z:101	CIGAR 75M 112N 21M リード	chr02 23112840	60
ゲノム配列	----- ← 1S	ATGTGTTCTCCATCACCGGTCGTGGTACCGTTG	
		=CC>@7==@@E=;?=?	
		AS:i:0 XN:i:0 XM:i:0 XO:i:0	
MG00HS14:530:C6M7YACXX:8:1103:1603:42247	113	chr02 23845634	60
75M112N21M1S = 31055654 7210007			
GTCAGCTTCGCTGCCGCCGTGCTGGGGCTCCTCGCAGCCATCCTCGGGTTCGTCGCGGAAGGCGCCAAGTCCAATTGTTCGTGCG			
GTTCGACGGGG C@>CB>>BDB7DDDDBDDC?8(B?BA?B@>;DDDCC@A?<DDDDDDDB@B??			
<FFFHHGEJJIIIIJIIIEFJIHFIIIHGGJIGAFHHFFFFFCCC AS:i:-14 XN:i:0 XM:i:0			
XO:i:0 XG:i:0 NM:i:0 MD:Z:96 YS:i:-2 YT:Z:DP XS:A:+ NH:i:1			

- ・ SAMフォーマットについては<http://samtools.github.io/hts-specs/SAMv1.pdf>を参照。
- ・ イントロンをまたぐリードアラインメントは「75M112N21M1S」のように、イントロン部分が"N"で表されている。112bpのイントロンを挟むアラインメントであるという意味。

# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

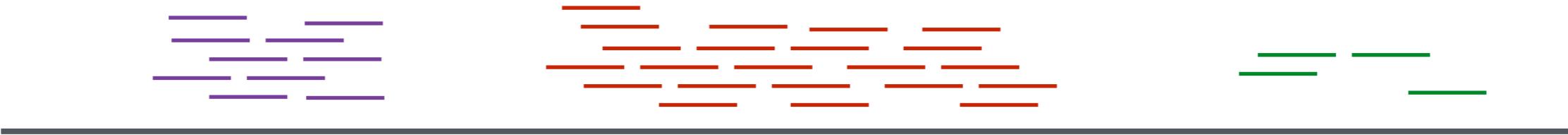
アラインメント

HISAT2

リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

統計解析、可視化

IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

# Step 4. StringTieによる遺伝子発現量の定量

StringTieによって、遺伝子アノテーション（GTF）と各サンプルのアラインメント情報（BAM）を元に、各遺伝子ごとにリードカウントや発現量を計算するシェルスクリプト。

```
$ less step4_StringTie-abundance_estimation.sh
```

- step4\_StringTie-abundance\_estimation.sh -

```
DataDir=$HOME/RNA-Seq/data
ToolDir=$HOME/RNA-Seq/tool
StringTie_bin=$ToolDir/stringtie-2.1.6.Linux_x86_64
```

```
export PATH=$StringTie_bin:$PATH
```

```
### Step4. Estimate transcript abundances
```

```
STRINGTIE_COMMON_PARAM="-e -B"
```

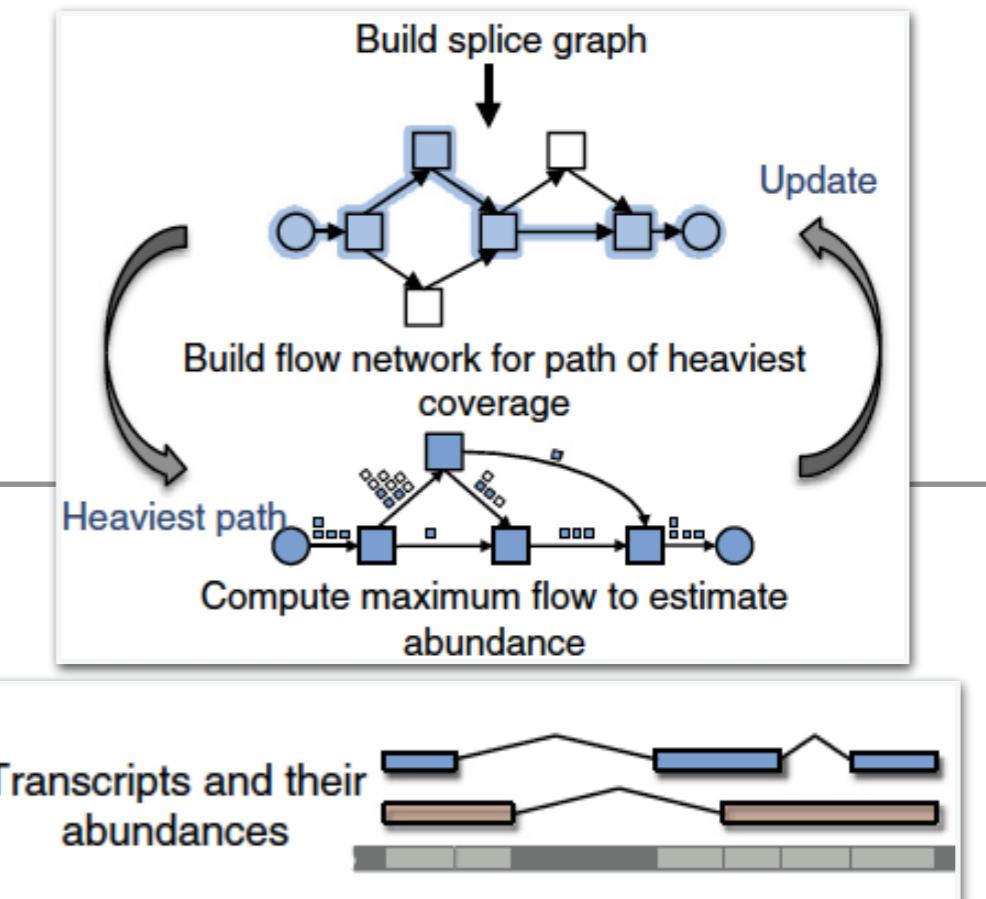
```
for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_N_rep1 rice_N_rep2 rice_N_rep3
```

```
do
```

```
    stringtie $STRINGTIE_COMMON_PARAM -G $DataDir/annotation.gtf -o ballgown/${DATASET}/${DATASET}.gtf ${DATASET}.bam
```

```
done
```

遺伝子アノテーションと出力ファイル、  
アラインメントデータを指定して、  
StringTieを実行し、発現量を定量。



modified Figure 1 from Pertea, M. et al. (2015)  
*Nature Biotechnology*, 33(3)

# Step 4. StringTieによる遺伝子発現量の定量

シェルスクリプトの実行 (実行時間：約10秒)

```
$ bash ./step4_StringTie-abundance_estimation.sh 2>&1 | tee \  
step4_StringTie-abundance_estimation.log
```

サンプルごとの発現量やCoverageデータの確認

```
$ ls ballgown/rice_D_rep1  
e2t.ctab    e_data.ctab   i2t.ctab   i_data.ctab   rice_D_rep1.gtf   t_data.ctab
```

各サンプルごとに指定したディレクトリに結果が出力されている。

- rice\_\*.gtf : 遺伝子構造と遺伝子発現量
- e2t.ctab : exon id と transcript id の対応表
- e\_data.ctab : exonごとの支持するリード数
- i2t.ctab : intron id と transcript id の対応表
- i\_data.ctab : intronごとの支持するリード数
- t\_data.ctab : transcriptごとの支持するリード数や発現量情報

# StringTieの出力結果の確認

transcriptごとの支持するリード数や発現量情報 (t\_data.ctab) の確認

```
$ less ballgown/rice_D_rep1/t_data.ctab
```

t_id	chr	strand	start	end	<b>t_name</b>	num_exons	length	gene_id
gene_name	cov				<b>FPKM</b>			
1	chr02	-	29998326		30002783			<b>Os02t0722500-01</b> 9 1436
Os02g0722500	-		5.784122		<b>228.790451</b>			
2	chr02	+	30004088		30004574			<b>Os02t0722600-01</b> 1 487
Os02g0722600	-		0.000000		<b>0.000000</b>			
3	chr02	+	30008280		30008810			<b>Os02t0722650-00</b> 1 531
Os02g0722650	-		2.084746		<b>82.461945</b>			
4	chr02	+	30011066		30015609			<b>Os02t0722700-01</b> 7 1436
Os02g0722700	-		121.268120		<b>4796.750000</b>			
5	chr02	-	30015998		30025935			<b>Os02t0722800-01</b> 13 4485
Os02g0722800	OsWD40-53		4.016716			158.880859		
6	chr02	-	30016149		30018289			<b>Os02t0722800-02</b> 6 1316
Os02g0722800	-		38.850327		<b>1536.721436</b>			
7	chr02	+	30051750		30053275			<b>Os02t0723200-01</b> 1 1526
Os02g0723200	OsGT7		0.394495		<b>15.604217</b>			

- 1転写産物／1行で位置や構造情報、transcript id やgene id、遺伝子名、支持するリード数、発現量 (FPKM) が記載されている。
- 点線  は遺伝子座の区切り。

# 遺伝子発現量の指標 (RPKM/FPKM/TPM)

シーケンス量や遺伝子の長さで正規化した発現量指標

RPKMやFPKM=Read (Fragment) 数／総リード数(M)／遺伝子の長さ(kb)

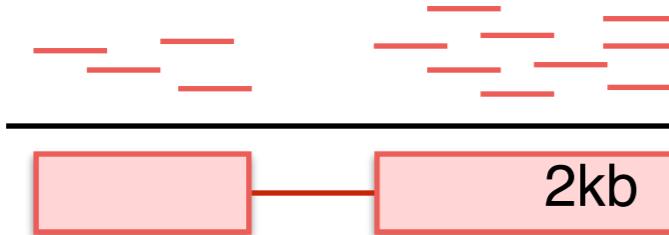
RPKM: reads per kilobase of exon model per million mapped reads

FPKM: fragments per kilobase of transcript per million fragments mapped

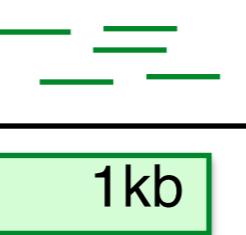
## 条件1

リードカウント (≠発現量)

13リード



6リード

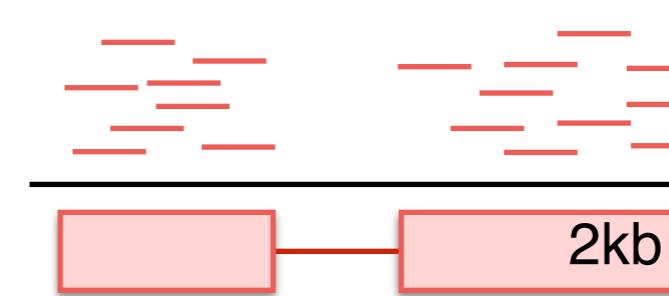


RPKM:  $13/2/19=0.34$

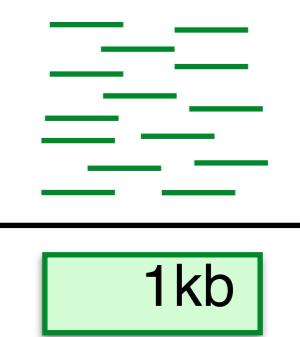
6/1/19=0.32

## 条件2

18リード



14リード



18/2/32=0.28

14/1/32=0.44

- 全リード中に含まれるある特定の転写産物由来のリードの割合。つまり、サンプル中の全RNAのうちのどれくらいがある特定の転写産物由来かを表す相対的な値。
- 発現する遺伝子の顔ぶれやたくさんの遺伝子の発現量が大きく異なるサンプル間の比較ではサンプル間で発現量を正規化する（発現量の分布を揃える）必要がある。

# 遺伝子発現量の指標 (RPKM/FPKM/TPM)



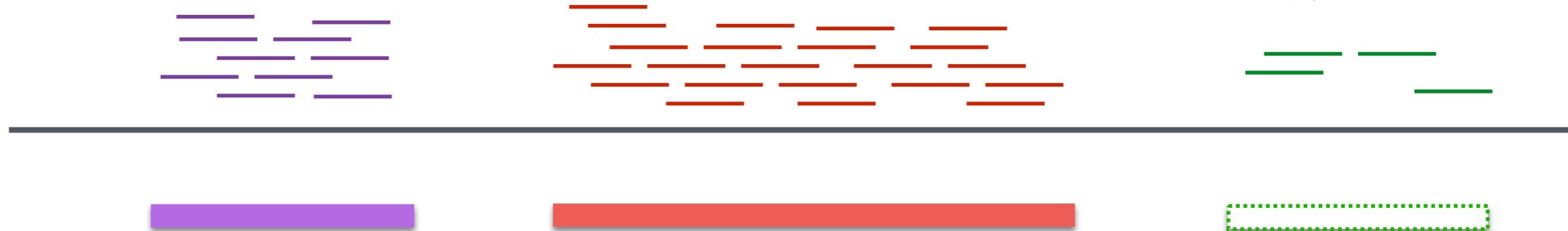
	RPKM/FPKM	TPM
Step.1	総リード数を補正 (100万リード換算)	遺伝子の長さを補正 (1kb換算)
Step.2	遺伝子の長さを補正 (1kb換算)	総リード数を補正 (100万リード換算)

- 全遺伝子のTPM(transcripts per million)の総和はサンプル間で等しいので、個々の遺伝子のTPM値はそのままサンプル間で比較が可能。
- 最近はTPMを使った比較が勧められている。

- Wagner et al. (2012) Theory in Biosciences
- <https://www.rna-seqblog.com/rpkf-fpkf-and-tpm-clearly-explained/>

# おまけ：新規転写産物構造の予測

RNA-Seq  
リード配列



RNA-Seqのアラインメント  
から転写領域の予測が可能

ゲノム配列

遺伝子

1. サンプルごとにStringtieを実行し、遺伝子構造を予測 (sample\*.gtfとして出力)

```
$ stringtie -o sample1.gtf -l sample1 sample1.bam
$ stringtie -o sample2.gtf -l sample2 sample2.bam
$ ...
```

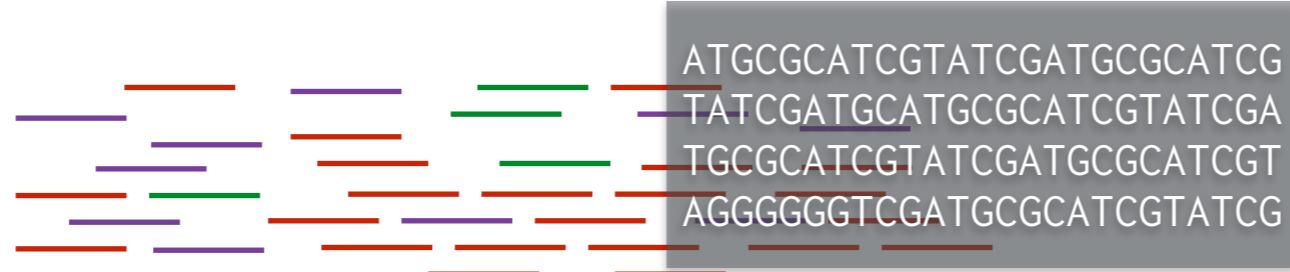
2. 得られたサンプルごとの遺伝子構造 (\*.gtf) をマージする。

```
$ stringtie --merge -G annotation.gtf -o stringtie_merged.gtf assemblies.txt
```

- assemblies.txtは、1.の出力のGTFファイル名が1行ごとに並んだテキストファイル
- -Gオプションで指定することで、既存の遺伝子構造 (annotation.gtf) も含めて、遺伝子構造を統合してくれる。
- step4のStringtie実行時に、この統合された遺伝子構造 (stringtie\_merged.gtf) を指定することで予測遺伝子も含めた発現量が得られる。

# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

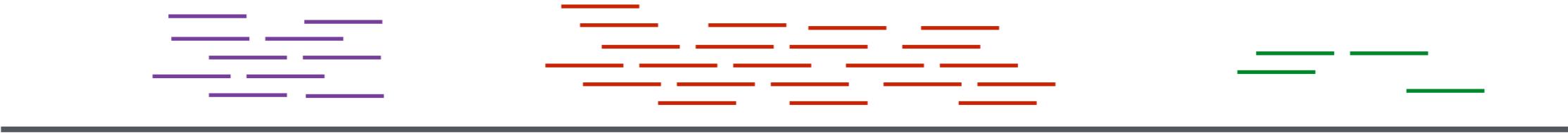
アラインメント

HISAT2

リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

統計解析、可視化

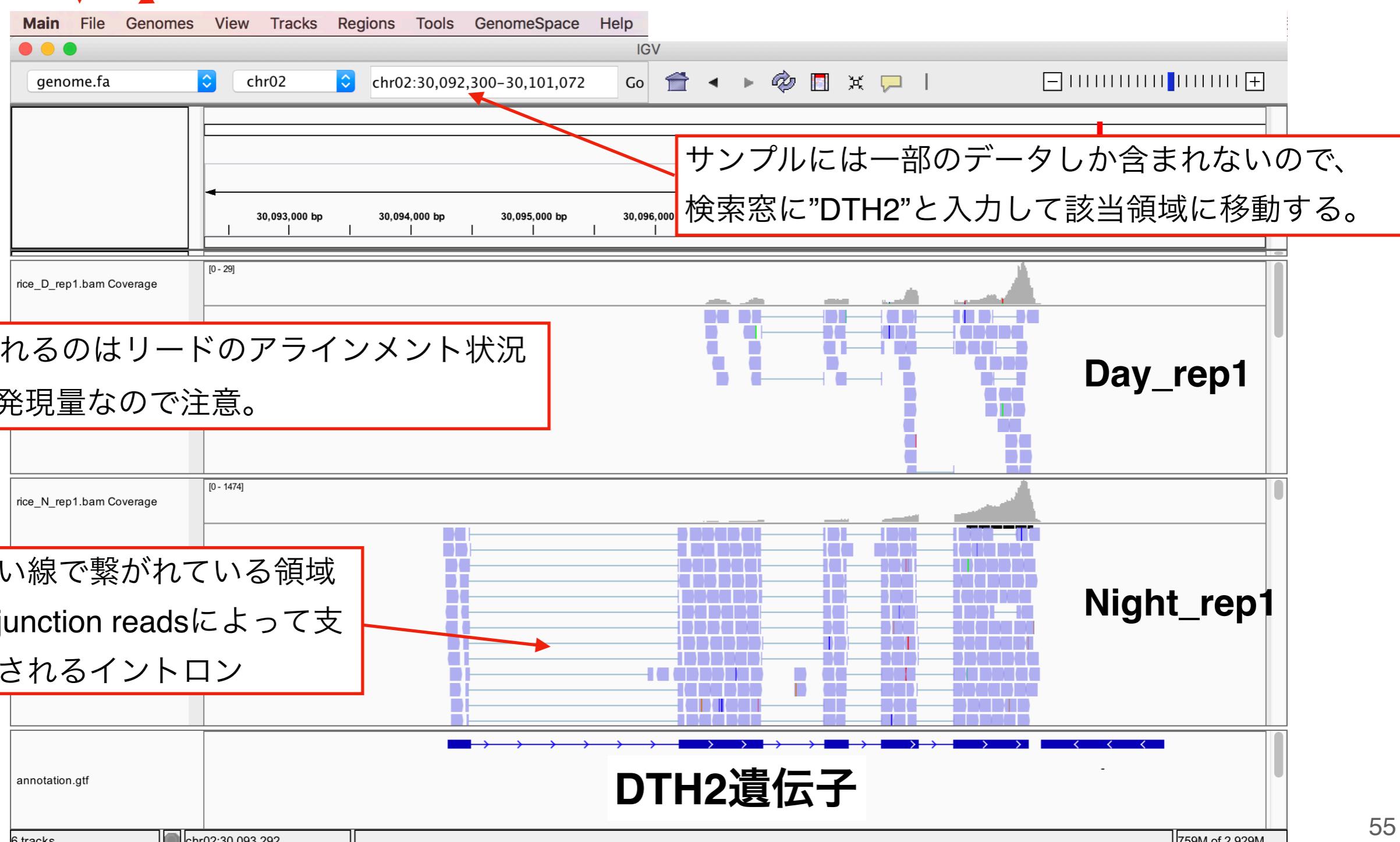
IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

# IGVを用いたRNA-Seqデータの可視化

デスクトップ上の「RNA-Seq/useref/alignment」中のリファレンスゲノムと遺伝子アノテーション、HISAT2が出力するBAMファイルを読み込む。

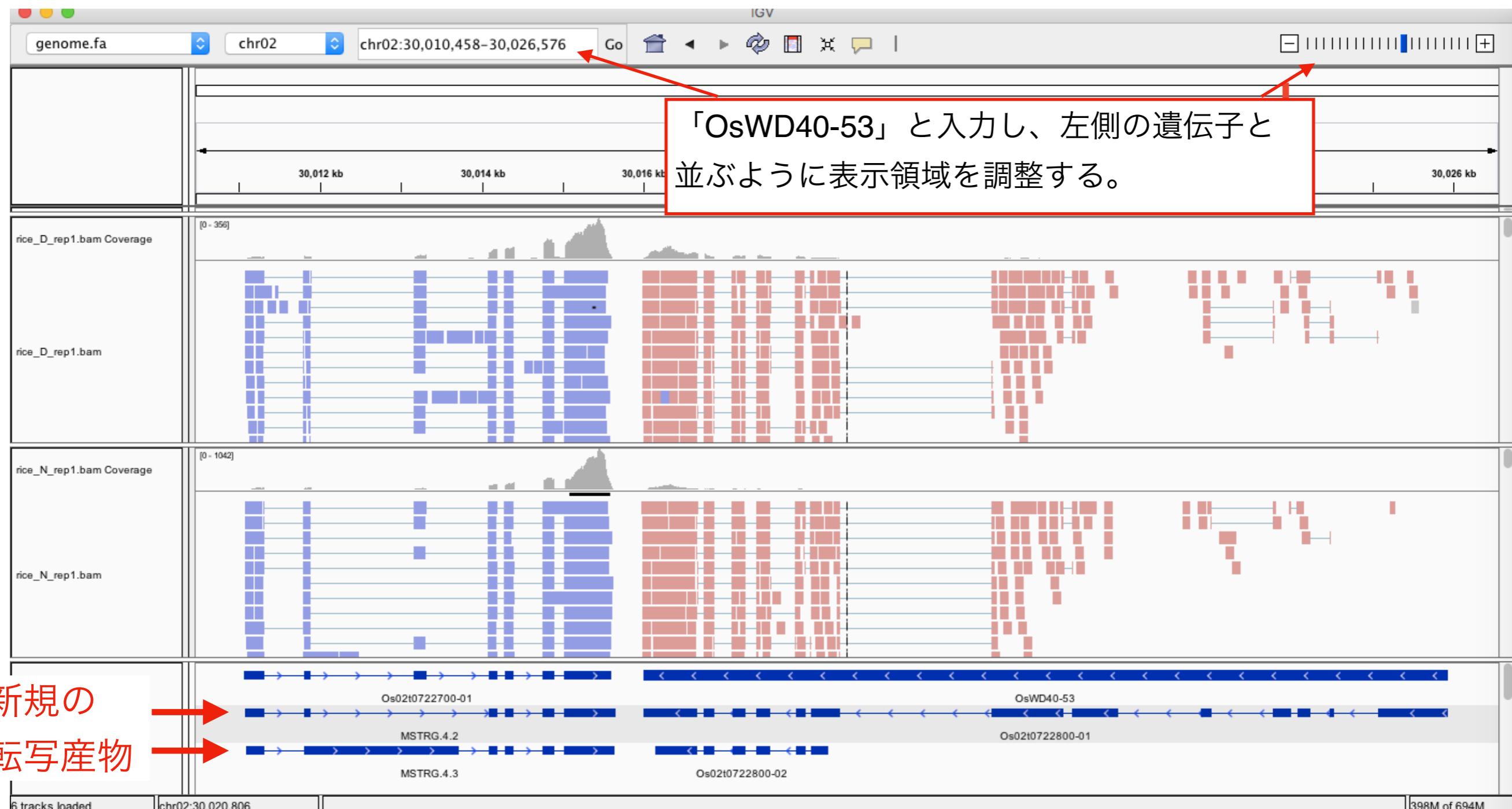
2. 「Load from File」からGTFファイルやBAMファイルを読み込む
1. 「Load Genome from File」からFASTAファイル（ゲノム配列）を読み込む



# IGVによるRNA-Seqデータのアライメント、遺伝子構造の可視化



- 本演習データはIllumina stranded mRNA-Seq法によるものなので、ペアのリード2の向きと転写方向が一致する（アライメントのトラック上で右クリックし、[Color alignments by] -> [first-of-pair strand] を選択すると色で区別できるようになる）。
- 新規の転写産物構造にはMSTRG.XX.XXといったIDが振られている。



# ロングリードによる転写産物配列の解読

転写産物配列の全長をロングリードで解読可能

アセンブルが不要なので精確な遺伝子構造が得られる

## Direct RNA / cDNA sequencing by ONT

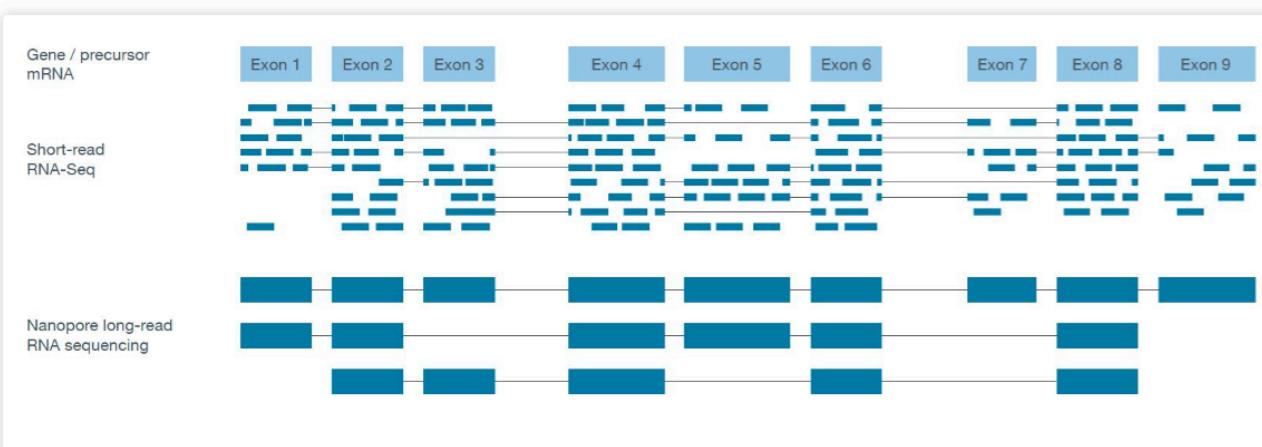
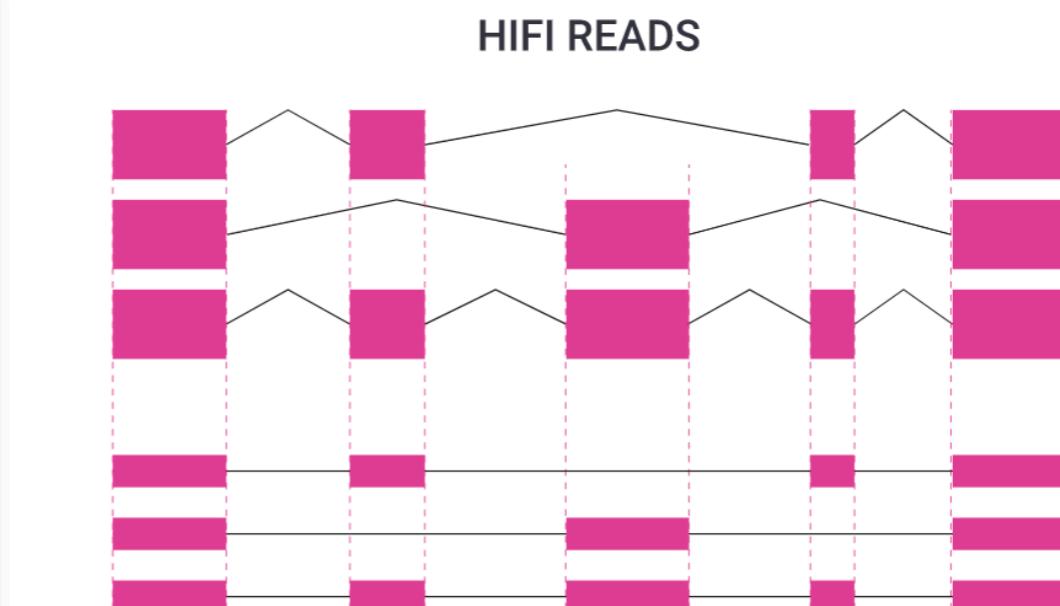


Figure 1: Alternative splicing can give rise to numerous mRNA isoforms per gene, which in turn can alter protein composition and function. The short reads generated by traditional RNA-Seq techniques lose positional information, making the correct assembly of alternative mRNA isoforms challenging. Long nanopore RNA sequencing reads can span full-length transcripts, simplifying their identification and quantification.

## Iso-Seq by PacBio HiFi



HiFi reads cover the full length of individual transcripts, providing reliable isoform information – no assembly required

[https://nanoporetech.com/  
applications/investigation/gene-expression](https://nanoporetech.com/applications/investigation/gene-expression)



[https://www.pacb.com/products-and-services/  
applications/rna-sequencing/](https://www.pacb.com/products-and-services/applications/rna-sequencing/)



# Desktop版のJBrowse 2も様々なデータの可視化に便利

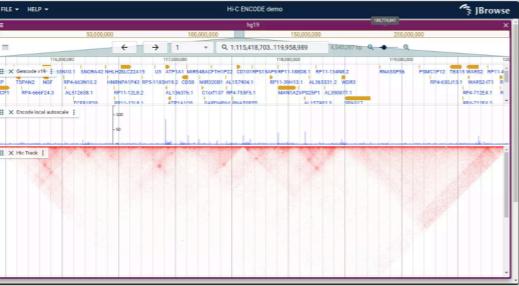
JBrowse

The next-generation genome browser

JBrowse is a new kind of genome browser that runs on the web, on your desktop, or embedded in your app.

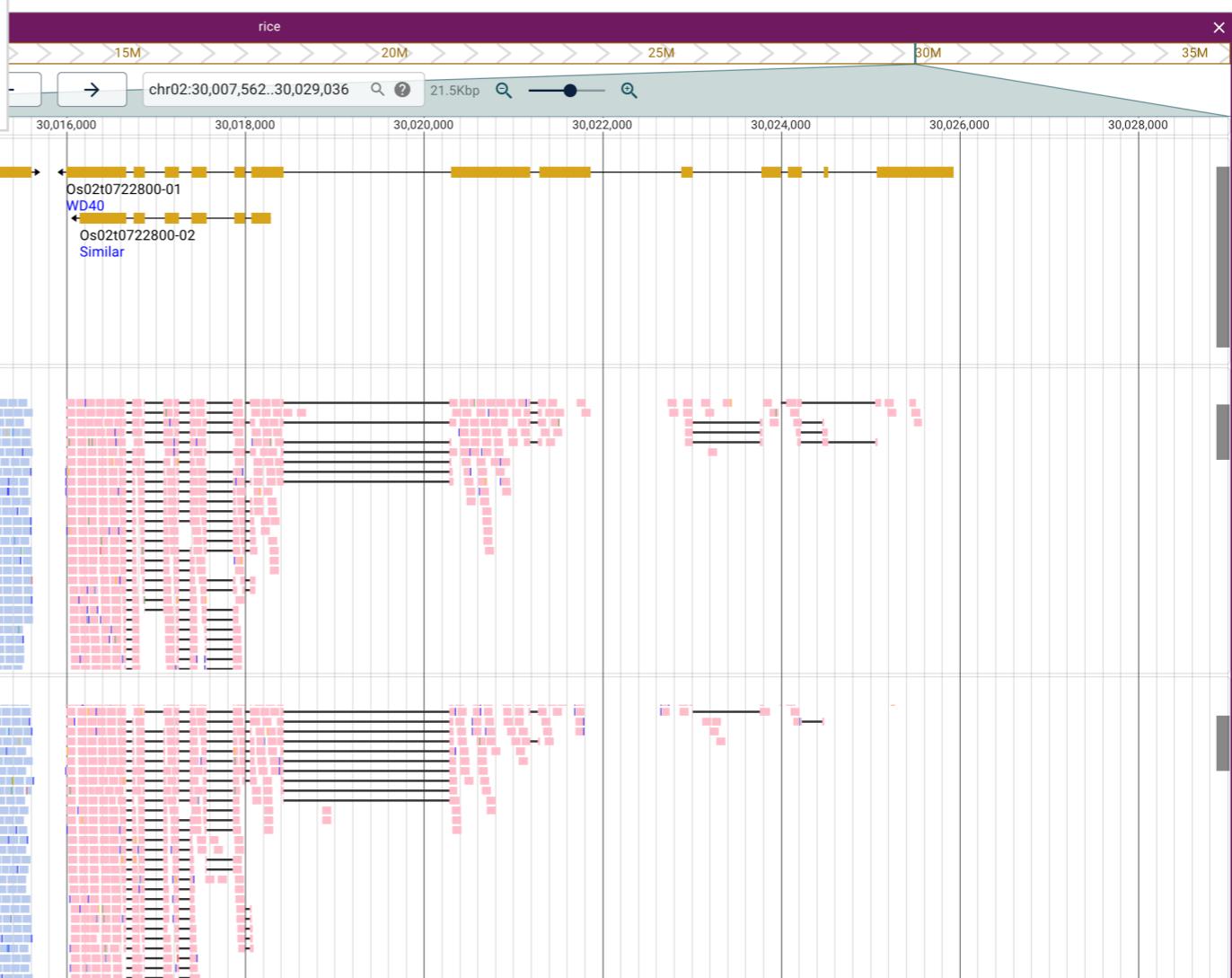
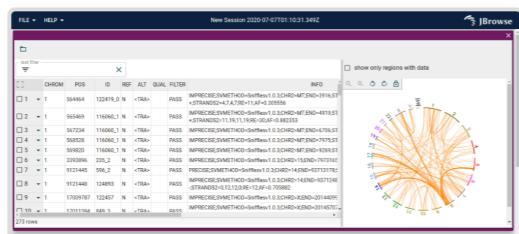
[DOWNLOAD](#) [BROWSE DEMO](#)

Also check out our [latest release blogpost](#), our [embedded components](#), and our [command line tools](#).



## Features

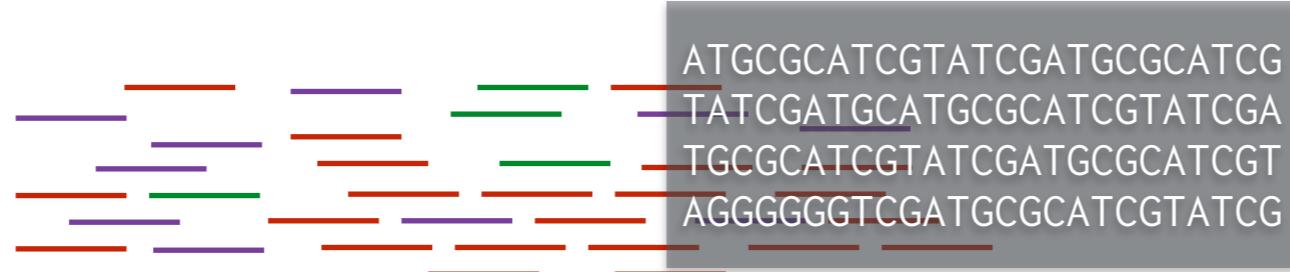
- Improved structural variant and comparative genomics visualization with linear, circular, dotplot, and synteny views
- Support for many common data types including BAM, CRAM, tabix indexed VCF, GFF, BED, BigBed, BigWig, and several specialized formats
- Endless extensibility with a plugin ecosystem which can add additional view types, track types, data adapters, and more!
- See a [summary of new features and a comparison to JBrowse 1](#)



OSごとにプログラムが用意されており、  
ダウンロードしてすぐに使える。  
<https://jbrowse.org/jb2/>

# リファレンス情報を用いたRNA-Seq解析の流れ

RNA-Seq  
リード配列



リードの前処理

FastQC  
Trimmomatic

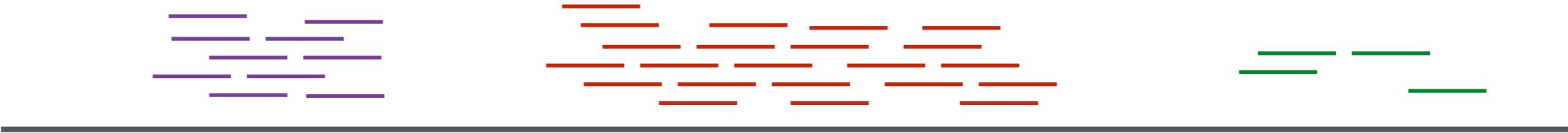
アラインメント

HISAT2

リードカウント、発現量の定量

StringTie

ゲノム配列



遺伝子



リードカウント

10

22

4

遺伝子発現量

10

11

4

統計解析、可視化

IGV  
Ballgown

2サンプル間比較によるDifferentially expressed gene (DEG)の検出、結果の可視化など

# R/Bioconductorによる統計解析と可視化

 Bioconductor  
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Home    Install    Help    Developers    About    Search:

Home » Bioconductor 3.5 » Software Packages » ballgown

## ballgown

platforms all    downloads top 5%    posts 10 / 0.9 / 0.7 / 1    in Bioc 3 years  
 build ok    commits 2.17    test coverage 56%

DOI: [10.18129/B9.bioc.ballgown](https://doi.org/10.18129/B9.bioc.ballgown) [f](#) [t](#)

Flexible, isoform-level differential expression analysis

Bioconductor version: Release (3.5)  
 Tools for statistical analysis of assembled transcriptomes, including flexible differential expression analysis, visualization of transcript structures, and matching of assembled transcripts to annotation.

Author: Jack Fu [aut], Alyssa C. Frazee [aut, cre], Leonardo Collado-Torres [aut], Andrew E. Jaffe [aut], Jeffrey T. Leek [aut, ths]  
 Maintainer: Jack Fu <jmfu at jhsph.edu>

**ballgown (Bioconductor)**  
<https://www.bioconductor.org/packages/release/bioc/html/ballgown.html>

**ballgown**は、R/Bioconductorのパッケージの一つであり、StringTieの結果を読み込んで、「発現比較解析（統計解析）」や「発現情報の可視化」の手法を提供している。

利用するためには統計計算とグラフィックスのための言語・環境である「R」をインストールする必要がある。また、RStudioはRの統合開発環境であり、使いやすいGUIを備えておりとても便利。

 The R Project for Statistical Computing

Getting Started  
 R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To [download R](#), please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

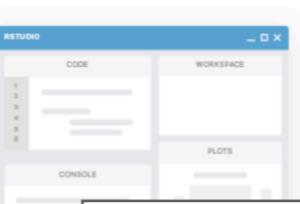
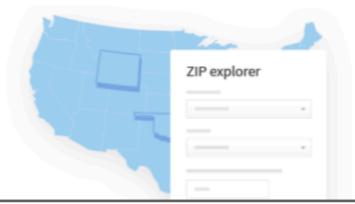
News  
 • R version 3.4.2 (Short Summer) has been released on Thursday 2017-09-28.

**R**  
<https://www.r-project.org/>

 Studio    rstudio::conf    Products    Resources    Pricing    About Us    Blogs    

**RStudio**  
 Open source and enterprise-ready professional software for R

Download RStudio    Discover Shiny    shinyapps.io Login    Discover RStudio Connect

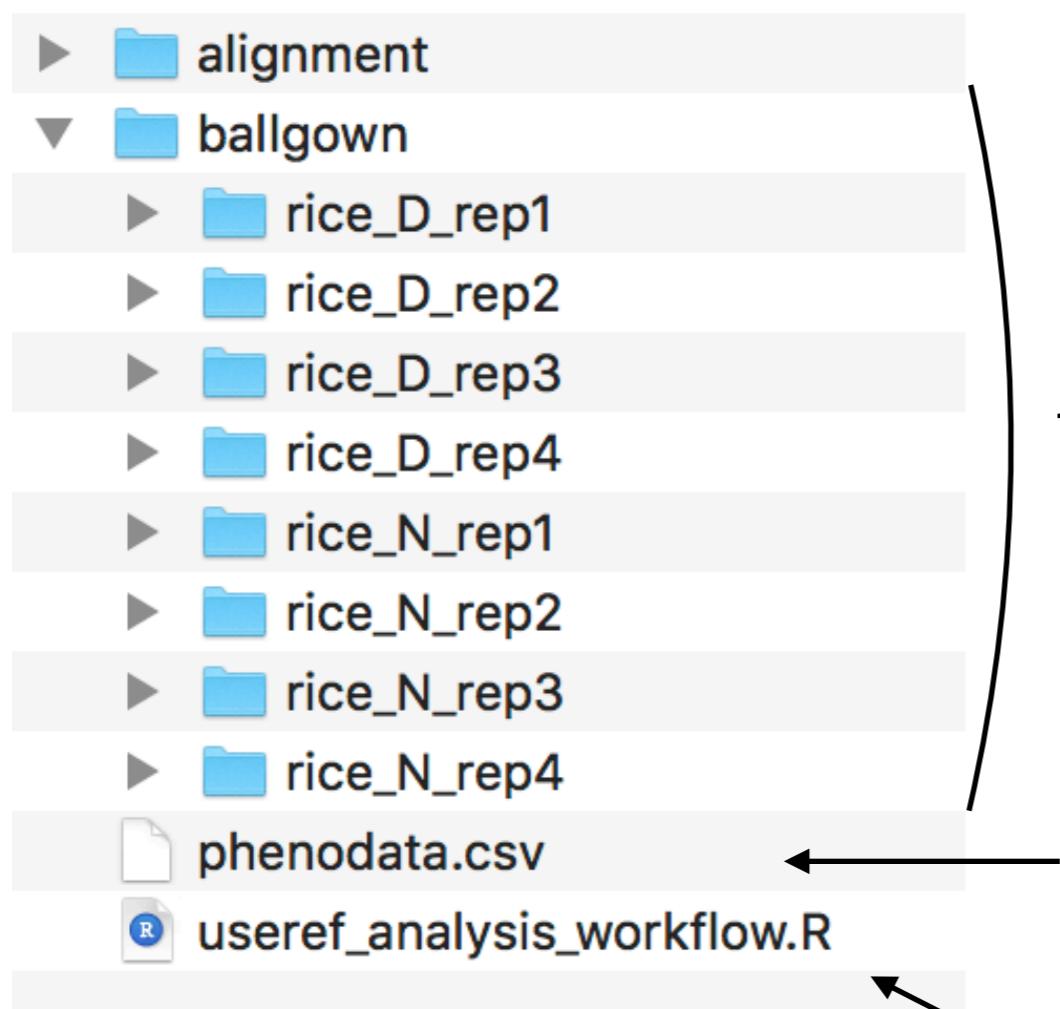


**RStudio**  
<https://www.rstudio.com/>

# データやスクリプトの置かれているディレクトリの確認



デスクトップ上の「RNA-Seq/useref」ディレクトリ中にある、以下の3つのデータを用いて解析を行なう。



1. 「ballgown」ディレクトリ中にある8サンプル分（2条件、4反復）のStringTieの結果
2. サンプル名や条件を記載した、カンマ区切りのテキストファイル（phenodata.csv）
3. 実行するコマンドを記載したRのスクリプト（useref\_analysis\_workflow.R）

演習で用いたデータよりも1反復多く、  
全ranscriptomeデータを使った解析結果になります

# Rstudioの起動



デフォルトでは4つの区画（ペイン）に分かれている

The screenshot shows the RStudio interface with four main panes:

- Script Editor (Left):** Displays an R script named "useref\_analysis\_workflow.R". A box highlights the title "Rスクリプトの編集".
- Environment (Top Right):** Shows the Global Environment with objects like "bg", "bg\_filtered", and "pheno\_data". A box highlights "変数一覧" (Variable List) and "作業履歴" (History).
- Files (Bottom Right):** Displays a file tree and a list of files in the current directory. A box highlights "ディレクトリ構造" (Directory Structure), "プロット" (Plots), and "ヘルプ" (Help).
- Console (Bottom Left):** Shows R command-line output. A box highlights "Rコンソール" (R Console).

Script Editor content (useref\_analysis\_workflow.R):

```
16 pheno_data <- read.csv("phenodata.csv")
17 # サンプル情報を表示
18 pheno_data
19
20 ### ballgownの入力データの読み込み (StringTieの結果ディレクトリを指定)
21 bg <- ballgown(dataDir="ballgown", samplePattern="rice", pData=pheno_data)
22 bg
23
24 ### サンプル間の分散が1以上の中のものを絞り込む、つまり発現変動が小さい遺伝子を除く
25 bg_filtered <- subset(bg, "rowVars(expr(bg))>=1")
26 bg_filtered
27
28 ### 転写産物レベルでサンプル (Day/Night) 間での遺伝子発現変動を検定し、p-valueやq-valueを計算する
29 results_transcripts <- stattest(bg_filtered, feature="transcript",
30 covariate="cond", getFC=TRUE, meas="FPKM")
```

Console output:

```
8 rice_N_rep4 Night 4
> ### ballgownの入力データの読み込み (StringTieの結果ディレクトリを指定)
> bg <- ballgown(dataDir="ballgown", samplePattern="rice", pData=pheno_data)
Sun Sep 16 21:53:29 2018
Sun Sep 16 21:53:29 2018: Reading linking tables
Sun Sep 16 21:53:29 2018: Reading intron data files
Sun Sep 16 21:53:31 2018: Merging intron data
Sun Sep 16 21:53:32 2018: Merging transcript files
Sun Sep 16 21:53:37 2018
Sun Sep 16 21:53:38 2018
Sun Sep 16 21:53:39 2018
Wrapping up the results
Sun Sep 16 21:53:39 2018
> bg
ballgown instance with 44586 transcripts and 8 samples
> ### サンプル間の分散が1以上のものに絞り込む、つまり発現変動が小さい遺伝子を除く
> bg_filtered <- subset(bg, "rowVars(expr(bg))>=1")
> bg_filtered
ballgown instance with 19991 transcripts and 8 samples
>
```

# 作業ディレクトリの指定

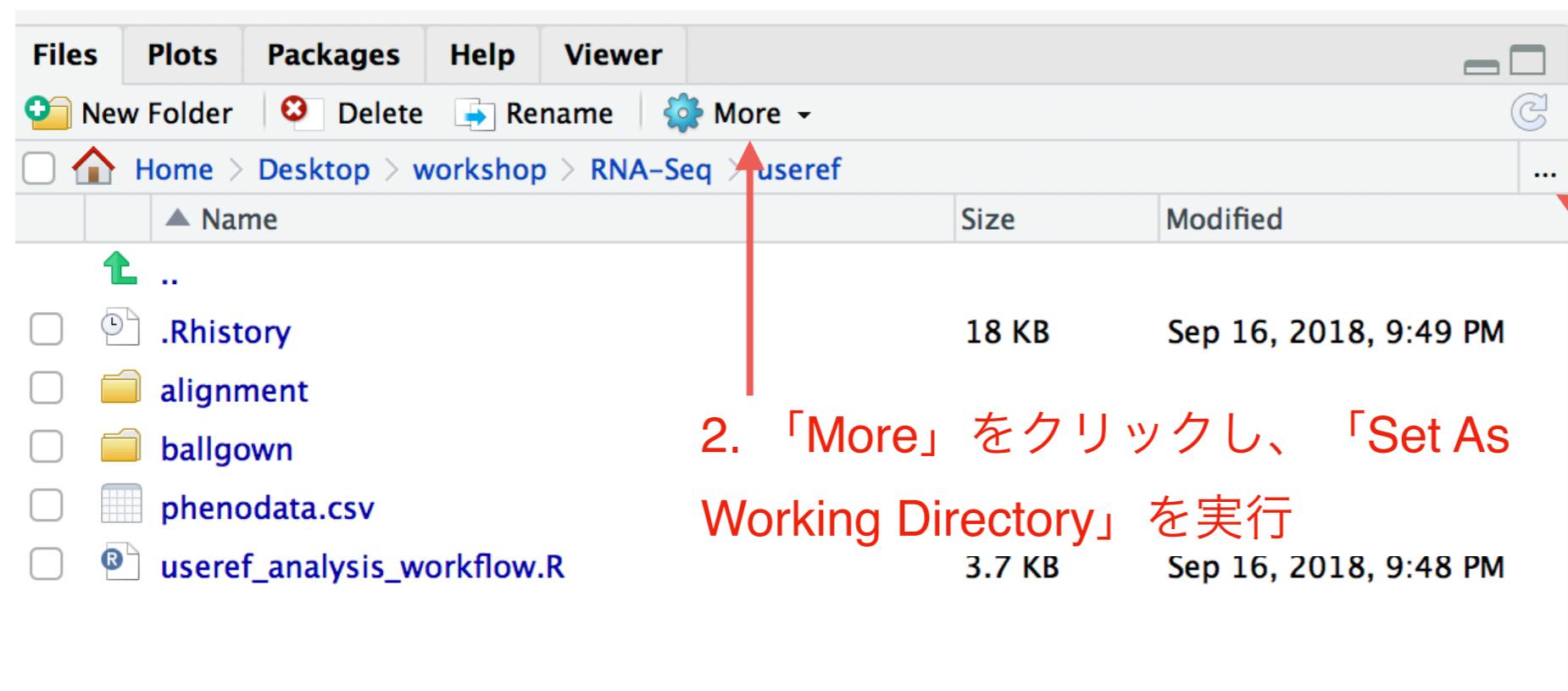
1. Rコンソール（左下のペイン）でコマンドを実行

```
> setwd("c:/Users/user/Desktop/NGSworkshop/RNA-Seq/")
```

\*各自の環境にあったパスに置き換えて下さい

もしくは、

2. ディレクトリ構造が表示されている右下のペインを操作



Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > Desktop > workshop > RNA-Seq > userref

	Name	Size	Modified
	..		
	.Rhistory	18 KB	Sep 16, 2018, 9:49 PM
	alignment		
	ballgown		
	phenodata.csv		
	userref_analysis_workflow.R	3.7 KB	Sep 16, 2018, 9:48 PM

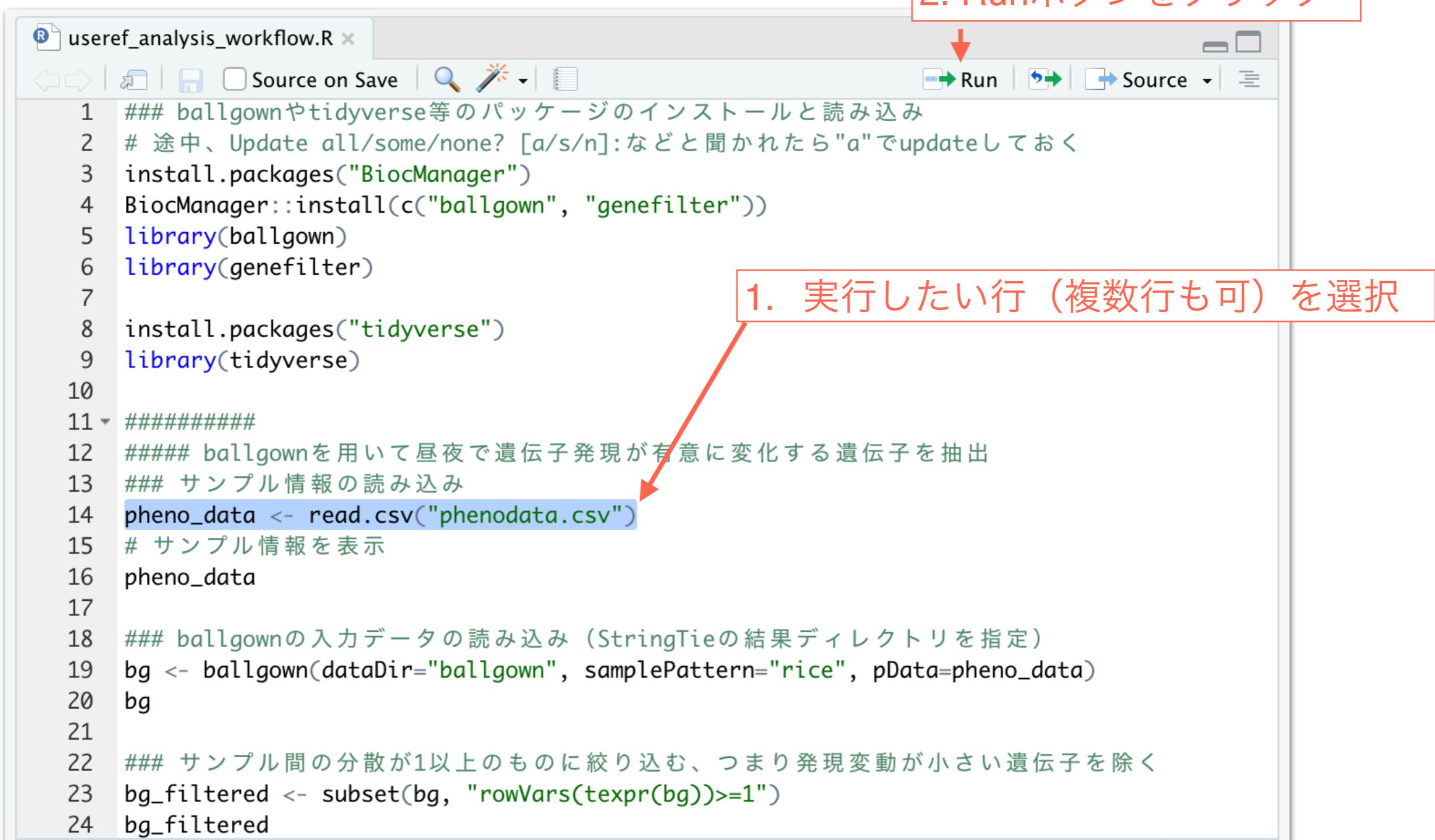
2. 「More」をクリックし、「Set As Working Directory」を実行

ballgownディレクトリがある場所を指定する。

1. 「...」をクリックし、「RNA-Seq」ディレクトリを選択

# Rスクリプト中のコマンドを順次実行していく

2. Runボタンをクリック



```

R useref_analysis_workflow.R
Source on Save | Run | Source
1 ### ballgownやtidyverse等のパッケージのインストールと読み込み
2 # 途中、Update all/some/none? [a/s/n]:などと聞かれたら "a"でupdateしておく
3 install.packages("BiocManager")
4 BiocManager::install(c("ballgown", "genefilter"))
5 library(ballgown)
6 library(genefilter)
7
8 install.packages("tidyverse")
9 library(tidyverse)
10
11 #####
12 ##### ballgownを用いて昼夜で遺伝子発現が有意に変化する遺伝子を抽出
13 ### サンプル情報の読み込み
14 pheno_data <- read.csv("phenodata.csv") ← 1. 実行したい行 (複数行も可) を選択
15 # サンプル情報を表示
16 pheno_data
17
18 ### ballgownの入力データの読み込み (StringTieの結果ディレクトリを指定)
19 bg <- ballgown(dataDir="ballgown", samplePattern="rice", pData=pheno_data)
20 bg
21
22 ### サンプル間の分散が1以上のものに絞り込む、つまり発現変動が小さい遺伝子を除く
23 bg_filtered <- subset(bg, "rowVars(expr(bg))>=1")
24 bg_filtered

```

シェルスクリプトと同様、どのような解析をおこなったかをあとで見直す際や、データやパラメータを変えて同じ解析をする際などにRスクリプトとして保存しておくと便利。

# パッケージのインストールと読み込み

ballgownを用いた統計解析に必要なパッケージのインストールと読み込み

```
> install.packages("BiocManager")
> BiocManager::install(c("ballgown", "genefilter", "DESeq2"))
> library(ballgown) ★
> library(genefilter) ★
> library(DESeq2) ★
```

- ・ DESeq2についても後ほど触れるので合わせてインストール、読み込みをしておく。
- ・ ballgownとgenefilter, DESeq2はR/Bioconductorのパッケージとして提供されており、  
BiocManagerのinstall関数でパッケージのインストールする。
- ・ 途中、「Update all/some/none? [a/s/n]:」と聞かれたら、「a」と入力しリターンを押し、全て  
アップデートする。多少エラーが出ても問題ないこともある。

```
> install.packages("tidyverse")
> install.packages("pheatmap")
> library(tidyverse) ★
> library(pheatmap) ★
```

Rのパッケージであるtidyverseとpheatmapは、  
install.packages関数を用いてインストールする。

\*すでにパッケージのインストールが完了していれば、

★印のついたlibrary関数によるパッケージの読み込みだけを行えばOKです。

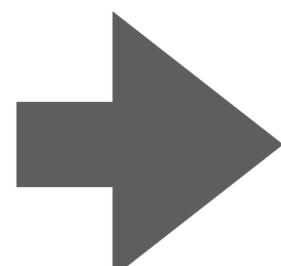
# サンプル情報 (phenodata.csv)

サンプル情報の読み込み

```
> pheno_data <- read.csv("phenodata.csv")
```

カンマ区切りの書式で、サンプル名、  
実験条件などを記載したファイル  
(phenodata.csv) を準備する。

ids,cond,rep	1行目は項目名
rice_D_rep1,Day,1	
rice_D_rep2,Day,2	
rice_D_rep3,Day,3	
rice_D_rep4,Day,4	
rice_N_rep1,Night,1	
rice_N_rep2,Night,2	
rice_N_rep3,Night,3	
rice_N_rep4,Night,4	



上のコマンドを実行すると  
データフレームとして読み込まれる。

> pheno_data	ids	cond	rep
1 rice_D_rep1	Day	1	
2 rice_D_rep2	Day	2	
3 rice_D_rep3	Day	3	
4 rice_D_rep4	Day	4	
5 rice_N_rep1	Night	1	
6 rice_N_rep2	Night	2	
7 rice_N_rep3	Night	3	
8 rice_N_rep4	Night	4	

# StringTieデータをballgownオブジェクトに格納する



StringTieデータを読み込み、ballgownオブジェクトに格納することで、様々な解析が可能になる。

```
> bg <- ballgown(dataDir="ballgown", samplePattern="rice", pData=pheno_data)
Mon Sep 10 08:00:18 2018
Mon Sep 10 08:00:18 2018: Reading linking tables
Mon Sep 10 08:00:18 2018: Reading intron data files
Mon Sep 10 08:00:20 2018: Merging intron data
Mon Sep 10 08:00:20 2018: Reading exon data files
Mon Sep 10 08:00:24 2018: Merging exon data
Mon Sep 10 08:00:25 2018: Reading transcript data files
Mon Sep 10 08:00:26 2018: Merging transcript data
Wrapping up the results
Mon Sep 10 08:00:26 2018
```

ballgownオブジェクトを指定してみると

```
> bg
ballgown instance with 44,586 transcripts and 8 samples
```

44,586 transcriptsについて、8サンプル分の遺伝子発現情報が読み込まれていることが分かる。

# 昼夜で遺伝子発現量が変化した遺伝子（DEG）の検出



ballgownのsubset関数を使い、サンプル間のFPKMの分散が1以上の転写産物に絞り込む、つまり発現変動が小さいものを除く

```
> bg_filtered <- subset(bg, "rowVars(expr(bg))>=1")
> bg_filtered
ballgown instance with 19991 transcripts and 8 samples
```

転写産物数が19,991 transcriptsとなり、絞り込まれている。

転写産物レベルでサンプル（Day/Night）間での遺伝子発現変動を検定し、p-valueやq-valueを計算する

```
> results_transcripts <- stattest(bg_filtered, feature="transcript",
covariate="cond", getFC=TRUE, meas="FPKM")
```

- covariateで比較対象の列名「cond」、measで利用する発現量の指標「FPKM」、getFCで結果にFold-changeを出力するよう指定している。
- 他にもタイムコースサンプルの場合に指定するものなど複数のオプションがある。

# 昼夜で遺伝子発現量が変化した遺伝子（DEG）の検出



検定の結果の確認。各転写産物ごとにDay/Night間のFold-changeやp-value、q-valueが計算されている。

```
> head(results_transcripts)
  feature id      fc      pval      qval
1 transcript 1 0.9180662 0.5169128022 0.68235630
2 transcript 6 0.8906488 0.2519004300 0.44433240
3 transcript 7 1.0106792 0.9216822885 0.95930393
4 transcript 9 0.4228804 0.1215415562 0.29074712
5 transcript 10 0.9335729 0.4569417800 0.63598991
6 transcript 11 1.4025491 0.0003284612 0.02168817
```

- ・このままでは遺伝子IDや遺伝子名などもなく分かりにくい。
- ・IDや遺伝子名の追加、q-valueによるソートなどを行うとよい。

# 昼夜で遺伝子発現量が変化した遺伝子 (DEG) の検出



遺伝子名や遺伝子ID、転写産物IDの列を追加する。

```
> results_transcripts = data.frame(geneNames=ballgown::geneNames(bg_filtered),  
geneIDs=ballgown::geneIDs(bg_filtered),  
transcriptIDs=ballgown::transcriptNames(bg_filtered), results_transcripts)
```

検定結果をp-valueの小さい順にソートする

```
> results_transcripts = arrange(results_transcripts,pval)
```

適当なq-valueの閾値(qval < 0.01)で切った遺伝子を抽出する（75転写産物がヒット）。

```
> subset(results_transcripts, results_transcripts$qval<0.01)
```

	geneNames	geneIDs	transcriptIDs	feature	id	fc	pval	qval
1	-	Os02g0623932	Os02t0623932-00	transcript	9347	1.334693e-01	3.016259e-07	0.003330334
2	LHY	Os04g0583900	Os04t0583900-01	transcript	19641	2.584726e+02	3.331834e-07	0.003330334
3	-	Os06g0660800	Os06t0660800-01	transcript	27355	6.321374e+00	6.792785e-07	0.003964490
4	OsBBX19, OsM, DTH2, qDTH-2	Os06g0298200	Os06t0298200-01	transcript	25939	6.919100e+01	9.295324e-07	0.003964490
5	PsbP	Os03g0279950	Os03t0279950-01	transcript	13109	2.380650e+01	9.915688e-07	0.003964490
6	-	Os08g0360100	Os08t0360100-01	transcript	32883	1.538616e-01	1.254840e-06	0.004073630
7	-	Os06g0281400	Os06t0281400-01	transcript	25839	3.943439e-01	1.426413e-06	0.004073630
...								
38	SIGA	Os08g0163400	Os08t0163400-01	transcript	31942	1.358213e-01	1.574039e-05	0.008169480
...								

# 遺伝子発現量の分布の可視化



全サンプルの平均発現量が1以上の転写産物を選び、遺伝子発現量（FPKM値に0.01を足してlog2をとったもの）を取得する（22,292個の転写産物）。

```
> bg.expressed <- subset(bg, "rowMeans(expr(bg))>=1")
> bg.expressed
ballgown instance with 22292 transcripts and 8 samples
> log2fpkm <- log2(expr(bg.expressed, meas="FPKM") + 0.01)
```

変数log2fpkmには、転写産物・サンプルごとに全FPKM値が格納されている。

```
> head(log2fpkm)
      FPKM.rice_D_rep1 FPKM.rice_D_rep2 FPKM.rice_D_rep3 FPKM.rice_D_rep4 FPKM.rice_N_rep1
1       3.539679        3.831836        4.0835609       4.1796822       3.897224
4       1.147461        1.055207        0.5679202      -0.4774179       1.388669
6       3.677559        3.757414        4.0814173       3.9224231       3.593344
7       3.333910        4.173667        4.0908662       4.1635102       3.890806
9       5.925956        6.953405        6.8972279       4.9053619       5.080402
10      3.458069        4.015651        4.1832976      4.0796425       3.893308
      FPKM.rice_N_rep2 FPKM.rice_N_rep3 FPKM.rice_N_rep4
1       4.2026522       4.325626       3.7204283
4       0.1995184       1.501587       0.6323222
6       3.8188390       4.363668       4.1252938
7       4.4518494       4.566686       4.6076217
9       5.5129445       5.091533       4.7157670
10      4.1539759       4.123816       4.2123396
```

# 遺伝子発現量の分布の可視化

データを整形する。

```
> log2fpkm.long <- as.data.frame(log2fpkm) %>%  
  tidyr::gather(Dataset, FPKM)
```

ワイドからロングフォーマットのデータフレームに変換されている。

```
> head(log2fpkm.long)  
  Dataset      FPKM  
1 FPKM.rice_D_rep1 3.539679  
2 FPKM.rice_D_rep1 1.147461  
3 FPKM.rice_D_rep1 3.677559  
4 FPKM.rice_D_rep1 3.333910  
5 FPKM.rice_D_rep1 5.925956  
6 FPKM.rice_D_rep1 3.458069
```

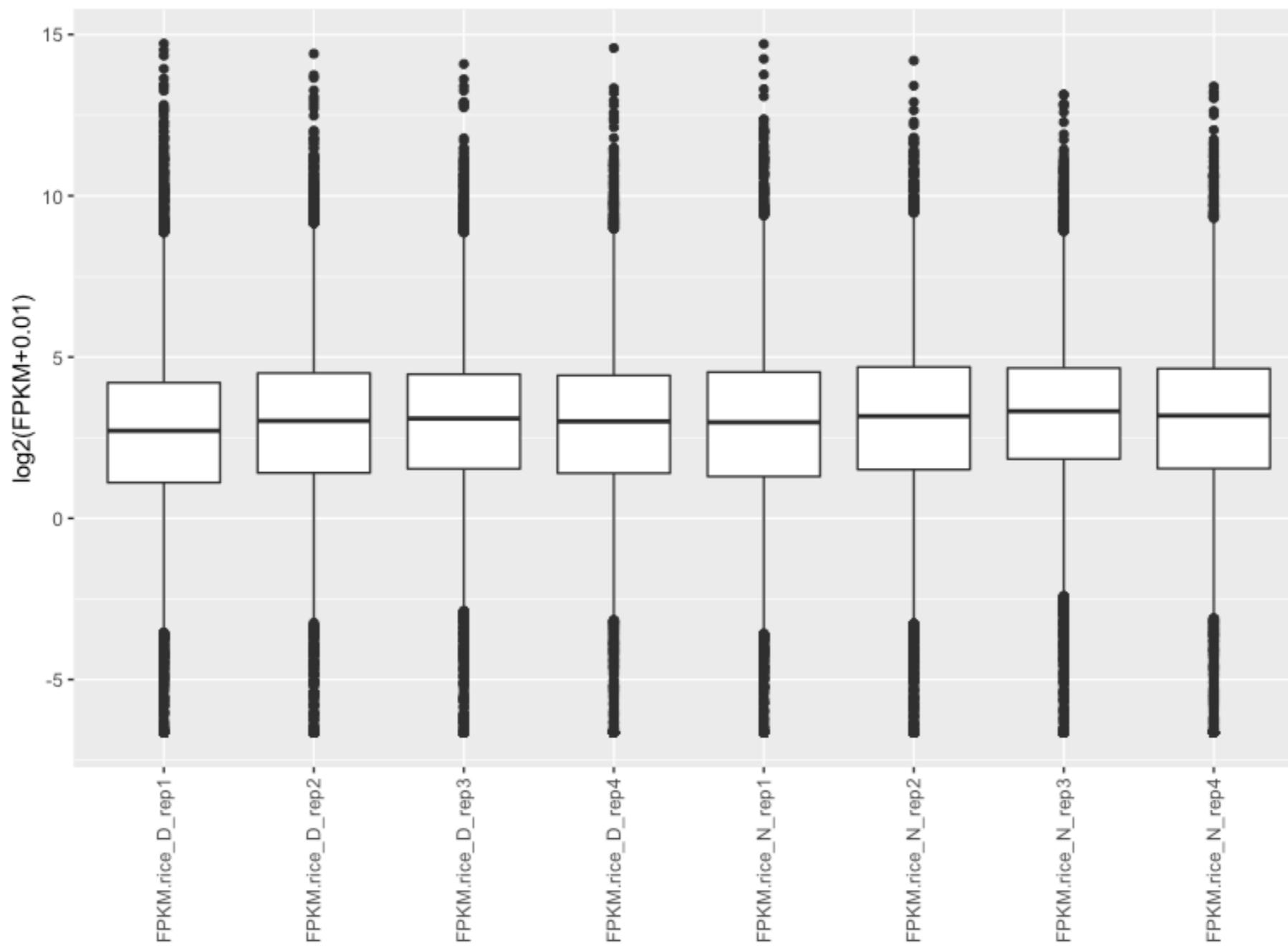
このような書式にしておくと、グラフ描画関数ggplotでのグラフ化が容易になる。

データ解析の大部分はこのようなデータのフィルタリングや整形などの地味な仕事。

# 遺伝子発現量の分布の可視化

箱ひげ図を描く。

```
> ggplot(log2fpkm.long, aes(y=FPKM, x=Dataset)) + geom_boxplot() +  
  theme(axis.text.x = element_text(angle=90, hjust=0, vjust=.5)) +  
  xlab("") + ylab("log2(FPKM+0.01)")
```



# 特定の遺伝子の構造や発現プロファイルを可視化する



ballgownオブジェクトに入れる際に振られた"id"で遺伝子を指定する必要があるため、遺伝子のIDや遺伝子名と「ballgownが割り振るID」との対応を調べる。

```
> subset(results_transcripts, results_transcripts$qval<0.01)
```

	geneNames	geneIDs	transcriptIDs	feature	id	fc	pval	qval
1	-	Os02g0623932	Os02t0623932-00	transcript	9347	1.334693e-01	3.016259e-07	0.003330334
2	LHY	Os04g0583900	<b>Os04t0583900-01</b>	transcript	<b>19641</b>	2.584726e+02	3.331834e-07	0.003330334
3	-	Os06g0660800	Os06t0660800-01	transcript	27355	6.321374e+00	6.792785e-07	0.003964490
4	OsBBX19, OsM, DTH2, qDTH-2	Os06g0298200	Os06t0298200-01	transcript	25939	6.919100e+01	9.295324e-07	0.003964490
5	PsbP	Os03g0279950	Os03t0279950-01	transcript	13109	2.380650e+01	9.915688e-07	0.003964490
6	-	Os08g0360100	Os08t0360100-01	transcript	32883	1.538616e-01	1.254840e-06	0.004073630
7	-	Os06g0281400	Os06t0281400-01	transcript	25839	3.943439e-01	1.426413e-06	0.004073630
...								
38	SIGA	Os08g0163400	<b>Os08t0163400-01</b>	transcript	<b>31942</b>	1.358213e-01	1.574039e-05	0.008169480
...								

- ballgown id: 19641 = LHY = Os04t0583900-01
- Ballgown id: 31942 = SIGA = Os08t0163400-01  
であることが分かる。

# 特定の遺伝子の構造や発現プロファイルを可視化する



LHY遺伝子（ballgown id: 19641）の発現量の箱ひげ図を作成し、個々のサンプルの発現量を個別に重ねてプロットする。

まずはballgown idを指定し、特定の転写産物の発現量データを取り出す。

```
> log2fpkm["19641", ]  
FPKM.rice_D_rep1 FPKM.rice_D_rep2 FPKM.rice_D_rep3 FPKM.rice_D_rep4  
-2.0672131      -0.7016209      -3.1568201      -4.5881052  
FPKM.rice_N_rep1 FPKM.rice_N_rep2 FPKM.rice_N_rep3 FPKM.rice_N_rep4  
8.2926856       8.4396024       7.9646139       8.1115949
```

# 特定の遺伝子の構造や発現プロファイルを可視化する

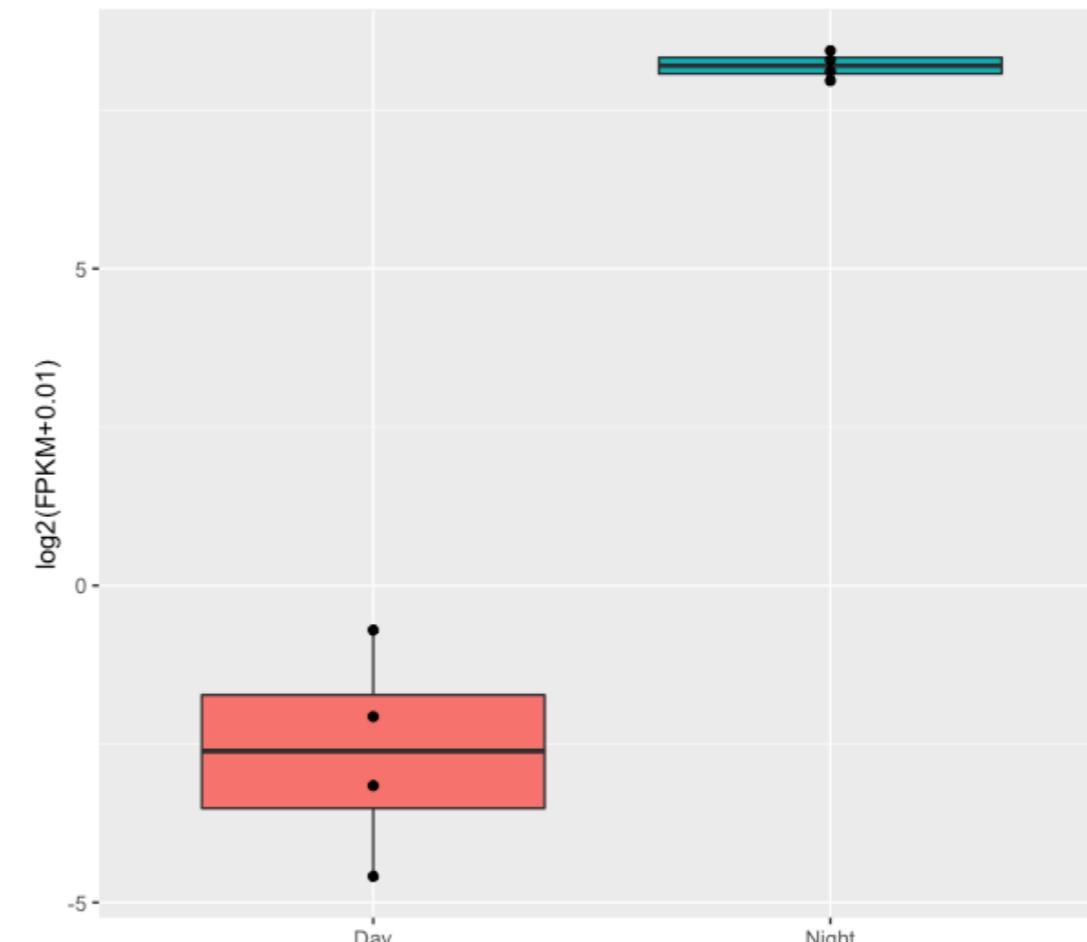
取り出した発現量データを整形し、ggplotで箱ひげ図と散布図を描画。

```
> log2fpkm.LHY.long <- as.data.frame(log2fpkm["19641", ]) %>%  
  tidyr::gather(Dataset, FPKM)  
> log2fpkm.LHY.long <- data.frame(log2fpkm.LHY.long, Condition=c(rep("Day", 4),  
rep("Night", 4)))  
> ggplot(log2fpkm.LHY.long, aes(x=Condition, y=FPKM, fill=Condition)) +  
  geom_boxplot() + geom_point() +  
  xlab("") + ylab("log2(FPKM+0.01)") + theme(legend.position = "bottom")
```

描画用のデータの確認

```
> log2fpkm.LHY.long
```

	Dataset	FPKM	Condition
1	log2fpkm["19641", ]	-2.0672131	Day
2	log2fpkm["19641", ]	-0.7016209	Day
3	log2fpkm["19641", ]	-3.1568201	Day
4	log2fpkm["19641", ]	-4.5881052	Day
5	log2fpkm["19641", ]	8.2926856	Night
6	log2fpkm["19641", ]	8.4396024	Night
7	log2fpkm["19641", ]	7.9646139	Night
8	log2fpkm["19641", ]	8.1115949	Night



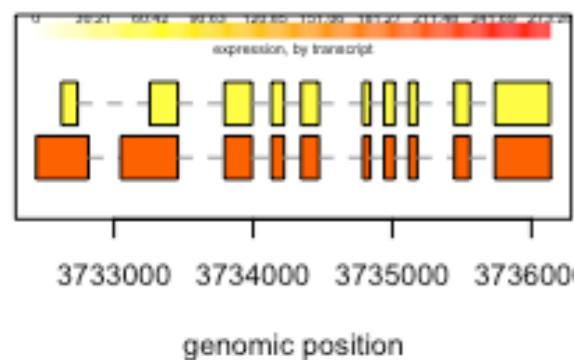
# 特定の遺伝子の構造や発現プロファイルを可視化する

sigA遺伝子 (ballgown id: 31942) の転写産物構造と発現量を合わせて可視化する。

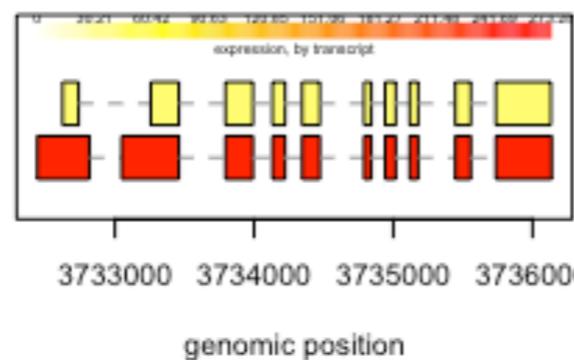
```
> plotTranscripts(ballgown::geneIDs(bg)[31942], bg, main=c('SIGA'),  
sample=c('rice_D_rep1','rice_D_rep2','rice_D_rep3','rice_D_rep4','rice_N_rep1',  
'rice_N_rep2','rice_N_rep3','rice_N_rep4'))
```

SIGA

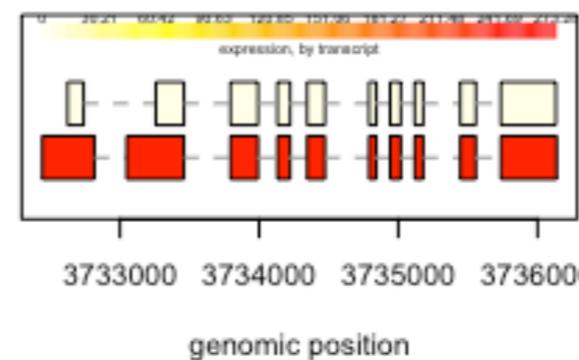
rice\_D\_rep1



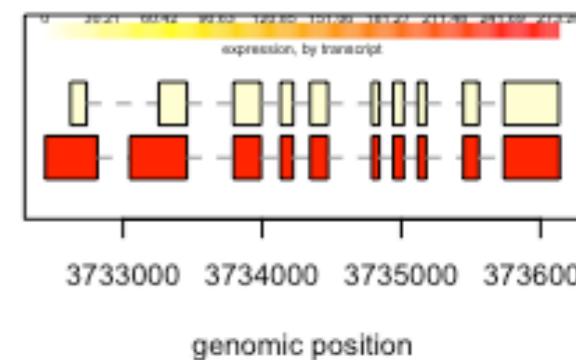
rice\_D\_rep2



rice\_D\_rep3

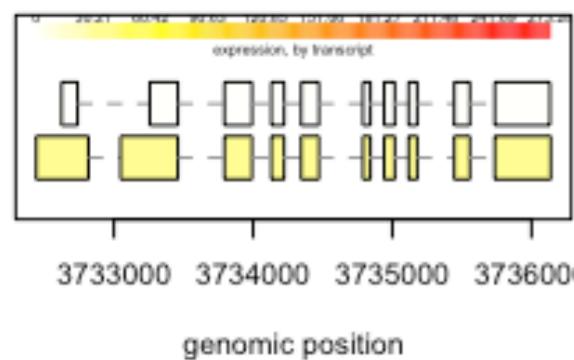


rice\_D\_rep4

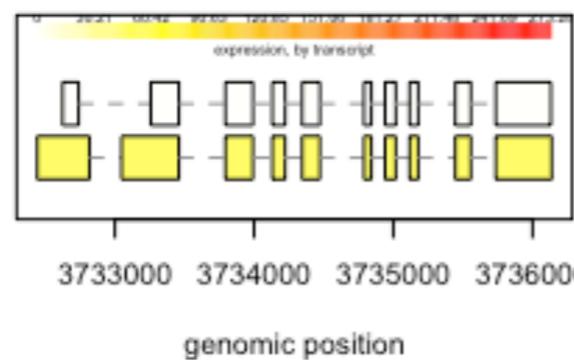


で

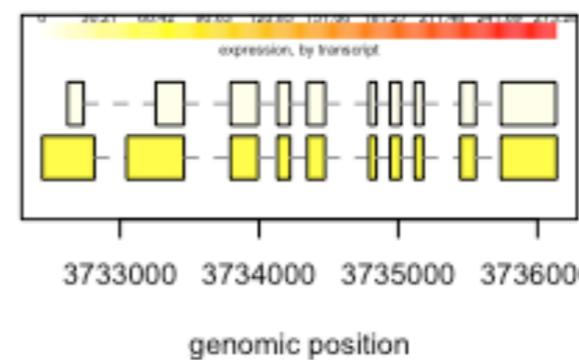
rice\_N\_rep1



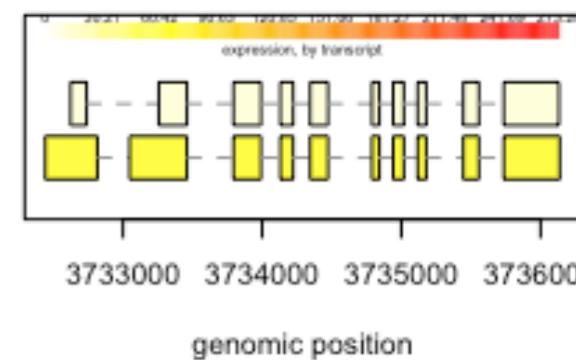
rice\_N\_rep2



rice\_N\_rep3



rice\_N\_rep4



転写調節に関わるイネのsigA遺伝子の発現量は昼に高く、夜に低いパターンを示し、2つあるsplicing isoformうちの一方が主に発現していることが分かる。

# その他の発現解析ツール

**StringTie**  
Transcript assembly and quantification for RNA-Seq

JOHNS HOPKINS UNIVERSITY  
CENTER FOR COMPUTATIONAL BIOLOGY  
CCB

Home Manual FAQ CCB » Software » StringTie

- Overview
- News
- Obtaining and installing StringTie
- Licensing and contact Information
- Publications

**Overview**

StringTie is a fast and highly efficient assembler of RNA-Seq alignments into potential transcripts. It uses a novel network flow algorithm as well as an optional *de novo* assembly step to assemble and quantitate full-length transcripts representing multiple splice variants for each gene locus. Its input can include not only alignments of short reads that can also be used by other transcript assemblers, but also alignments of longer sequences that have been assembled from those reads. In order to identify differentially expressed genes between experiments, StringTie's output can be processed by specialized software like Ballgown, Cuffdiff or other programs (DESeq2, edgeR, etc.).

<https://ccb.jhu.edu/software/stringtie/index.shtml>

- 発現変動遺伝子の検出や可視化には DESeq2やedgeRもよく使われている
- どちらもBallgownと同様にRのパッケージとして提供されている。
- StringTieに付属のスクリプトで入力データ（リードカウントデータ）を作成可能

## Using StringTie with DESeq2 and edgeR

DESeq2 and edgeR are two popular Bioconductor packages for analyzing differential expression, which take as input a matrix of read counts mapped to particular genomic features (e.g., genes). We provide a Python script (`prepDE.py`) to extract this read count information directly from the files generated by StringTie (run with the `-e` parameter).

`prepDE.py` derives hypothetical read counts for each transcript from the coverage values estimated by StringTie for each transcript, by using this simple formula:  $\text{reads\_per\_transcript} = \text{coverage} * \text{transcript\_len} / \text{read\_len}$

There are two ways to provide input to the `prepDE.py` script:

- one option is to provide a path to a directory containing all sample sub-directories, with the same structure as the *ballgown* directory in the [StringTie protocol paper](#) in preparation for Ballgown. By default (no `-i` option), the script is going to look in the current directory for all sub-directories having `.gtf` files in them, as in this example:

```
./sample1/sample1.gtf
./sample2/sample2.gtf
./sample3/sample3.gtf
```

- Alternatively, one can provide a text file listing sample IDs and their respective paths ([sample\\_lst.txt](#)).

Usage: `prepDE.py [options]`

generates two CSV files containing the count matrices for genes and transcripts, using the coverage values found in the output of `stringtie -e`

<https://ccb.jhu.edu/software/stringtie/index.shtml?t=manual>

# その他の発現解析ツール

使い方はググればたくさん出てきます

Google 検索結果 (DESeq2)

- bi.biopapyrus.jp › analysis › de-analysis › 2g-deseq2 › 二群間比較 (DESeq2) | 一般化線形モデルによる発現変動 ...
- DESeq2はRNA-seqのリードカウントデータから発現変動遺伝子を検出するためのパッケージである。一般化線形モデルをサポートしているため複雑な解析にも柔軟に対応できる。二群間比較に対しては、Wald検定と尤度検定の2種類の ...
- bioconductor.org › bioc › html › DE... このページを訳す
- DESeq2 - Bioconductor**
- DOI: 10.18129/B9.bioc.DESeq2. Differential gene expression analysis based on the negative binomial distribution. Bioconductor version: Release (3.11). Estimate variance-mean dependence in count data from high-throughput sequencing ...
- bioconductor.org › bioc › vignettes › DESeq2 › inst › doc
- Analyzing RNA-seq data with DESeq2 - Bioconductor**
- 2020/07/30 - DESeq2 provides a function collapseReplicates which can assist in combining the counts from technical replicates into single columns of the count matrix. The term technical replicate implies multiple sequencing runs of the ...
- 他の人はこちらも検索
  - DESeq2 使い方
  - DESeq2 install
  - DESeq2 heatmap
  - Tximport
  - Deseq2 GitHub
  - edgeR
- ncrna.jp › ブログ, 技術紹介 ...
- サルマップ2018(3) DESeq2による標準化と可視化まで - ノン ...
- 2018/06/09 - 前回作ったカウントデータがはいったリスト fc をそのまま引き継いで使います。
- # normalize with DESeq2 library(DESeq2) dt <- as.matrix(fc\$counts) dt1 <- dt[ apply(dt, ...
- mecobalamin.hatenablog.com › entry › 2019/06/25
- RNA-seqその9、DESeq2を使った正規化 - mecobalamin's diary
- 2019/06/25 - 1細胞RNAseqデータ解析のための統計学知識集 (English Edition)作者:ryamada 発売日: 2016/01/26メディア: Kindle版 以前edgeRでリードカウントの正規化をした同じ結果を DESeq2を使って正規化してみる 正規化の方法が ...
- qiita.com › RNA-seq
- StringTieで計算したデータをDESeq2に乗せるprepDE.py - Qiita
- 2020/02/03 - などと考えていたが、マニュアルにも記載があるように、DESeq2やedgeRというソフトも計算に利用できるらしい。今回はStringTieで出力されたGTFファイルからDESeq2用のファイルを書き出し、最終的にDESeq2の ...

## Analyzing RNA-seq data with DESeq2

Michael I. Love, Simon Anders, and Wolfgang Huber

08/24/2020

### Abstract

A basic task in the analysis of count data from RNA-seq is the detection of differentially expressed genes. The count data are presented as a table which reports, for each sample, the number of sequence fragments that have been assigned to each gene. Analogous data also arise for other assay types, including comparative ChIP-Seq, HiC, shRNA screening, and mass spectrometry. An important analysis question is the quantification and statistical inference of systematic changes between conditions, as compared to within-condition variability. The package DESeq2 provides methods to test for differential expression by use of negative binomial generalized linear models; the estimates of dispersion and logarithmic fold changes incorporate data-driven prior distributions. This vignette explains the use of the package and demonstrates typical workflows. An RNA-seq workflow on the Bioconductor website covers similar material to this vignette but at a slower pace, including the generation of count matrices from FASTQ files. DESeq2 package version: 1.29.13

#### Standard workflow

- Quick start
- How to get help for DESeq2
- Acknowledgments
- Input data
  - Why un-normalized counts?
  - The DESeqDataSet
  - Transcript abundance files and tximport / tximeta
  - Tximeta for import with automatic metadata
  - Count matrix input

### Count matrix input

Alternatively, the function `DESeqDataSetFromMatrix` can be used if you already have a matrix of read counts prepared from another source. Another method for quickly producing count matrices from alignment files is the `featureCounts` function (Liao, Smyth, and Shi 2013) in the `Rsubread` package. To use `DESeqDataSetFromMatrix`, the user should provide the counts matrix, the information about the samples (the columns of the count matrix) as a `DataFrame` or `data.frame`, and the design formula.

To demonstrate the use of `DESeqDataSetFromMatrix`, we will read in count data from the `pasilla` package. We read in a count matrix, which we will name `cts`, and the sample information table, which we will name `coldata`. Further below we describe how to extract these objects from, e.g. `featureCounts` output.

```
library("pasilla")
pasCts <- system.file("extdata",
                      "pasilla_gene_counts.tsv",
                      package="pasilla", mustWork=TRUE)
pasAnno <- system.file("extdata",
                      "pasilla_sample_annotation.csv",
                      package="pasilla", mustWork=TRUE)
cts <- as.matrix(read.csv(pasCts, sep="\t", row.names=1))
coldata <- read.csv(pasAnno, row.names=1)
coldata <- coldata[,c("condition", "type")]
coldata$condition <- factor(coldata$condition)
coldata$type <- factor(coldata$type)
```

We examine the count matrix and column data to see if they are consistent in terms of sample order.

`head(cts, 2)`

```
##          untreated1 untreated2 untreated3 untreated4 treated1 treated2
## FBgn0000003          0          0          0          0          0          0
## FBgn0000008         92         161         76         70        140         88
##                      treated3
## FBgn0000003          1
## FBgn0000008         70
```

`coldata`

```
##           condition      type
## treated1fb    treated single-read
## treated2fb    treated paired-end
## treated3fb    treated paired-end
## untreated1fb untreated single-read
## untreated2fb untreated single-read
## untreated3fb untreated paired-end
## untreated4fb untreated paired-end
```

# おまけ：DESeq2を用いた解析

## 解析の流れ

1. HISAT2によるアラインメント
2. StringTieによる発現量の定量
3. StringTie付属のprepDE.pyによるカウントデータの作成
4. DESeq2による発現データの可視化、発現変動解析

参考URL：<http://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html>

## Analyzing RNA-seq data with DESeq2

Michael I. Love, Simon Anders, and Wolfgang Huber

05/19/2021

### Abstract

A basic task in the analysis of count data from RNA-seq is the detection of differentially expressed genes. This is typically presented as a table which reports, for each sample, the number of sequence fragments that have been assigned to each gene. Analogous data also arise for other assay types, including comparative ChIP-Seq, HiC, shRNA screening, and RNA-seq. An important analysis question is the quantification and statistical inference of systematic changes between samples compared to within-condition variability. The package DESeq2 provides methods to test for differential expression using negative binomial generalized linear models; the estimates of dispersion and logarithmic fold changes incorporate prior distributions. This vignette explains the use of the package and demonstrates typical workflows. An RNA-seq vignette on the Bioconductor website covers similar material to this vignette but at a slower pace, including the generation of RNA-seq FASTQ files. DESeq2 package version: 1.32.0

- Standard workflow
  - Quick start
  - How to get help for DESeq2
  - Acknowledgments
  - Funding
  - Input data

### Count matrix input

Alternatively, the function `DESeqDataSetFromMatrix` can be used if you already have a matrix of read counts prepared from another source. Another method for quickly producing count matrices from alignment files is the `featureCounts` function (Liao, Smyth, and Shi 2013) in the `Rsubread` package. To use `DESeqDataSetFromMatrix`, the user should provide the counts matrix, the information about the samples (the columns of the count matrix) as a `DataFrame` or `data.frame`, and the design formula.

To demonstrate the use of `DESeqDataSetFromMatrix`, we will read in count data from the `pasilla` package. We read in a count matrix, which we will name `cts`, and the sample information table, which we will name `coldata`. Further below we describe how to extract these objects from, e.g. `featureCounts` output.

```
library("pasilla")
pasCts <- system.file("extdata",
                      "pasilla_gene_counts.tsv",
                      package="pasilla", mustWork=TRUE)
pasAnno <- system.file("extdata",
                      "pasilla_sample_annotation.csv",
                      package="pasilla", mustWork=TRUE)
cts <- as.matrix(read.csv(pasCts,sep="\t",row.names="gene_id"))
coldata <- read.csv(pasAnno, row.names=1)
coldata <- coldata[,c("condition","type")]
coldata$condition <- factor(coldata$condition)
coldata$type <- factor(coldata$type)
```

We examine the count matrix and column data to see if they are consistent in terms of sample order.

```
head(cts,2)
```

	untreated1	untreated2	untreated3	untreated4	treated1	treated2
## FBgn0000003	0	0	0	0	0	0
## FBgn0000008	92	161	76	70	140	88
## treated3						
## FBgn0000003	1					
## FBgn0000008	70					

# Step5. prepDE.pyによるカウントデータの作成

StringTieに付属するprepDE.pyを使って、StringTieの結果ファイルからリードカウントデータのマトリックスを作成するスクリプト。

```
$ less step5_make_count_matrix.sh
```

- step5\_make\_count\_matrix.sh -

```
ToolDir=$HOME/RNA-Seq/tool
StringTie_bin=$ToolDir/stringtie-2.1.6.Linux_x86_64

export PATH=$StringTie_bin:$PATH

### Step5. Make count matrix

prepDE.py -l 101 -i ballgown/ -p rice_
```

平均リード長、StringTieの結果があるディレクトリ、  
サンプル名のPrefixを指定している。

# Step5. prepDE.pyによるカウントデータの作成

シェルスクリプトの実行 (実行時間：約1秒)

```
$ bash ./step5_make_count_matrix.sh
```

遺伝子、転写産物ごとのリードカウントデータが出力されている。

```
$ ls -lhtr  
...  
transcript_count_matrix.csv      gene_count_matrix.csv
```

遺伝子、サンプルごとにリードカウントがカンマ区切りで保存されている。

```
$ less gene_count_matrix.csv  
gene_id,rice_D_rep1,rice_D_rep2,rice_D_rep3,rice_N_rep1,rice_N_rep2,rice_N_rep3  
Os02g0756800|-,658,116,18,343,97,2  
Os02g0736200|RDR1,250,242,584,355,753,500  
Os02g0731900|-,75,113,511,72,89,465  
Os02g0744000|-,996,1126,1414,17696,15949,13622
```

# おまけ：DESeq2を用いた解析

## 解析の流れ

1. HISAT2によるアラインメント
2. StringTieによる発現量の定量
3. StringTie付属のprepDE.pyによるカウントデータの作成
4. DESeq2による発現データの可視化、発現変動解析

参考URL : <http://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html>

## Analyzing RNA-seq data with DESeq2

Michael I. Love, Simon Anders, and Wolfgang Huber

05/19/2021

### Abstract

A basic task in the analysis of count data from RNA-seq is the detection of differentially expressed genes. This is typically presented as a table which reports, for each sample, the number of sequence fragments. Analogous data also arise for other assay types, including comparative ChIP-Seq, RNA-seq, and microarray analysis. An important analysis question is the quantification and statistical inference of systematic differences between samples, compared to within-condition variability. The package DESeq2 provides methods to fit negative binomial generalized linear models; the estimates of dispersion and log-ratio distributions. This vignette explains the use of the package and demonstrates typical analyses. The Bioconductor website covers similar material to this vignette but at a slower pace, and includes more detailed descriptions of the methods. DESeq2 package version: 1.32.0

- Standard workflow

- Quick start
- How to get help for DESeq2
- Acknowledgments
- Funding
- Input data
  - Why un-normalized counts?
  - The DESeqDataSet
  - Transcript abundance files and tximport / tximeta
  - Tximeta for import with automatic metadata
  - Count matrix input

### Count matrix input

Alternatively, the function `DESeqDataSetFromMatrix` can be used if you already have a matrix of read counts prepared from another source.

Another method for quickly producing count matrices from alignment files is the `featureCounts` function (Liao, Smyth, and Shi 2013) in the `Rsubread` package. To use `DESeqDataSetFromMatrix`, the user should provide the counts matrix, the information about the samples (the columns of the count matrix) as a `DataFrame` or `data.frame`, and the design formula.

To demonstrate the use of `DESeqDataSetFromMatrix`, we will read in count data from the `pasilla` package. We read in a count matrix, which we will name `cts`, and the sample information table, which we will name `coldata`. Further below we describe how to extract these objects from, e.g. `featureCounts` output.

```
library("pasilla")
pasCts <- system.file("extdata",
                      "pasilla_gene_counts.tsv",
                      package="pasilla", mustWork=TRUE)
pasAnno <- system.file("extdata",
                      "pasilla_sample_annotation.csv",
                      package="pasilla", mustWork=TRUE)
cts <- as.matrix(read.csv(pasCts,sep="\t",row.names="gene_id"))
coldata <- read.csv(pasAnno, row.names=1)
coldata <- coldata[,c("condition","type")]
coldata$condition <- factor(coldata$condition)
coldata$type <- factor(coldata$type)
```

We examine the count matrix and column data to see if they are consistent in terms of sample order.

```
head(cts,2)
```

	untreated1	untreated2	untreated3	untreated4	treated1	treated2
## FBgn0000003	0	0	0	0	0	0
## FBgn0000008	92	161	76	70	140	88
## treated3						
## FBgn0000003	1					
## FBgn0000008	70					

# おまけ：DESeq2を用いた解析

「useref\_analysis\_workflow.R」の後半にDESeq2を用いた発現解析のスクリプトを載せてあるので、時間があれば順番に実行してみましょう。

## Differential expression analysis

The standard differential expression analysis steps are wrapped into a single function, *DESeq*. The estimation steps performed by this function are described [below](#), in the manual page for `?DESeq` and in the Methods section of the DESeq2 publication (Love, Huber, and Anders 2014).

Results tables are generated using the function *results*, which extracts a results table with log2 fold changes, *p* values and adjusted *p* values. With no additional arguments to *results*, the log2 fold change and Wald test *p* value will be for the **last variable** in the design formula, and if this is a factor, the comparison will be the **last level** of this variable over the **reference level** (see previous [note on factor levels](#)). However, the order of the variables of the design do not matter so long as the user specifies the comparison to build a results table for, using the `name` or `contrast` arguments of *results*.

Details about the comparison are printed to the console, directly above the results table. The text, `condition treated vs untreated`, tells you that the estimates are of the logarithmic fold change  $\log_2(\text{treated}/\text{untreated})$ .

```
dds <- DESeq(dds)
res <- results(dds)
res
```

### Heatmap of the sample-to-sample distances

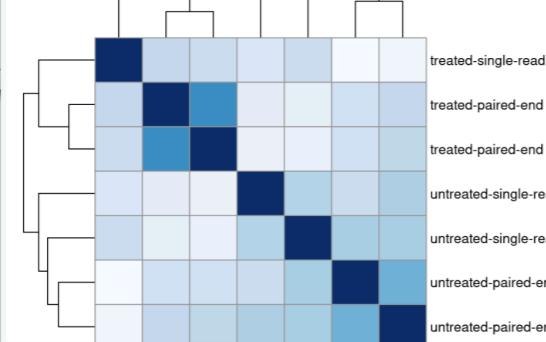
Another use of the transformed data is sample clustering. Here, we apply the *dist* function to the transpose of the transformed count matrix to get sample-to-sample distances.

```
## log2 fold c
## Wald test p
## DataFrame w:
##
## FBgn0000008
## FBgn0000014
## FBgn0000017
## FBgn0000018
## FBgn0000024
## ...
## FBgn0261570
## FBgn0261572
## FBgn0261573
## FBgn0261574
## FBgn0261575
```

```
sampleDists <- dist(t(assay(vsd)))
```

A heatmap of this distance matrix gives us an overview over similarities and dissimilarities between samples. We have to provide a hierarchical clustering `hc` to the heatmap function based on the sample distances, or else the heatmap function would calculate a clustering based on the distances between the rows/columns of the distance matrix.

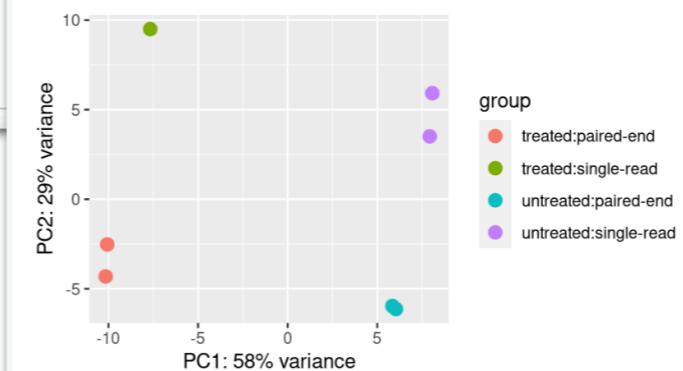
```
library("RColorBrewer")
sampleDistMatrix <- as.matrix(sampleDists)
rownames(sampleDistMatrix) <- paste(vsd$condition, vsd$type, sep="-")
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
heatmap(sampleDistMatrix,
        clustering_distance_rows=sampleDists,
        clustering_distance_cols=sampleDists,
        col=colors)
```



### Principal component plot of the samples

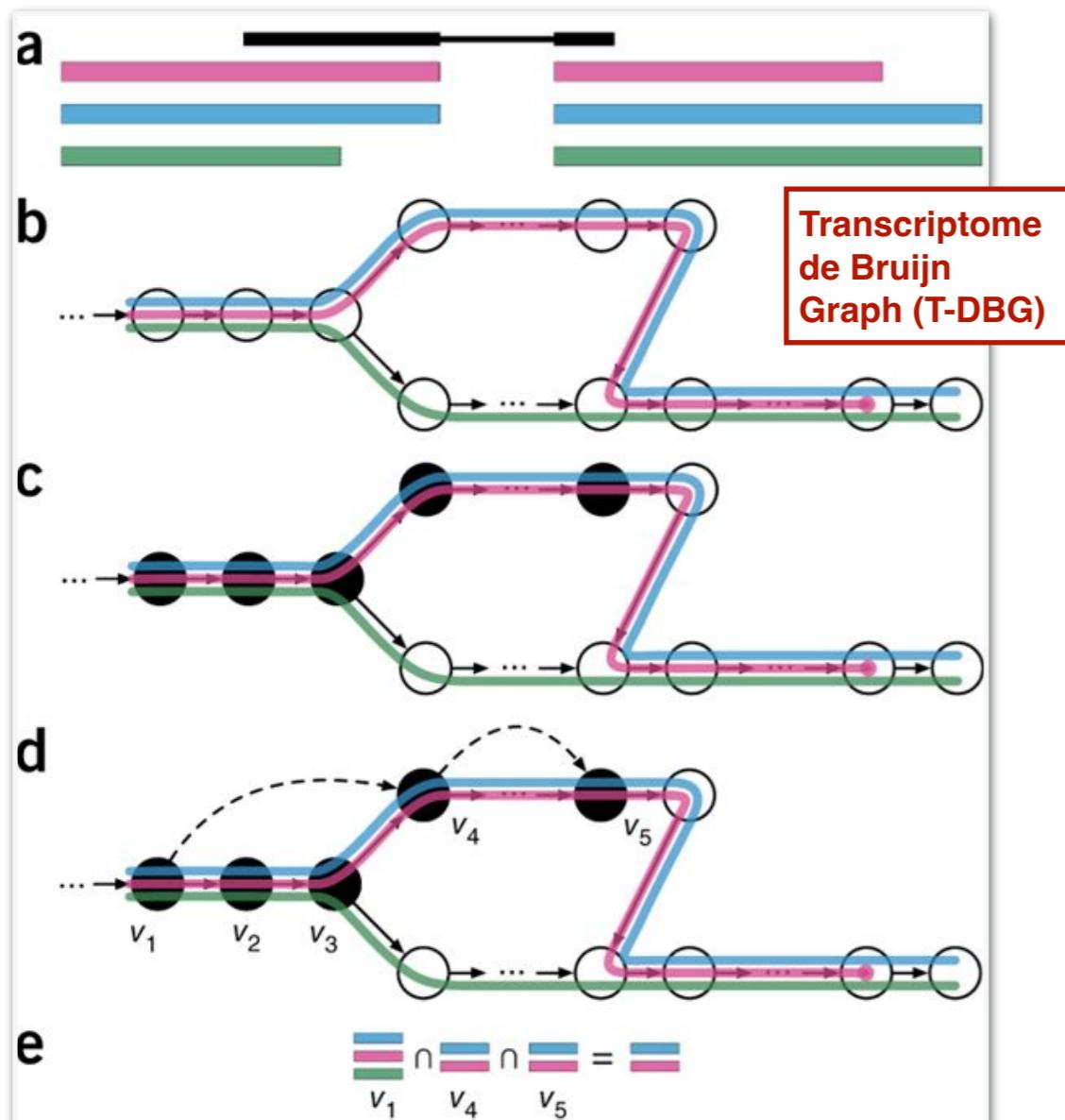
Related to the distance matrix is the PCA plot, which shows the samples in the 2D plane spanned by their first two principal components. This type of plot is useful for visualizing the overall effect of experimental covariates and batch effects.

```
plotPCA(vsd, intgroup=c("condition", "type"))
```



# おまけ：Kallistoによるアラインメントフリーの発現解析

「全転写産物配列」と「RNA-Seqリードデータ」のみを用いて、アラインメントせず！に発現量を定量する方法。**計算速度がとにかく早い。**



**a-b:** 全転写産物配列をkmerに分解し、T-DBGを構築。

**c-e:** RNA-Seqリードも同様にkmerに分解し、T-DBG上のどのパスに由来するか照合、転写産物に対応付け、発現量を推定する。

Kallistoによるトランскルiptーム解析用  
ディレクトリ「noaln」に移動

```
$ cd ~/RNA-Seq/noaln
```

Figure 1: Overview of kallisto  
From Bray NL et al. 2016 Nature Biotech.

# おまけ：Kallistoによるアライメントフリーの発現解析



Kallistoを使って発現量を定量するシェルスクリプト

```
$ less ./run_kallisto.sh
```

- run\_kallisto.sh -

```
DataDir=$HOME/RNA-Seq/data  
ToolDir=$HOME/RNA-Seq/tool  
Kallisto_bin=$ToolDir/kallisto  
  
### Step1. Make transcriptome index  
$Kallisto_bin/kallisto index -i transcript_index $DataDir/transcript.fasta  
  
### Step2. Quantification  
$Kallisto_bin/kallisto quant -i transcript_index -o output ../useref/  
rice_D_rep1_r1.pe.fastq.gz ../useref/rice_D_rep1_r2.pe.fastq.gz
```

T-DBGの構築

RNA-Seqリードの対応付けと発現量推定

ここでは1サンプル (rice\_D\_rep1) のみ発現解析を行っている。

シェルスクリプトを実行

```
$ bash ./run_kallisto.sh 2>&1 | tee run_kallisto.log
```

(実行時間：約1秒)

# おまけ：Kallistoによるアラインメントフリーの発現解析



解析が終わるとoutputディレクトリが作成され、  
その中に発現量データ等の結果が出力される

```
$ less output/abundance.tsv
```

## - abundance.tsv -

target_id	length	eff_length	est_counts	tpm
Os02t0722500-01	1436	1275.73	43	200.913
Os02t0722600-01	487	327.326	0	0
Os02t0722650-00	531	371.12	13	208.797
Os02t0722700-01	1436	1275.73	900	4205.15
Os02t0722800-01	4485	4324.73	161.034	221.951
Os02t0722800-02	1316	1155.73	187.966	969.438
Os02t0723200-01	1526	1365.73	3	13.0935
Os02t0723300-01	890	729.8	211	1723.36
Os02t0723400-01	716	555.868	1	10.7232
Os02t0723400-02	1003	842.727	7	49.5117
Os02t0723600-00	903	742.727	0	0

- 各転写産物のTPM値が出力されている。
- サンプル間で発現量が異なる遺伝子（DEG）の検出用に、専用のRパッケージ「sleuth」が開発されている。

- ▶ リファレンス情報を用いたRNA-Seq解析
- ▶ **RNA-Seqデータを用いた多型検出**

# RNA-Seqデータを用いた多型検出の特徴

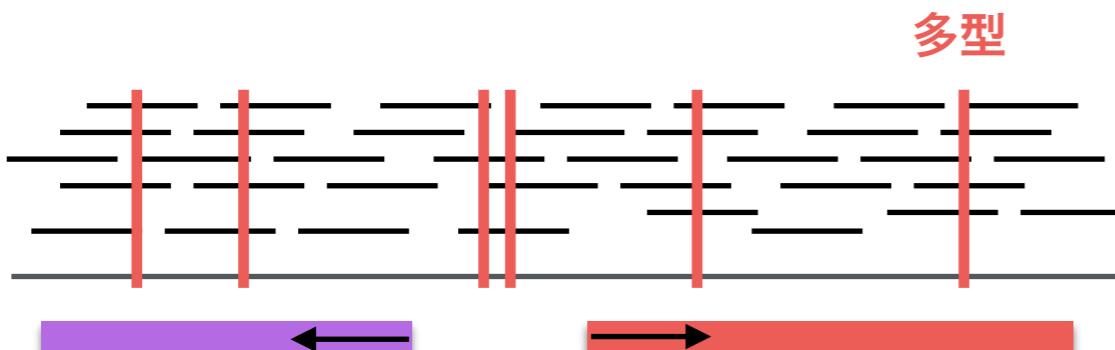
## 良い点

- ・ 巨大なゲノムでも効率よく多型情報が得られる。
- ・ 遺伝子発現量と多型情報が同時に得られる。
- ・ 転写領域は進化的に保存されており、進化的距離の離れた種間の比較でも効率よく多型情報が得られる。

## 悪い点

- ・ 低発現、または発現していない遺伝子の多型情報は得られない。
- ・ 遺伝子発現解析と同時にできるが、転写開始点上流の発現調節領域の多型情報は得られない。
- ・ アラインメント位置によるリードの偏りが大きい

## ゲノムリシーケンスデータ



## RNA-Seqデータ

検出できない多型  
(非転写領域)



検出可能な多型

目的：RNA-Seqデータを用いて日本晴とコシヒカリの間のゲノム配列の違い（多型）を調べる

日本晴  
(リファレンスゲノム)

VS



コシヒカリ (RNA-Seq)



12:00 PM



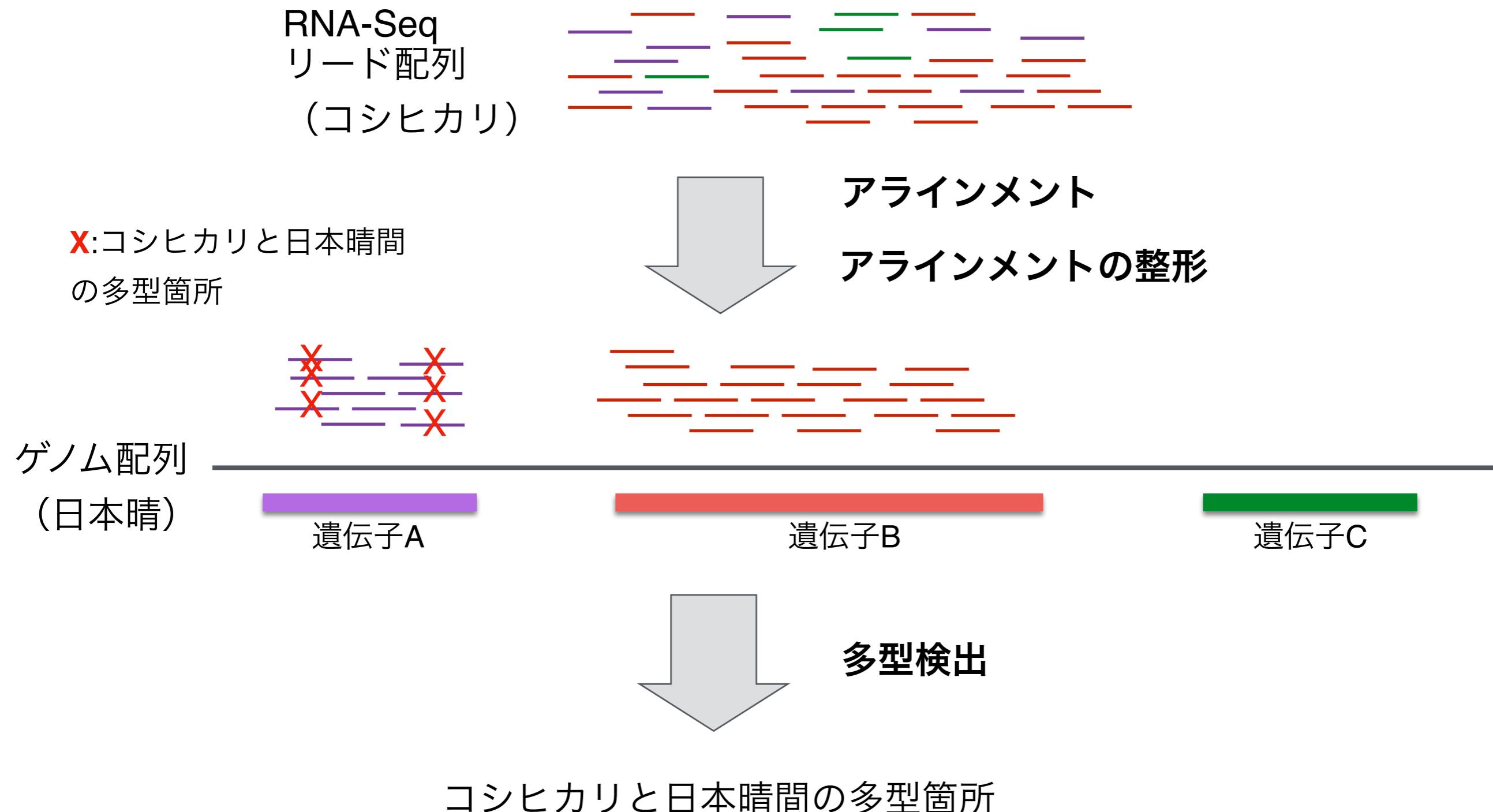
00:00 AM

- 田んぼで栽培する、イネ（コシヒカリ）の葉身のサンプル。
- 3つの異なる生育ステージ（3反復）において、**昼（12時）** と **夜（0時）** にサンプリング（2条件）。
- Illumina社の**Stranded mRNA-Seq法**でライブラリ調製。
- Illumina社のHiSeq2000による、**101bpのPaired-endシーケンシング**。

\*全データでは解析に時間がかかるため、2番染色体の特定の領域（2Mbp）にアラインメントされるリードだけを事前に抽出しており、演習ではそれを利用する。

# RNA-Seqデータを用いた多型検出の流れ

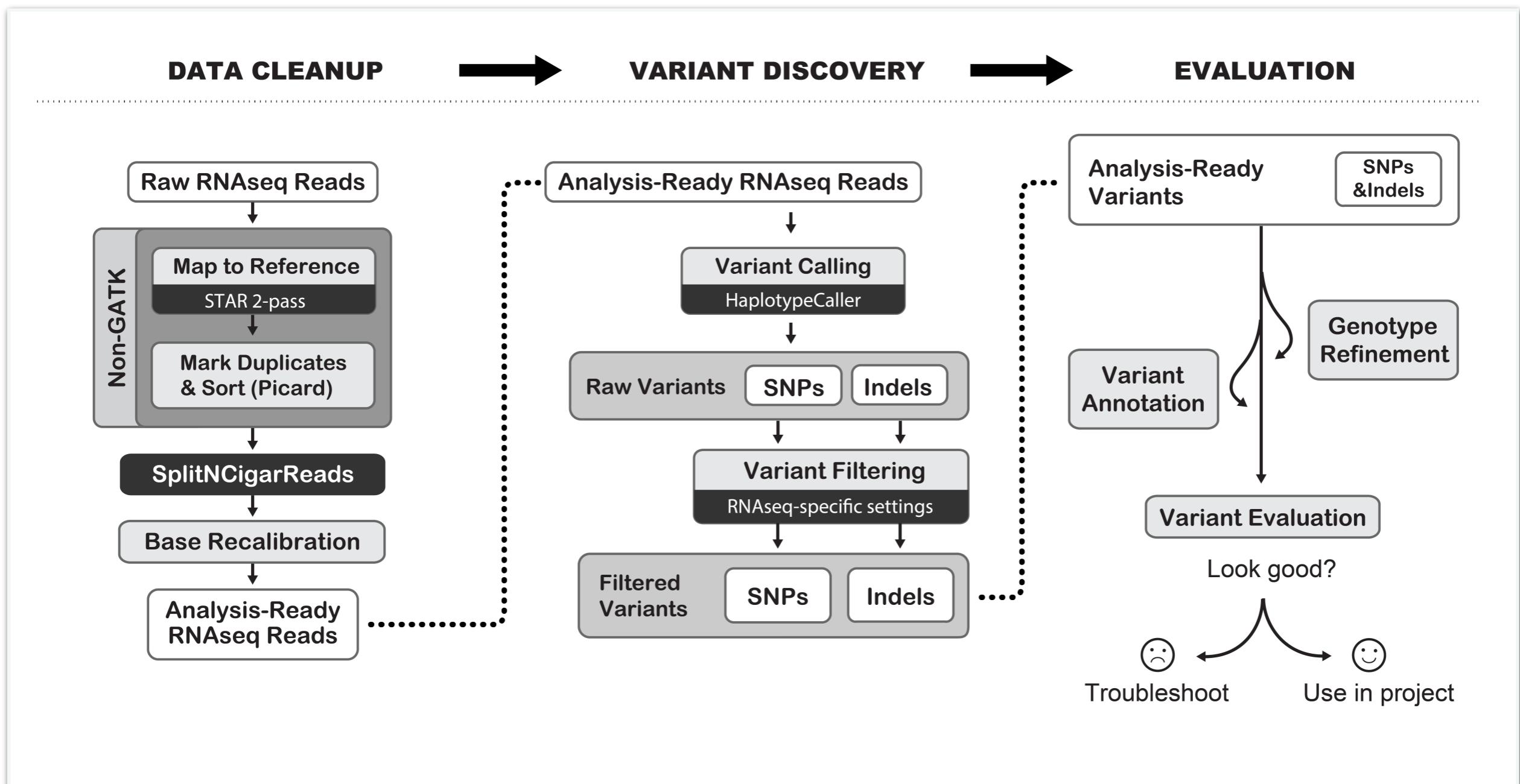
リファレンスとは異なる種や品種由来のRNA-Seqリードをゲノム配列にアラインメント（マッピング）し、種間や品種間の遺伝子配列の違い（多型情報）を得る。



# GATKのウェブサイトのBest Practices

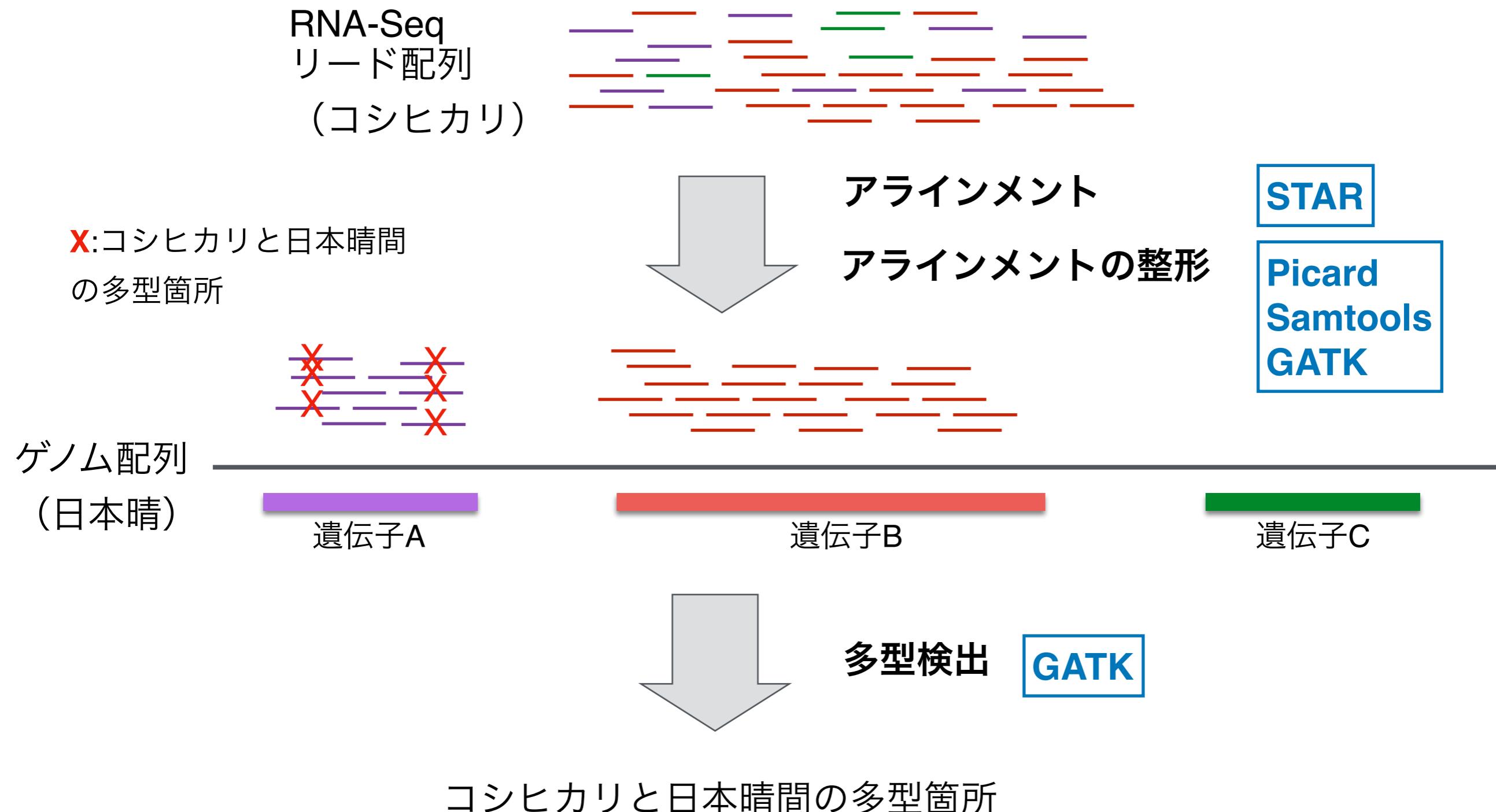


<https://gatk.broadinstitute.org/hc/en-us/articles/360035531192-RNAseq-short-variant-discovery-SNPs-Indels->



# RNA-Seqデータを用いた多型検出の流れ

リファレンスとは異なる種や品種由来のRNA-Seqリードをゲノム配列にアラインメント（マッピング）し、種間や品種間の遺伝子配列の違い（多型情報）を得る。



# Step1. STARのためのインデックス作成

RNA-Seqデータを用いた多型検出用ディレクトリ「varcall」に移動

```
$ cd ~/RNA-Seq/varcall
```

解析用ディレクトリ内のファイルを確認

```
$ ls
step1_make_STAR_index.sh      step3_BAM_Processing.sh
step2_STAR.sh                  step4_GATK.sh
```

シェルスクリプトが4つ用意されており、順番にシェルスクリプトを実行することで解析が進められるようになっている。

```
$ bash ./step1_make_STAR_index.sh
$ bash ./step2_STAR.sh
$ bash ./step3_BAM_Processing.sh
$ bash ./step4_GATK.sh
```

# Step1. STARのためのインデックス作成

STAR用のゲノム配列のインデックスを作成するシェルスクリプト

```
$ less step1_make_STAR_index.sh
```

- step1\_make\_STAR\_index.sh -

```
DataDir=$HOME/RNA-Seq_whole_genome/data  
ToolDir=$HOME/RNA-Seq_whole_genome/tool  
STAR_bin=$ToolDir/STAR_2.7.10b/Linux_x86_64_static
```

```
export PATH=$STAR_bin:$PATH
```

```
### Step1. Make genome index for STAR
```

```
GenomeDir=STAR_index  
mkdir $GenomeDir
```

```
STAR --runMode genomeGenerate \
```

..... インデックス作成モードを指定してSTARを実行

```
--genomeFastaFiles $DataDir/genome.fa \
```

..... ゲノム配列 (FASTA) の指定

```
--genomeDir $GenomeDir \
```

..... インデックス出力ディレクトリの指定

```
--sjdbGTFfile $DataDir/annotation.gtf \
```

..... 遺伝子アノテーション (GTF) の指定

```
--sjdbOverhang 100 --runThreadN 1
```

..... sjdbOverhangは (リード長-1) を指定

パス等の設定

インデックスの出力

ディレクトリの作成

# Step1. STARのためのインデックス作成

シェルスクリプトの実行 (実行時間：約1分)

```
$ bash ./step1_make_STAR_index.sh 2>&1 | tee step1_make_STAR_index.log
```

STAR\_indexディレクトリ中に作成されたインデックスファイルの確認

```
$ ls -lh STAR_index/
合計 1.8G
-rw-rw-r-- 1 guest01 guest01 35M 11月 11 14:04 Genome
-rw-rw-r-- 1 guest01 guest01 4.5K 11月 11 14:04 Log.out
-rw-rw-r-- 1 guest01 guest01 285M 11月 11 14:04 SA
-rw-rw-r-- 1 guest01 guest01 1.5G 11月 11 14:04 SAindex
-rw-rw-r-- 1 guest01 guest01 9 11月 11 14:03 chrLength.txt
-rw-rw-r-- 1 guest01 guest01 6 11月 11 14:03 chrName.txt
-rw-rw-r-- 1 guest01 guest01 15 11月 11 14:03 chrNameLength.txt
-rw-rw-r-- 1 guest01 guest01 11 11月 11 14:03 chrStart.txt
-rw-rw-r-- 1 guest01 guest01 41K 11月 11 14:03 exonGeTrInfo.tab
-rw-rw-r-- 1 guest01 guest01 20K 11月 11 14:03 exonInfo.tab
-rw-rw-r-- 1 guest01 guest01 13K 11月 11 14:03 geneInfo.tab
-rw-rw-r-- 1 guest01 guest01 714 11月 11 14:04 genomeParameters.txt
-rw-rw-r-- 1 guest01 guest01 26K 11月 11 14:04 sjdbInfo.txt
-rw-rw-r-- 1 guest01 guest01 29K 11月 11 14:03 sjdbList.fromGTF.out.tab
-rw-rw-r-- 1 guest01 guest01 26K 11月 11 14:04 sjdbList.out.tab
-rw-rw-r-- 1 guest01 guest01 20K 11月 11 14:03 transcriptInfo.tab
```

# Step2. STARによるRNA-Seqリードのアラインメント



STARによってRNA-Seqリードをゲノム配列にアラインメントするシェルスクリプト

```
$ less step2_STAR.sh
```

- step2\_STAR.sh -

```
DataDir=$HOME/RNA-Seq/data  
ToolDir=$HOME/RNA-Seq/tool  
STAR_bin=$ToolDir/STAR_2.7.10b/Linux_x86_64_static  
  
export PATH=$STAR_bin:$PATH  
  
### Step2. Alignment of RNA-Seq reads by STAR  
GenomeDir=STAR_index  
Read1_list=$DataDir/rice_D_rep1_r1.org.fastq.gz,$DataDir/  
rice_D_rep2_r1.org.fastq.gz,$DataDir/rice_D_rep3_r1.org.fastq.gz,$DataDir/  
rice_N_rep1_r1.org.fastq.gz,$DataDir/rice_N_rep2_r1.org.fastq.gz,$DataDir/  
rice_D_rep3_r1.org.fastq.gz  
Read2_list=$DataDir/rice_D_rep1_r2.org.fastq.gz,$DataDir/  
rice_D_rep2_r2.org.fastq.gz,$DataDir/rice_D_rep3_r2.org.fastq.gz,$DataDir/  
rice_N_rep1_r2.org.fastq.gz,$DataDir/rice_N_rep2_r2.org.fastq.gz,$DataDir/  
rice_D_rep3_r2.org.fastq.gz  
  
STAR --genomeDir $GenomeDir --readFilesCommand "gunzip -c" \  
--readFilesIn $Read1_list $Read2_list \  
--alignIntronMin 20 --alignIntronMax 10000 \  
--outSAMtype BAM SortedByCoordinate \  
--runThreadN 1
```

パス等の設定

全サンプルをまとめて  
変数に格納

リードファイルの圧縮形式に応じて指定

STARによる  
アラインメントを実行

# Step2. STARによるRNA-Seqリードのアラインメント



シェルスクリプトの実行 (実行時間：約1分)

```
$ bash ./step2_STAR.sh 2>&1 | tee step2_STAR.log
```

作成されたアラインメントファイルの確認

```
$ ls -lhtr
合計 133M
...
-rw-rw-r-- 1 guest01 guest01 59K 11月 11 14:10 SJ.out.tab
-rw-rw-r-- 1 guest01 guest01 133M 11月 11 14:10 Aligned.sortedByCoord.out.bam
-rw-rw-r-- 1 guest01 guest01 364 11月 11 14:10 Log.progress.out
-rw-rw-r-- 1 guest01 guest01 1.2K 11月 11 14:10 step2_STAR.log
-rw-rw-r-- 1 guest01 guest01 12K 11月 11 14:10 Log.out
-rw-rw-r-- 1 guest01 guest01 2.0K 11月 11 14:10 Log.final.out
```

すべてのサンプルをまとめて1サンプルとしてアラインメントしたので、1つの BAMファイル (Aligned.sortedByCoord.out.bam) が出力されている。

# Step3. アラインメントデータの加工

PicardやGATKでアラインメントデータを加工し、下流の解析に適したものにするシェルスクリプト。

```
$ less step3_BAM_Processing.sh
```

- step3\_BAM\_Processing.sh の前半 -

```
DataDir=$HOME/RNA-Seq/data  
ToolDir=$HOME/RNA-Seq/tool  
Picard_bin=$ToolDir/picard-2.27.5  
Samtools_bin=$ToolDir/samtools-1.13  
GATK_bin=$ToolDir/gatk-4.2.6.1  
  
Export PATH=$Samtools_bin:$GATK_bin:$PATH  
  
### Step3. Processing of alignment (BAM) file  
java -jar $Picard_bin/picard.jar AddOrReplaceReadGroups \  
  -I Aligned.sortedByCoord.out.bam \  
  -O Aligned.sortedByCoord.RG.out.bam \  
  -SO coordinate \  
  -RGID Koshihikari -RGLB TruSeq_RNA_stranded -RGPL illumina \  
  -RGPU HiSeq2000 -RGSM Koshihikari
```

パス等の設定

アラインメントデータにRG (read group)  
情報を追加した上で、位置でソートする。

# Step3. アラインメントデータの加工

- step3\_BAM\_Processing.shのつづき -

```
java -jar $Picard_bin/picard.jar MarkDuplicates \
-I Aligned.sortedByCoord.RG.out.bam \
-O Aligned.sortedByCoord.RG.MD.out.bam \
-CREATE_INDEX true \
-VALIDATION_STRINGENCY SILENT \
-M Aligned.sortedByCoord.RG.MD.metrics
```

PCRによって重複した（冗長になった）リードを取り除く

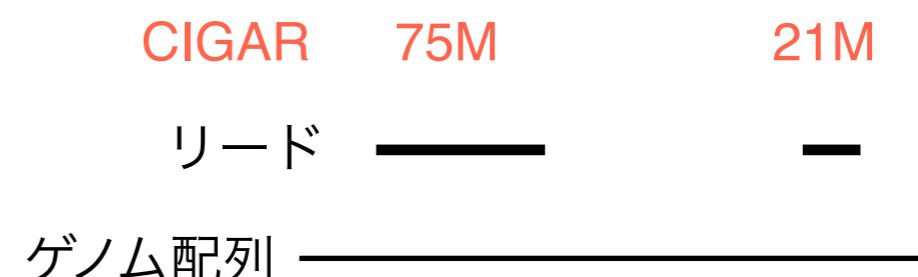
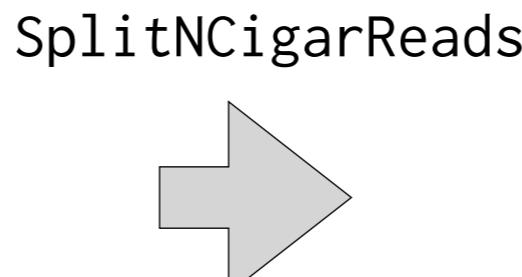
```
samtools faidx $DataDir/genome.fa
```

```
java -jar $Picard_bin/picard.jar CreateSequenceDictionary \
-R $DataDir/genome.fa \
-O $DataDir/genome.dict
```

ゲノム配列のindexやdictionaryの作成

```
gatk SplitNCigarReads \
-R $DataDir/genome.fa \
-I Aligned.sortedByCoord.RG.MD.out.bam \
-O Aligned.sortedByCoord.RG.MD.SplitN.out.bam
```

アラインメント中のイントロン領域情報を除く



# Step3. アラインメントデータの加工



シェルスクリプトの実行 (実行時間：約2分)

```
$ bash ./step3_BAM_Processing.sh 2>&1 | tee step3_BAM_Processing.log
```

作成されたアラインメントファイルの確認

```
$ ls -lhtr
合計 562M
...
-rw-rw-r-- 1 guest01 guest01 118M 11月 11 14:17 Aligned.sortedByCoord.RG.out.bam
-rw-rw-r-- 1 guest01 guest01 120M 11月 11 14:17 Aligned.sortedByCoord.RG.MD.out.bam
-rw-rw-r-- 1 guest01 guest01 23K 11月 11 14:17 Aligned.sortedByCoord.RG.MD.out.bai
-rw-rw-r-- 1 guest01 guest01 4.6K 11月 11 14:17 Aligned.sortedByCoord.RG.MD.metrics
-rw-rw-r-- 1 guest01 guest01 192M 11月 11 14:19 Aligned.sortedByCoord.RG.MD.SplitN.out.bam
-rw-rw-r-- 1 guest01 guest01 22K 11月 11 14:19 Aligned.sortedByCoord.RG.MD.SplitN.out.bai
-rw-rw-r-- 1 guest01 guest01 12K 11月 11 14:19 step3_BAM_Processing.log
```

各ステップの出力ファイルが複数存在するが、最終アラインメントは「Aligned.sortedByCoord.RG.MD.SplitN.out.bam」である。

# Step4. GATK HaplotypeCallerによる多型検出



GATK HaplotypeCallerによってRNA-Seqリードのアラインメントを元に多型(SNP、InDel)を検出するシェルスクリプト

```
$ less step4_GATK.sh
```

- step4\_GATK.sh -

```
DataDir=$HOME/RNA-Seq/data  
ToolDir=$HOME/RNA-Seq/tool  
GATK_bin=$ToolDir/gatk-4.2.6.1
```

```
export PATH=$GATK_bin:$PATH
```

```
### Step4. Detection of variations by GATK HaplotypeCaller
```

```
gatk HaplotypeCaller -R $DataDir/genome.fa -I  
Aligned.sortedByCoord.RG.MD.SplitN.out.bam \  
-stand-call-conf 20 --dont-use-soft-clipped-bases \  
-O variation.vcf.gz
```

HaplotypeCallerによる多型検出

```
gatk VariantFiltration -R $DataDir/genome.fa -V variation.vcf.gz \  
-window 20 -cluster 3 \  
--filter-name "QD" --filter "QD < 2.0" \  
--filter-name "FS" --filter "FS > 60.0" \  
-O variation.filter.vcf.gz
```

複数の条件での多型のフィルタリング

フィルタリングで指定しているQDやFSについては以下のページが参考になる

<https://gatk.broadinstitute.org/hc/en-us/articles/360035890471?id=11069>

# Step4. GATK HaplotypeCallerによる多型検出



シェルスクリプトの実行 (実行時間：約2分)

```
$ bash ./step4_GATK.sh 2>&1 | tee step4_GATK.log
```

作成された多型情報ファイルの確認

```
$ ls -lhtr
合計 561M
...
-rw-rw-r-- 1 guest01 guest01 15K 11月 11 14:29 variation.vcf.gz
-rw-rw-r-- 1 guest01 guest01 949 11月 11 14:29 variation.vcf.gz.tbi
-rw-rw-r-- 1 guest01 guest01 16K 11月 11 14:29 variation.filter.vcf.gz
-rw-rw-r-- 1 guest01 guest01 963 11月 11 14:29 variation.filter.vcf.gz.tbi
-rw-rw-r-- 1 guest01 guest01 9.9K 11月 11 14:29 step4_GATK.log
```

基本的にはゲノムリシーケンスデータによる変異検出と同じで、多型情報はVCFファイル（gz圧縮）として出力されている。

# 検出された多型情報 (variation.filter.vcf.gz) の確認



VCF形式で記載された多型情報

```
> less variation.filter.vcf.gz
```

...

```
1 chr02 30473056 . T A 440.64 PASS
AC=1;AF=0.500;AN=2;BaseQRankSum=-0.312;DP=233;E
xcessHet=0.000;FS=16.873;MLEAC=1;MLEAF=0.500;MQ=60.00;MQRankSum=0.000;QD=2.10;ReadPosRankS
um=-6.091;SOR=2.491
GT:AD:DP:GQ:PL 0/1:173,37:210:99:448,0,6758
2 chr02 30473059 . T A 74.64 QD
AC=1;AF=0.500;AN=2;BaseQRankSum=1.180;DP=200;ExcessHet=0.000;FS=11.139;MLEAC=1;MLEAF=0.500
;MQ=60.00;MQRankSum=0.000;QD=0.38;ReadPosRankSum=-5.911;SOR=2.030
GT:AD:DP:GQ:PL 0/1:169,28:197:82:82,0,6748
3 chr02 30473609 . G A 1065.06 PASS
AC=2;AF=1.00;AN=2;DP=34;ExcessHet=0.000;FS=0.000;MLEAC=2;MLEAF=1.00;MQ=60.00;QD=34.36;SOR=
0.756 GT:AD:DP:GQ:PL 1/1:0,31:31:93:1079,93,0
4 chr02 30474736 . T C 7210.06 PASS
AC=2;AF=1.00;AN=2;DP=252;ExcessHet=0.000;FS=0.000;MLEAC=2;MLEAF=1.00;MQ=60.00;QD=29.79;SOR
=0.815 GT:AD:DP:GQ:PL 1/1:0,242:242:99:7224,726,0
...
```

**PASS:**フィルタリングをパスした多型

# IGVでアラインメントと多型情報の可視化

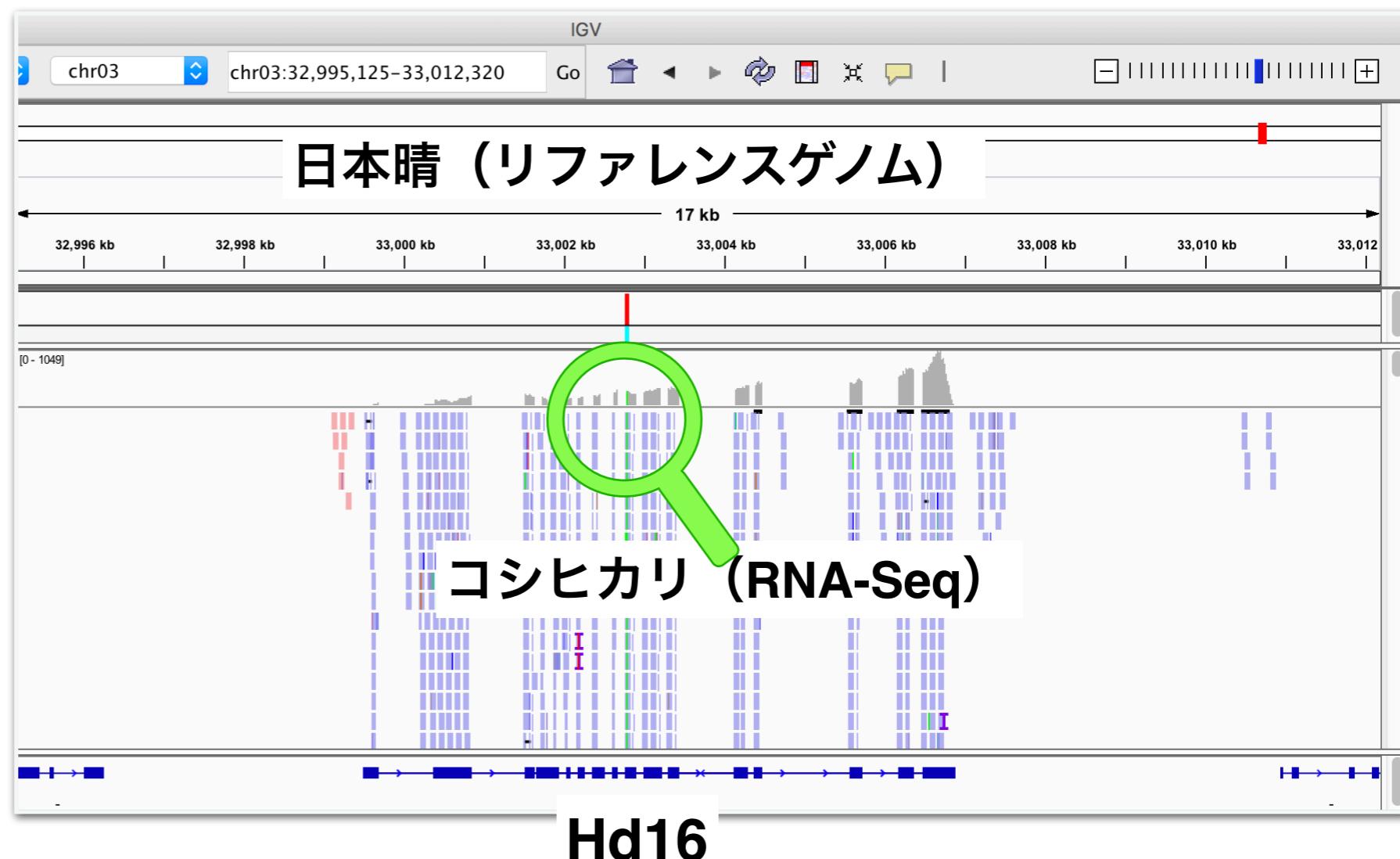
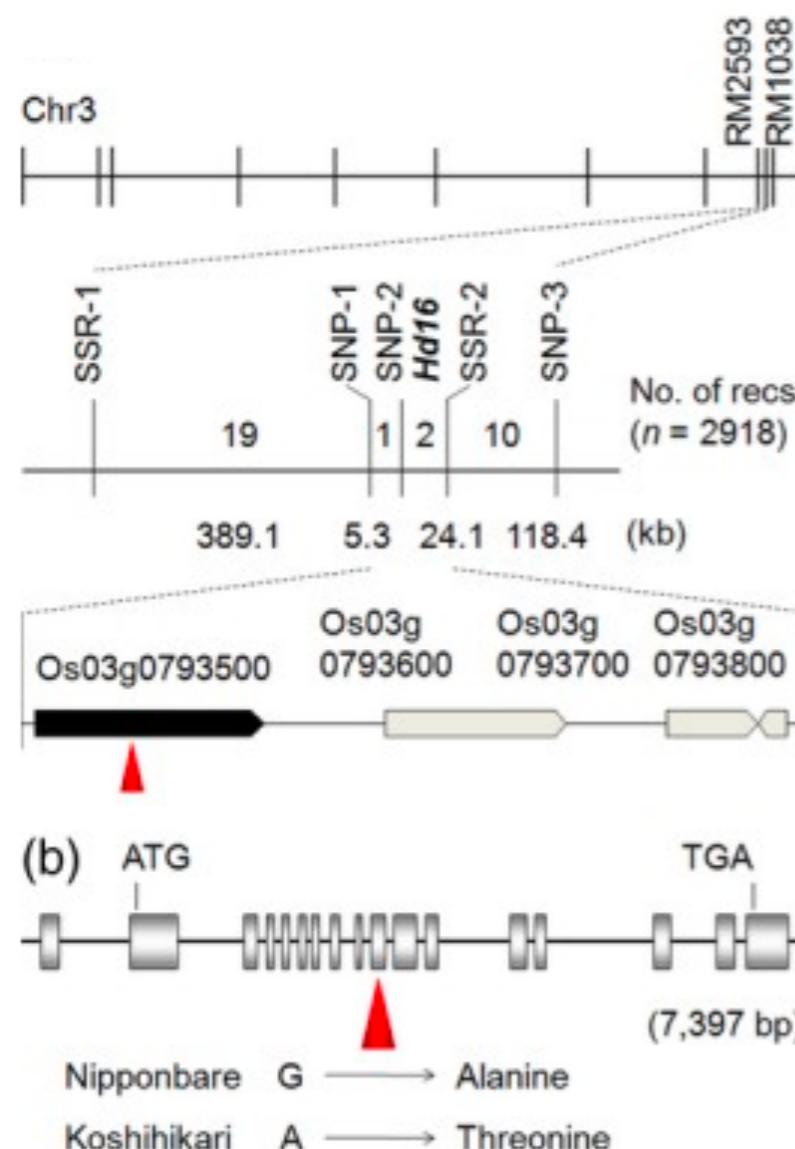
アラインメント情報 (Aligned.sortedByCoord.RG.MD.SplitN.out.bam) と多型情報 (variation.filter.vcf.gz) はIGVで閲覧することができる。



↓ : 前のスライドで示したVCFファイル中の4つの多型

# 既知の品種間多型は検出されているか？

Hd16遺伝子上にある、日本晴とコシヒカリの開花期の違いに関わる既知のFNP\*



from Hori K et al. Plant J. 2013

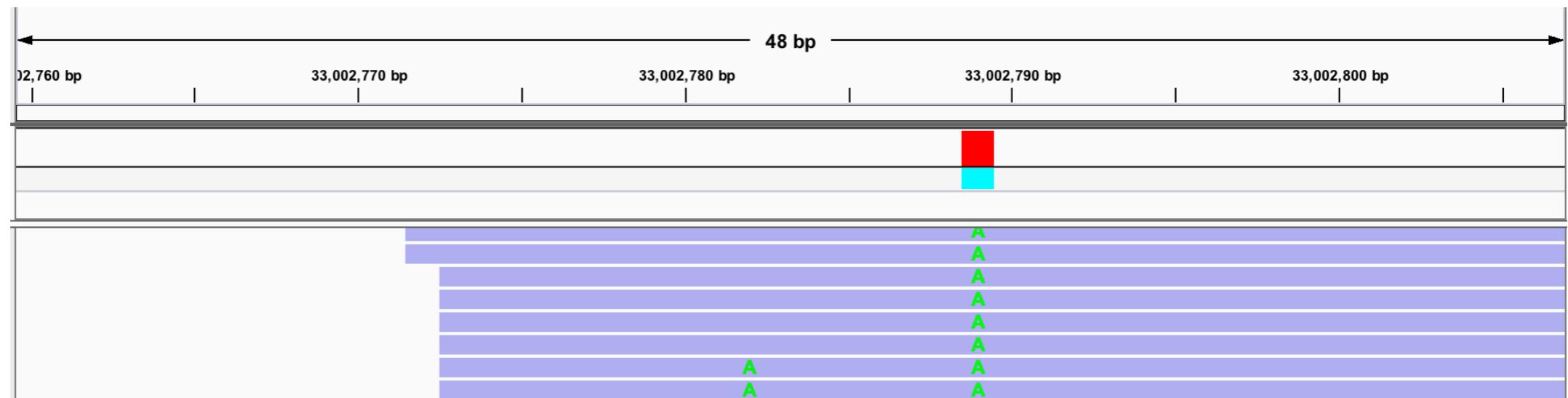
\*FNP=Functional Nucleotide Polymorphism

ローカルPC上(workshop/RNA-Seq/varcall)にゲノムワイドな解析結果から3番染色体の情報を抜き出した  
アラインメントとVCFファイルがあるのでIGVで開いて見てみよう。

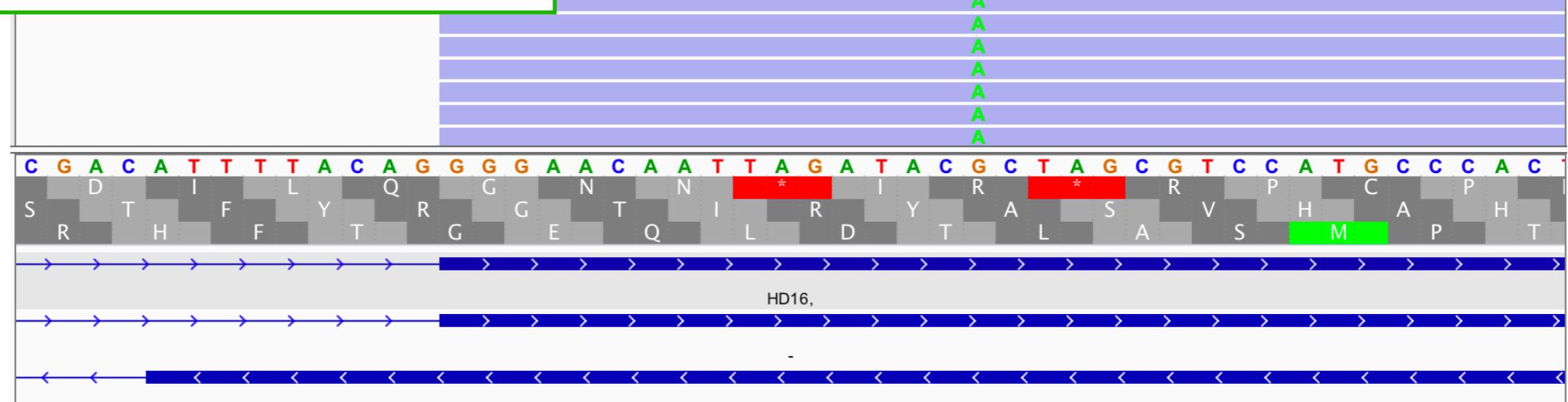
# アラインメントとVCFファイルの双方で確認

## VCFファイル中の多型情報

```
chr03    33002789      .      G      A      3117.77 PASS
AC=2;AF=1.00;AN=2;DP=92;ExcessHet=3.0103;FS=0.000;MLEAC=2;MLEAF=1.00;MQ=60.00;
QD=33.89;SOR=0.782          GT:AD:DP:GQ:PL  1/1:0,92:92:99:3146,276,0
```



## BAMファイル中のアラインメント 情報をIGVで可視化

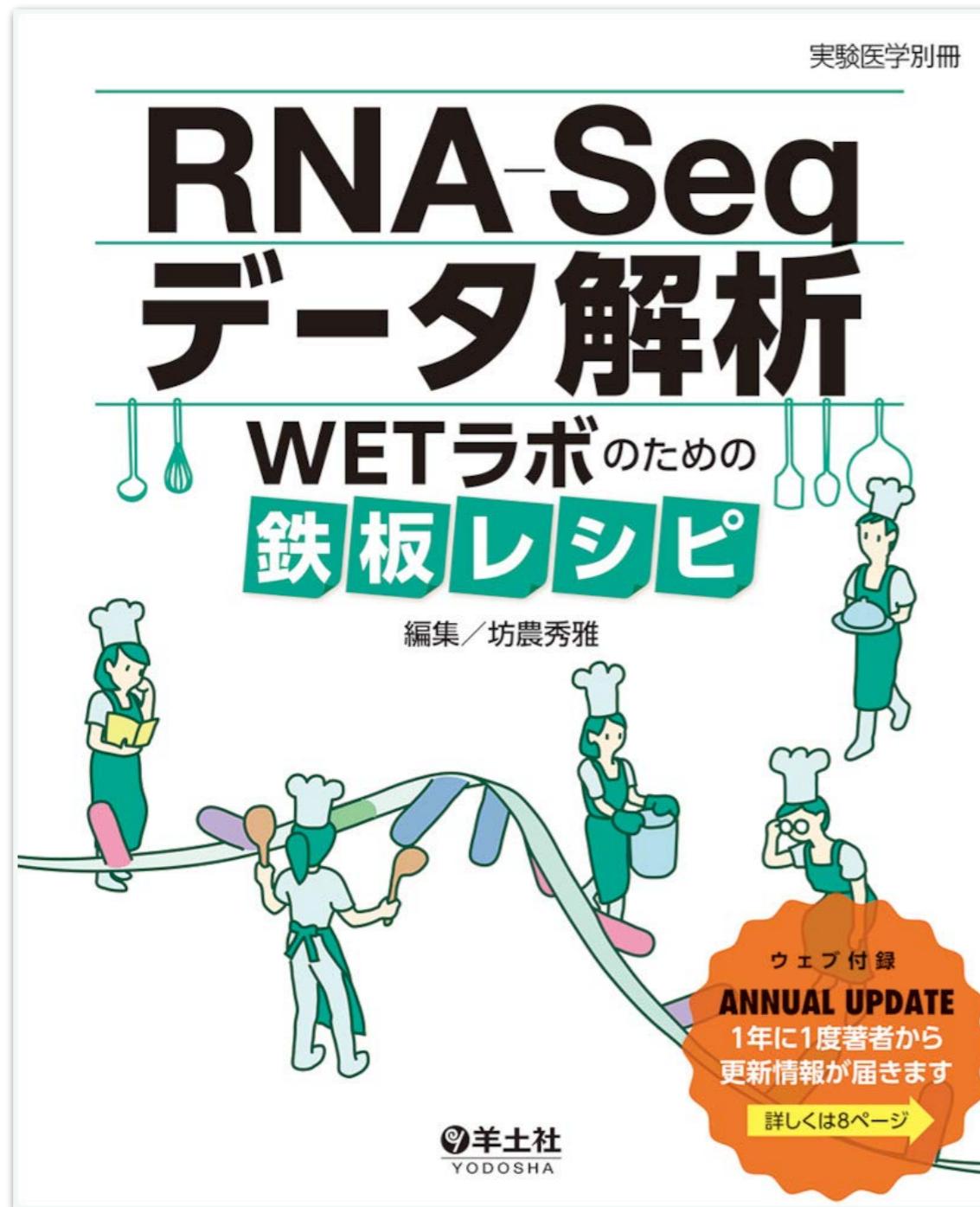


日本晴 (G) とコシヒカリ (A) の多型が確認された

# 参考図書・ウェブサイト

# 参考書籍

RNA-Seqデータ解析  
WETラボのための鉄板レシピ  
坊農 秀雅（編集）  
発売日：2019/12/2



次世代シークエンサーDRY解析教本  
清水厚志・坊農 秀雅（著・編集）  
発売日：2019/12/12



進展が早い分野なので、最新の情報はネットで入手するのが一番。

## まずはGoogleで検索！



Google 検索結果

Trimmomatic インストール

すべて 動画 画像 ニュース ショッピング もっと見る 設定 ツール 約 333 件 (0.31 秒)

Trimmomatic | FASTQ クリーニングツール - bioinformatics - biopapyrus  
<https://bi.biopapyrus.jp/rnaseq/qc/trimmomatic.html>

Trimmomatic はアダプターの除去のみならず、リードの末端から一定数の塩基をトリムしたりする、簡単なクオリティフィルタリングも行える。... インストール。Trimmomatic は Java によって書かれている。Trimmomatic 配布ウェブサイトからバイナリファイル (.jar ...)

Bash on Ubuntu on WindowsでSingle-cell RNA sequence解析 ③NGS ...  
<https://qiita.com/bioinformatics/items/2f3e0a2a2a2a2a2a2a2a>

2017/05/21 - 前回の続きから: trimmomaticのインストール、実行：シングルエンドの場合、シーケンスデータのクオリティチェックが終わったら、次に低品質なリードやアダプター配列と思しき配列を除去、トリミングしなくてはいけません。このような処理ツールの ...

NGS Surfer's Wiki | Trimmomaticインストールログ  
<https://cell-innovation.nig.ac.jp/wiki/tiki-index.php?...Trimmomaticインストールログ>

2016/09/23 - インストール手順は、以下のように \$ cd /work\_path \$ wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.36.zip \$ unzip Trimmomatic-0.36.zip \$ cp -r Trimmomatic-0.36 /tool\_path/ ...

アダプター配列の除去 cutadapt - The Cat Way  
<http://catway.jp/bioinformatics/qc/removeseq.html>

インストール: まず、http://hannanlab.cshl.edu/fastx\_toolkit/download.html から pre-compiled binary 版を落として、/usr/local/bin など ... アダプター配列のトリミングする Trimmomatic (Paired-end対応) 概要: paired-end に対応したアダプター配列をトリムする ...

0から始めるNGSデータ解析メモ : Trimmomatic  
<http://ngsdamemorandum.blogspot.com/2014/07/trimmomatic.html>

2014/07/24 - だいぶ慣れてきたので、インストールの仕方よりもツールの使い方をメインに記載するかな。アダプタートリマーツールであり、クリーニングも出来る Trimmomatic (<http://www.usadellab.org/cms/?page=trimmomatic>) 使い方としては、 ...

Trimmomatic - 井上 潤  
[http://www.geocities.jp/ancientfishtree/Trimmomatic\\_ji.html](http://www.geocities.jp/ancientfishtree/Trimmomatic_ji.html)

2013/11/25 - Q20といったオプションにより、read の両端にあるクオリティーの低い配列を取り除いてくれます。さらに、この処理によって短くなった read を取り除くことも可能です。アダプター配列も取り除いてくれるようです。インストールすると、メーカごとの ...

deep sequenceされたウィルスのアセンブルツール sparNA - macで ...  
<http://kazumaxneo.hatenablog.com/?page=1512063534>

2017/11/30 - Trimmomaticでのクオリティトリミング->ホストゲノムへのアライメント->アライメントされなかったリードの抽出、の順番で解析される。公式サイト、<https://bitbucket.org/biobakery/kneadata/wiki/Home>。インストール、依存。Trimmomatic ...

クオリティトリミングツール sickle - macでインフォマティクス  
<http://kazumaxneo.hatenablog.com/entry/2017/08/24/002643>

2017/08/24 - Trimmomaticと同様、ペアリードの順番が破壊されないよう、ペアの数を同じに崩えて出力できる (orphanリードは別出力)。インストール GitHub GitHub - najoshi/sickle: Windowed Adaptive Trimming for fastq files using quality brewで ...

- 最近では日本語でもかなり情報が充実してきている。
- エラーが出て解析ツールがうまく動かない場合は、エラーメッセージをコピペして検索すると、同じ状況に陥った人が解決方法を示してくれていることが多い。

# 参考ウェブサイト



進展が早い分野なので、最新の情報はネットで入手するのが一番。

親切な人がツールのインストールや解析方法などを記事にしてくれている

- **RNA-Seq Blog** <http://www.rna-seqblog.com>
- **Qiita** <https://qiita.com>
- **Hatena Blog** <http://hatenablog.com>

同じような問題で困っている人がいたり、解決方法が報告されているかも

- **SEQanswers** <http://seqanswers.com>
- **stackoverflow** <https://stackoverflow.com>

初心者向けにツールやデータベースの使い方を動画で紹介してくれている

- **TogoTV** <http://togotv.dbcls.jp>