

RNA-Seq法による トランスクリプトーム解析の基礎

農研機構 次世代作物開発研究センター 情報解析ユニット

(兼任：高度解析センター 大規模配列解析チーム)

川原 善浩

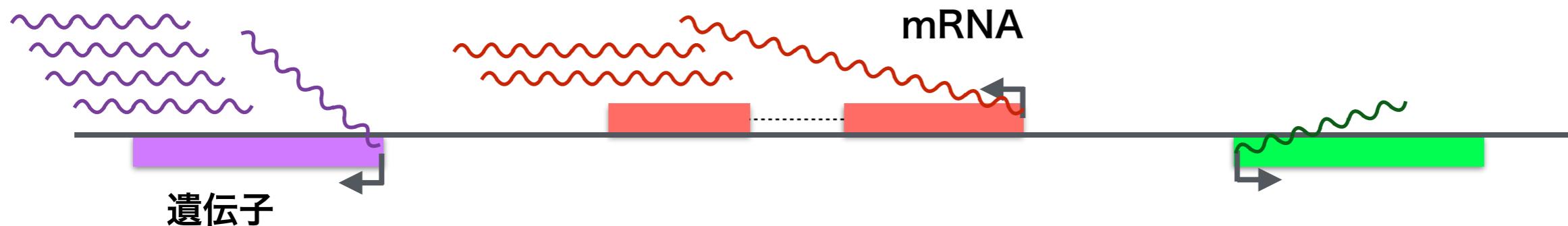


y.kawahara@affrc.go.jp

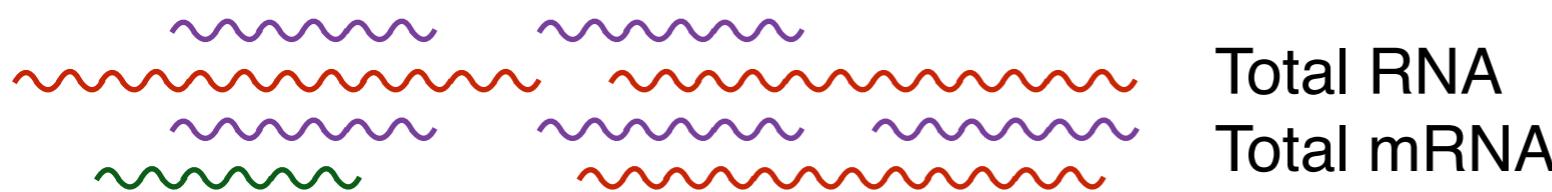


@YoshiKawahara

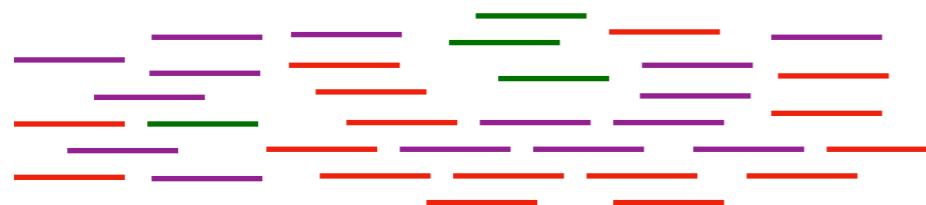
RNA-Seqによるトランскriプトーム解析



サンプリング、RNA抽出



断片化、ライブラリ調製、シーケンシング



データ解析

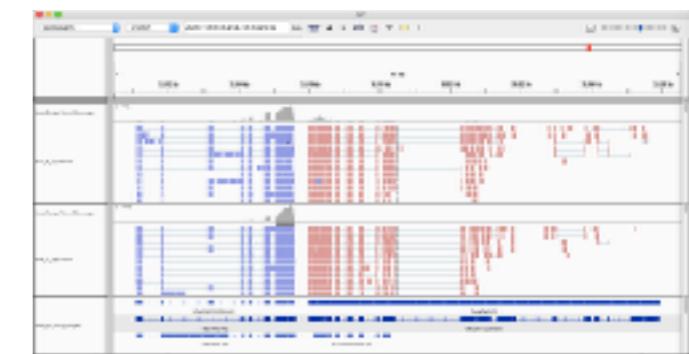
発現している遺伝子構造や発現量の定量



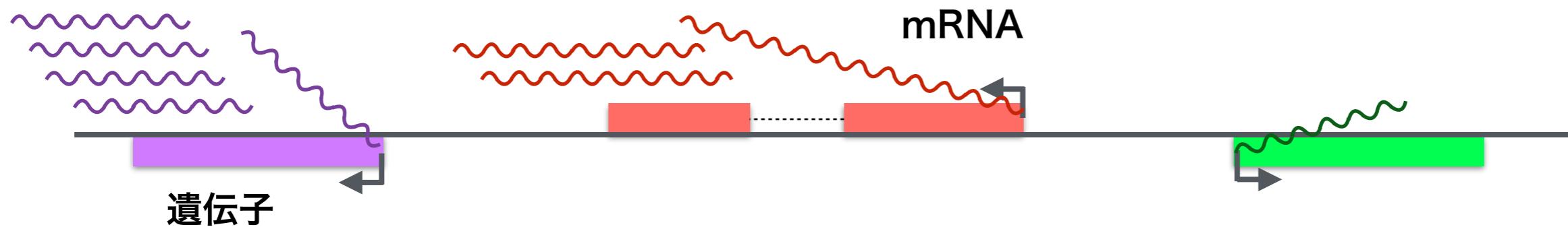
Licensed under CC-BY 4.0
©Togo picture gallery by DBCLS

RNA-Seqリード配列

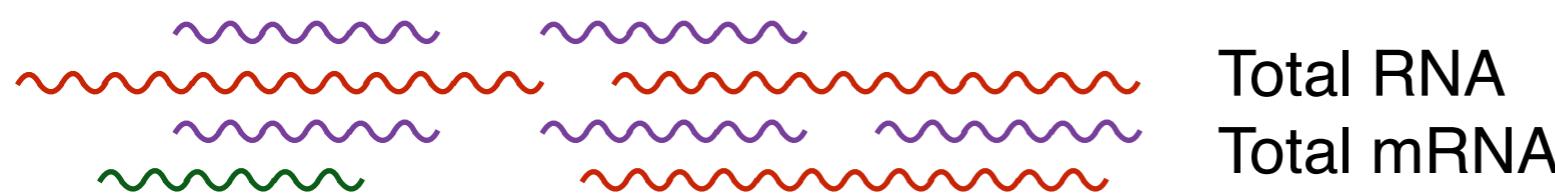
```
TATCGATGCATGCGCATCGTATCGA
TGCGCATCGTATCGATGCGCATCGT
AGGGGGGTCGATGCGCATCGTATCG
```



ロングリードタイプのシーケンサーによるRNA-Seq解析



サンプリング、RNA抽出



ライブラリ調製、シーケンシング



full-length cDNAやRNAを丸々シーケンシング

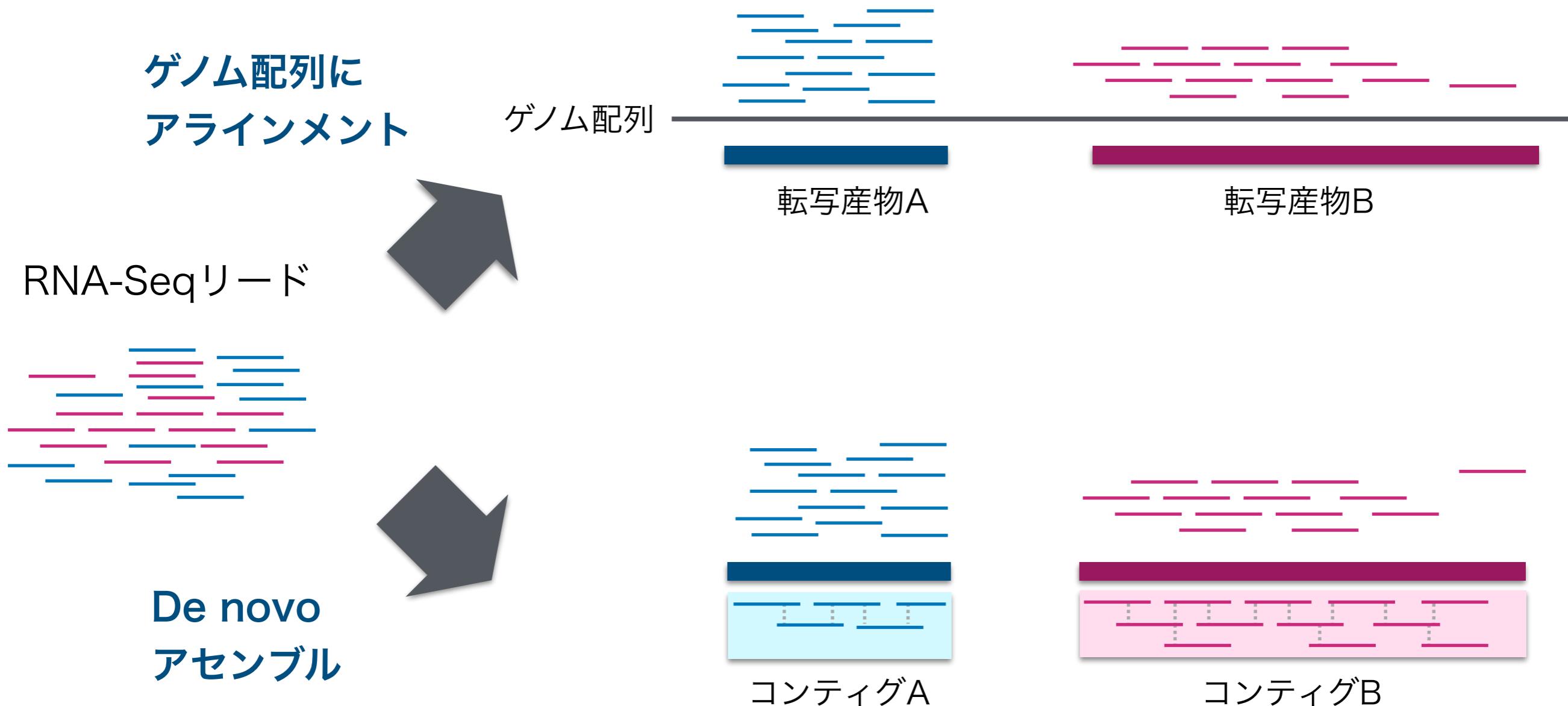


Licensed under CC-BY 4.0
©Togo picture gallery by DBCLS

ロングリードタイプのシーケンサー

データ解析手法は大きく2つに分けられる

RNA-Seqリードをゲノム配列にアラインメントし、
転写産物ごとにマップされたリードをカウント（発現量の定量）



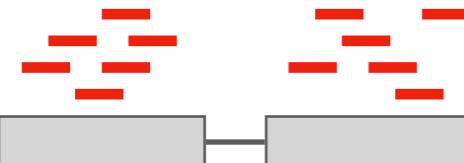
RNA-Seqリードをもとに転写産物配列をde novoアセンブル。そして、アセンブルされた転写産物配列にRNA-Seqリードをアライメントし、転写産物ごとにリード数をカウント（発現量の定量）

ライブラリ調製やシーケンシング方法の選択

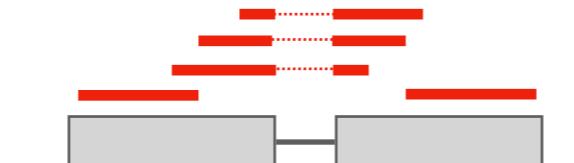
リード長

リードが長いほど得られる遺伝子構造 (Exon-Intron) 情報は多くなり
アセンブルなどには有利だが、コストが上がる。

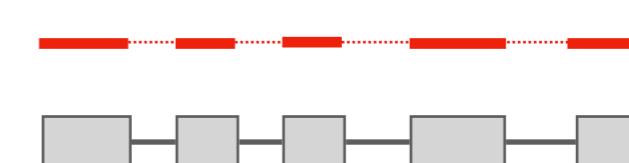
50 bp



150 bp



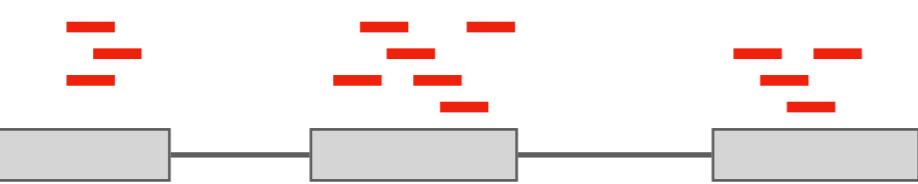
Long read



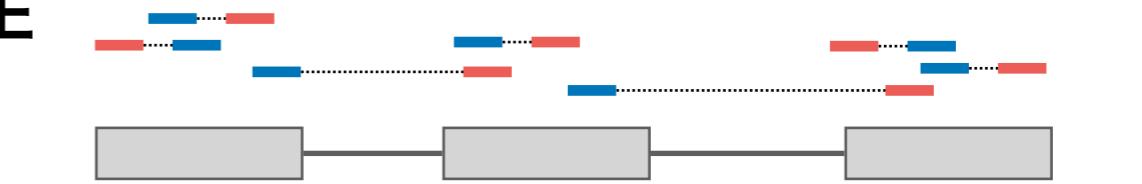
single or paired-end read

断片化したcDNAの片側だけを読むか両端を読むかの違い
ペア間の距離情報を用いることでアラインメントや
アセンブル精度の向上が期待される

SR

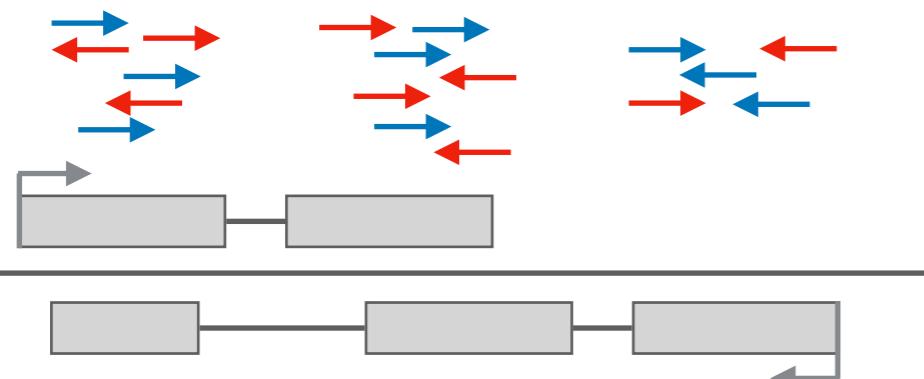


PE

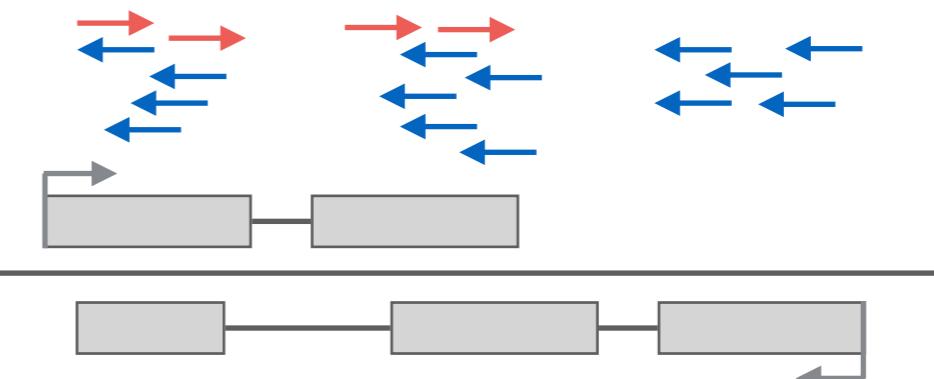


standard or stranded (directional)

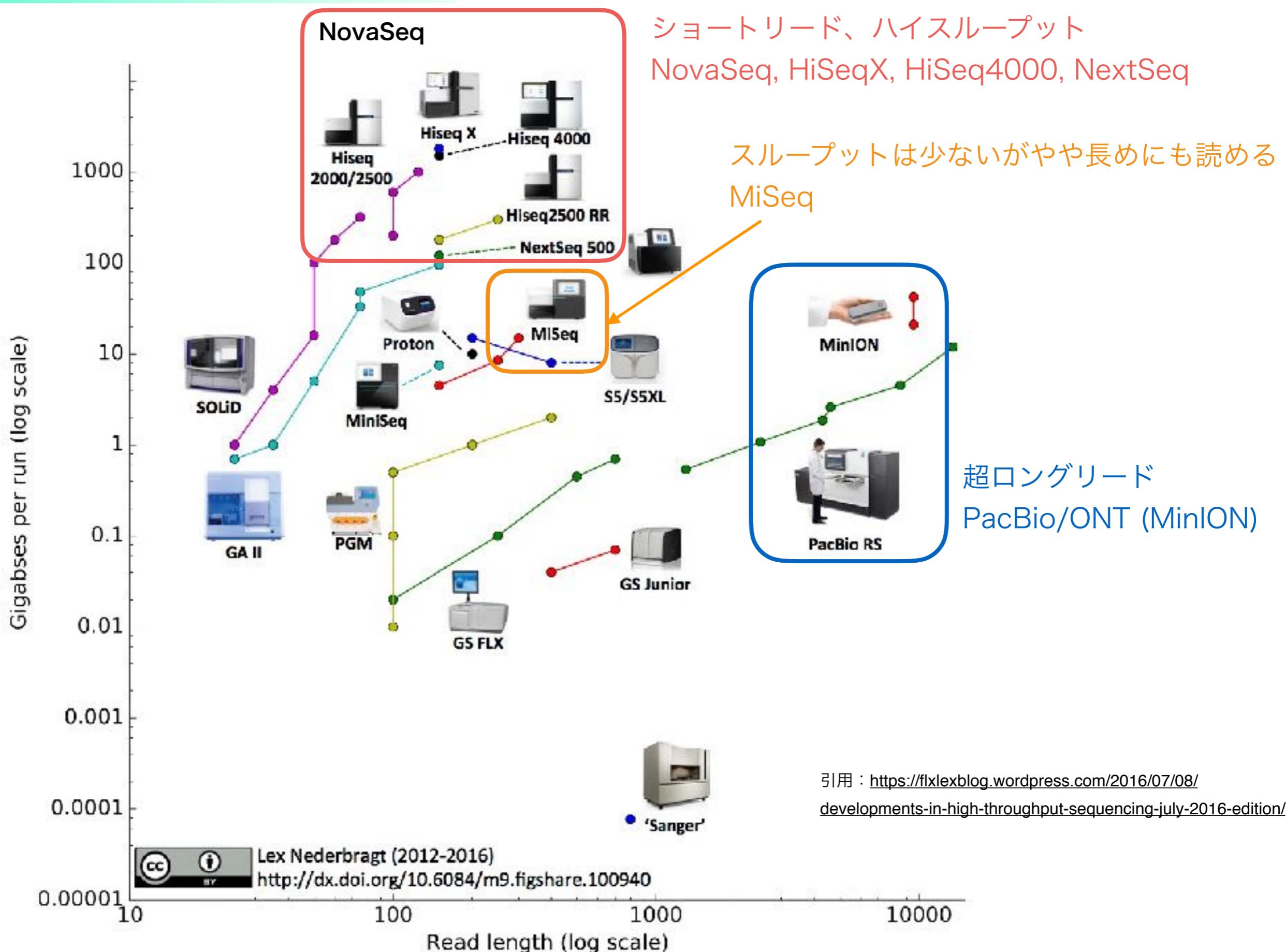
転写の向きとリードの向きは無関係



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。



シーケンサーの選択



目的にあったライブラリ調製法やシークエンシング手法を

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

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。

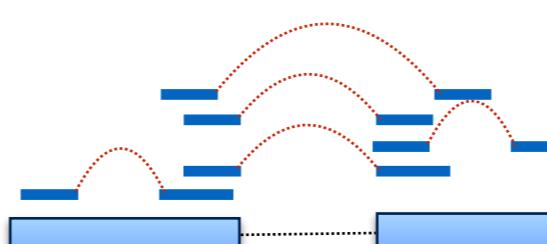
HiSeq, single-read, 50bp



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。

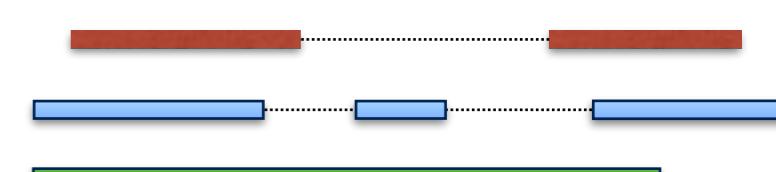
HiSeq, paired-end, 100bp



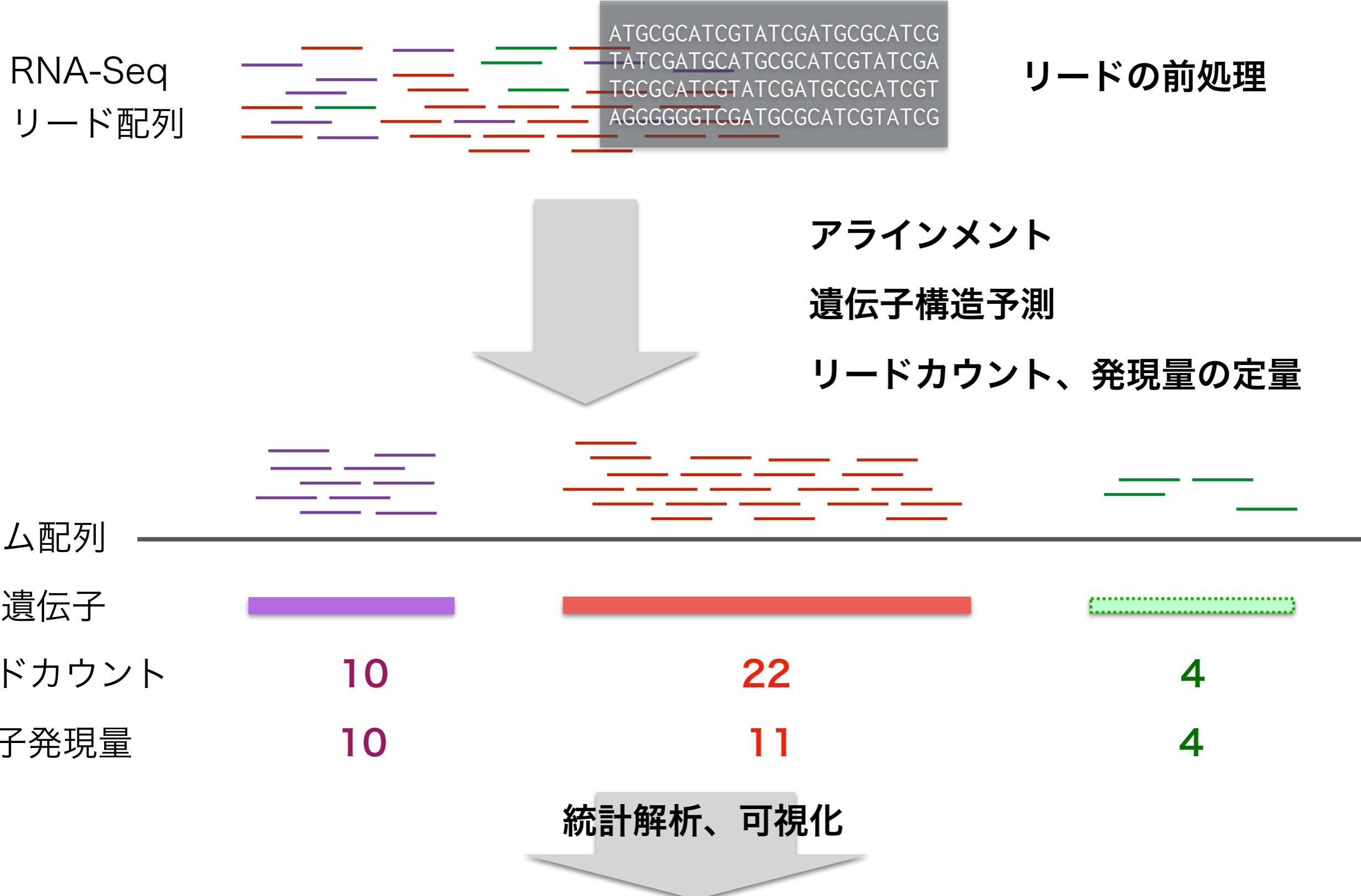
発現量よりもまずはどん
な遺伝子が発現している
か知りたい。

リード数は少ないが、
とにかく長いリードで
転写産物の構造を得る。

MiSeq, PacBio, ONT



本演習で行なうのはリファレンス情報を用いた遺伝子発現解析



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

本演習で取り組む課題

1. リファレンス情報を用いた遺伝子発現解析（Linuxサーバ）
2. IGVやRなどを用いた発現比較解析と可視化（ローカルPC）

本演習用で用いるデータと解析目的

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

VS

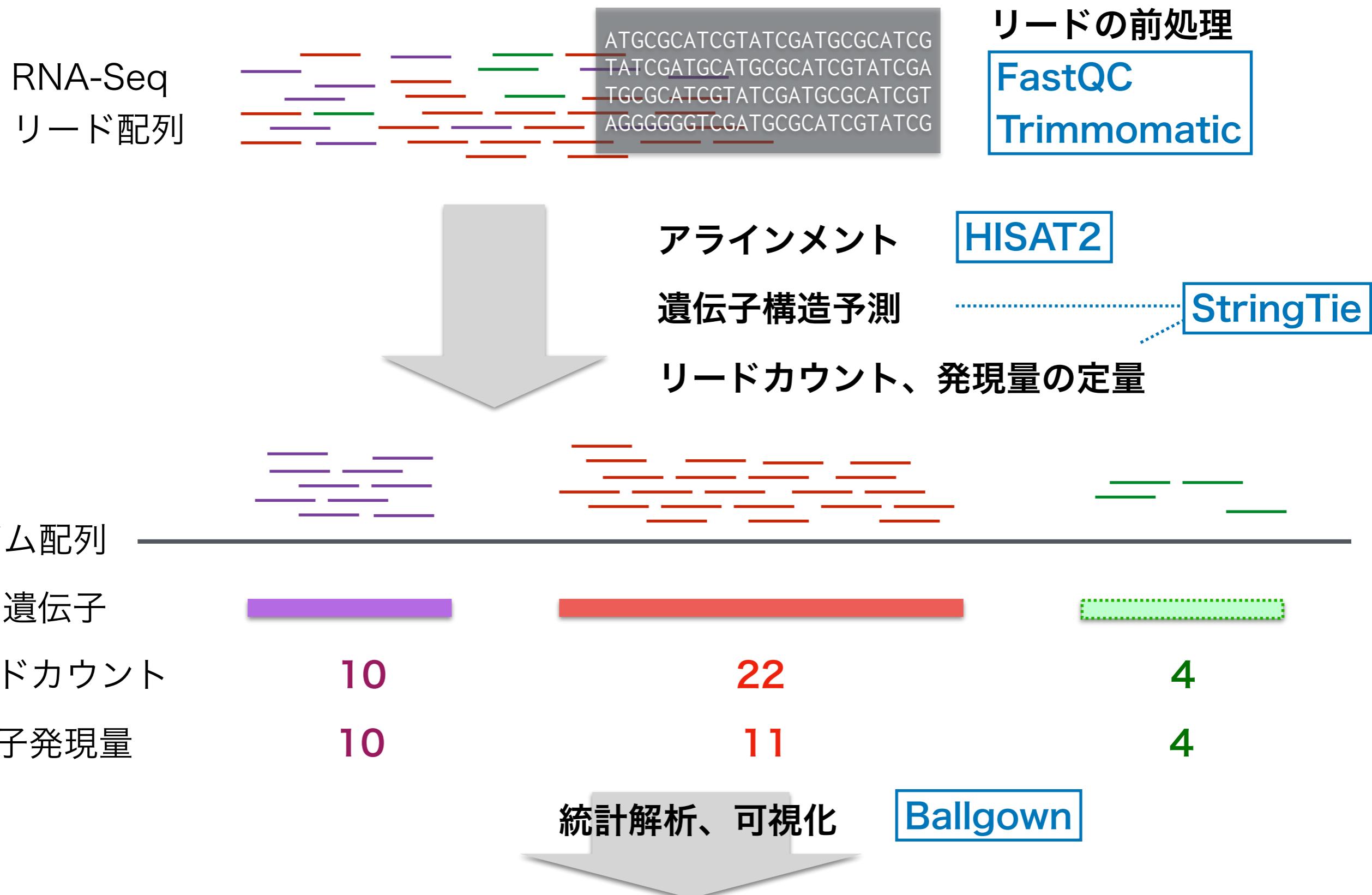


00:00 AM

- ・田んぼで栽培する、ある品種のイネの葉身のサンプル。
- ・4つの異なる生育ステージ（4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によって、**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。

本演習で行なうのはリファレンス情報を用いた遺伝子発現解析



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

演習用データの準備

演習用データ（/work/NGSworkshop2017/RNA-Seq_analysis.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。

```
$ 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
/home2/guest01
$ cp /work/NGSworkshop2017/RNA-Seq_analysis.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
total 438M
-rw-r--r-- 1 guest01 guest01 438M Oct  9 10:33 RNA-Seq_analysis.tar.gz
$ tar xfz RNA-Seq_analysis.tar.gz   ← 解析用データを解凍、展開
$ ls -lh
total 438M
drwxr-xr-x 2 guest01 guest01 4.0K Oct  9 10:05 RNA-Seq_analysis
-rw-r--r-- 1 guest01 guest01 438M Oct  9 10:33 RNA-Seq_analysis.tar.gz
$ cd RNA-Seq_analysis
$ ls
annotation.gtf          rice_D_rep4_r1.org.fastq.gz  ss.tab
assemblies.txt           rice_D_rep4_r2.org.fastq.gz  step1_install_tools.sh
exon.tab                 rice_N_rep1_r1.org.fastq.gz  step2_preprocessing.sh
genome.fa                rice_N_rep1_r2.org.fastq.gz  step3_HISAT2-build.sh
rice_D_rep1_r1.org.fastq.gz rice_N_rep2_r1.org.fastq.gz  step4_HISAT2.sh
rice_D_rep1_r2.org.fastq.gz rice_N_rep2_r2.org.fastq.gz  step5_StringTie-assemble.sh
rice_D_rep2_r1.org.fastq.gz rice_N_rep3_r1.org.fastq.gz  step6_StringTie-merge.sh
rice_D_rep2_r2.org.fastq.gz rice_N_rep3_r2.org.fastq.gz  step7_gffcompare.sh
rice_D_rep3_r1.org.fastq.gz rice_N_rep4_r1.org.fastq.gz  step8_StringTie-abundance_estimation.sh
rice_D_rep3_r2.org.fastq.gz rice_N_rep4_r2.org.fastq.gz
```

演習用のデータには時間節約のため事前に準備した解析結果も含まれているので注意。

RNA-Seq_analysis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)
- ・ アノテーションファイル (annotation.gtf, *.tab)
- ・ mRNA-Seqリード配列 (*.org.fastq.gz)
- ・ シェルスクリプト (step*.sh)

RNA-Seq解析に用いるツール群

HISAT2

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



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 to 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 uses a large set of small GFM indexes that collectively cover the whole genome (each index representing a genomic region of 56 Kbp, with 55,000 indexes needed to cover the human population). These small indexes (called local indexes), combined with several alignment strategies, enable rapid and accurate alignment of sequencing reads. This new indexing scheme is called a Hierarchical Graph FM Index (HGFM).

HISAT2 2.1.0 release 6/8/2017

[Site Map](#)

HISAT2

splice-awareなリードアラインメント

[list](#)

Implemented --new-summary option to output a new style of alignment summary.

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



StringTie

Transcript assembly and quantification for RNA-Seq



[Home](#) [Manual](#) [FAQ](#)

[CCB > CCB > StringTie](#)

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

StringTie

遺伝子構造のアセンブル・リードカウント

multiple transcript assemblies, fully expressed or other

Overview

StringTie is algorithm for splice variant detection that also identifies genes between programs.

News

2/15/2017 - v1.3.3 release

- fixed the computation of the TPM value that was reported incorrectly sometimes in the presence of guides (-G option)
- few other minor fixes

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

Tuxedo suite tools

BowtieやTopHat, Cufflinksの後継プログラム群



OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

[Home](#)

[Install](#)

[Help](#)

[Search](#)

[Developers](#)

[About](#)

[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: 2,117 test coverage: 35%

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

Flexible, isoform-level differential expression analysis

Bioconductor version: Release (3.5)

Tools for statistical analysis and visualization of transcriptomes

Author: Jack Fu [aut], Allyn Jeffrey T. Lock [aut, ths]

Maintainer: Jack Fu <jmfu@jhu.edu>

Citation (from within R, enter:

R> J. Fu, A. G. Fraze, A. G. Collado-Vides, Differential expression analysis. R package Version 2.0.4.

Ballgown

発現解析と可視化

Documentation

Bioconductor

- Package vignettes and manuals.
- Workflows for learning and use.
- Course and conference material.
- Videos.
- Community resources and tutorials.

[R / CRAN packages and documentation](#)

Support

Please read the [posting_guide](#). Post questions about Bioconductor to one of the following locations:

- [Support site](#) - for questions about Bioconductor packages.
- [Bio-devel mailing list](#) - for package developers

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

Step 1. 解析ツールの取得

各ツールのウェブサイトから最新版のツールのバイナリ、もしくはソースファイルをダウンロードし、インストールする。

ツールによっては、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」を用いても簡単にインストールすることができる。

本演習でLinuxサーバにインストールする解析ツール

1. FastQC
2. Trimmomatic
3. HISAT2
4. StringTie
5. gffcompare
6. SAMtools

*ballgownはR/Bioconductorのパッケージのため、のちほどローカルPCのR上にインストールする。

Step 1. 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 step1_install_tools.sh
```

- 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。

- step1_install_tools.shの前半部分 -

```
### Step.1: download and install tools ----- #で始まるのはコメント行
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
# install FastQC
wget http://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0.11.5.zip
unzip fastqc_v0.11.5.zip ----- 解凍・展開 FastQCのダウンロード
chmod +x FastQC/fastqc ----- fastqcプログラムに実行権限を与える
# install Trimmomatic
wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.36.zip
unzip Trimmomatic-0.36.zip ----- 解凍・展開 Trimmomaticのダウンロード
# install HISAT2
wget ftp://ftp.ccb.jhu.edu/pub/infphilo/hisat2/downloads/hisat2-2.1.0-Linux_x86_64.zip
unzip hisat2-2.1.0-Linux_x86_64.zip ----- 解凍・展開 HISAT2のダウンロード
```

- 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。

Step 1. 解析ツールのダウンロードとインストール

- step1_install_tools.shの後半部分 -

```
# install StringTie
wget http://ccb.jhu.edu/software/stringtie/d1/stringtie-1.3.3b.Linux_x86_64.tar.gz
tar xfz stringtie-1.3.3b.Linux_x86_64.tar.gz ----- 解凍・展開 ----- StringTieのダウンロード
# install gffcompare
wget http://ccb.jhu.edu/software/stringtie/d1/gffcompare-0.10.1.Linux_x86_64.tar.gz
tar xfz gffcompare-0.10.1.Linux_x86_64.tar.gz ----- 解凍・展開 ----- gffcompareのダウンロード
# install Samtools
wget https://github.com/samtools/samtools/releases/download/1.6/samtools-1.6.tar.bz2
tar xfj samtools-1.6.tar.bz2 ----- 解凍・展開 ----- Samtoolsのダウンロード
cd samtools-1.6 ----- 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
```

- ・ *.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」）。
- ・ 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。

Step 1. 解析ツールのダウンロードとインストール

シェルスクリプトの実行

- 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エラー出力をターミナル画面に表示する。

```
$ bash ./step1_install_tools.sh
```

- リダイレクト機能を使うと、スクリプトの結果はターミナル画面に表示せず、標準出力と標準エラー出力をそれぞれ別ファイルに書き出す。あとでログを見直すことができて便利です。

```
$ bash ./step1_install_tools.sh 2> step1_install_tools.stderr > \
step1_install_tools.stdout
```

(実行時間：約2分)

このバックスラッシュはここまで改行（実行）するのではなく、コマンドが続いていることを表す。

- 一連のコマンドが書かれたシェルスクリプトにより、コマンドを連續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。

解析ツールの取得 (FastQC)

<http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>



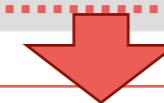
Babraham Bioinformatics

About | People | Services | Projects | Training | Publications

FastQC

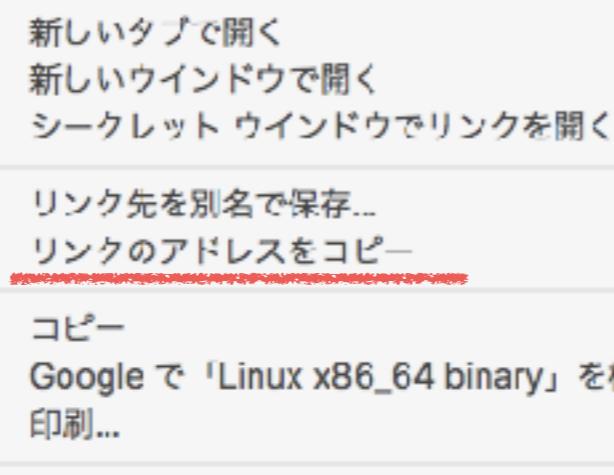
Function	A quality control tool for high throughput sequence data.
Language	Java
Requirements	A suitable Java Runtime Environment The Picard BAM/SAM Libraries (included in download)
Code Maturity	Stable. Mature code, but feedback is appreciated.
Code Released	Yes, under GPL v3 or later .
Initial Contact	Simon Andrews

Download Now



FastQC A quality control application for high throughput sequence data

- [README](#)
- [Installation and setup instructions](#)
- [Release Notes](#) Please read these before using the program.
- [FastQC v0.11.5 \(Win/Linux zip file\)](#)
- [FastQC v0.11.5 \(Mac DMG image\)](#)
- [Source Code for FastQC v0.11.5 \(zip\)](#)



サーバ上にバイナリ／ソースファイルを直接ダウンロードしたい場合は、以下のようにwgetコマンドを使う。

\$ wget [URL]

ウェブサイトのリンク部分で右クリックし、「リンクのアドレスをコピー」を選択すれば、URLが簡単に取得できる。

\$ wget http://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0.11.5.zip

*MacのChromeの場合

解析ツールの取得 (Trimmomatic)

<http://www.usadellab.org/cms/?page=trimmomatic>

USADELLAB.org

Home Research Education Service & Software Publications
Supporting Info About Us NGS, DE and other things

Trimmomatic: A flexible read trimming tool for Illumina NGS data

Citations

Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. *Bioinformatics*, btu170.

Downloading Trimmomatic

Version 0.36: [binary](#), [source](#) and [manual](#)

Downloading Trimmomatic

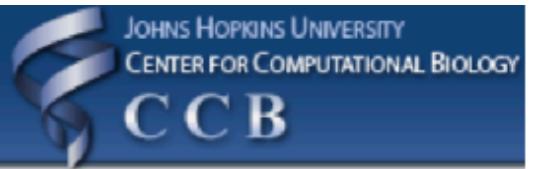
Version 0.36: [binary](#), [source](#) and [manual](#)

解析ツールの取得 (HISAT2)

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

HISAT2

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



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 to 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 uses a large set of small GFM indexes that collectively cover the whole genome (each index representing a genomic region of 56 Kbp, with 55,000 indexes needed to cover the human population). These small indexes (called local indexes), combined with several alignment strategies, enable rapid and accurate alignment of sequencing reads. This new indexing scheme is called a Hierarchical Graph FM index (HGFM).



HISAT2 2.1.0 release 6/8/2017

- This major version includes the first release of HISAT-genotype, which currently performs HLA typing, DNA fingerprinting analysis, and CYP typing on whole genome sequencing (WGS) reads. We plan to extend the system so that it can refer to the HISAT-genotype website.
- HISAT2 can be directly compiled by Dyer.
- Implemented --new-summary parse for programming purposes.
- Implemented --summary-file option (e.g. stderr).
- Fixed discrepancy in HISAT2's RNA-seq reads.

Releases

version 2.1.0	6/8/2017
Source code	
Linux x86_64 binary	
Mac OS X x86_64 binary	
Windows binary	

Site Map

- [Home](#)
- [Manual](#)
- [FAQ](#)

News and Updates

New releases and related tools will be announced through the Bowtie [mailing list](#).

Getting Help

Please use hisat2.genomics@gmail.com for private communications only. Please do not email technical questions to HISAT2 contributors directly.

Releases

version 2.1.0	6/8/2017
Source code	
Linux x86_64 binary	
Mac OS X x86_64 binary	
Windows binary	

Please cite:

Kim D, Langmead B and Salzberg SL.

HISAT2 2.0.5 release 11/4/2016

Version 2.0.5 is a minor release with the following changes.

- Due to a policy change (HTTP to HTTPS) in using SRA data (`--sra-option`), users are strongly encouraged to use this version. As of 11/9/2016, NCBI will begin a permanent redirect to HTTPS, which means the previous versions of HISAT2 no longer works with `--sra-acc` option soon.
- Implemented -I and -X options for specifying minimum and maximum fragment lengths. The options are valid only when used with --no-spliced-alignment, which is used for the alignment of DNA-seq reads.

解析ツールの取得 (StringTie)

<http://wwwccb.jhu.edu/software/stringtie/>

StringTie
Transcript assembly and quantification for RNA-Seq

JOHNS HOPKINS UNIVERSITY
CENTER FOR COMPUTATIONAL BIOLOGY
CCB

Home Manual FAQ CCB » CBCC » 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 splice variants for each gene to not only align longer sequences between experiments, but also alignments longer sequences between experiments, S programs (DESeq2, edgeR, etc).

News

- » 2/15/2017 - v1.3.3 release
 - fixed the computation of
 - few other minor fixes

Obtaining and installing StringTie

The current version of StringTie can be downloaded as precompiled binary or as a source package:

- [stringtie-1.3.3b.tar.gz](#) : source package
- [stringtie-1.3.3b.Linux_x86_64.tar.gz](#) : Linux x86_64 binary package
- [stringtie-1.3.3b.OSX_x86_64.tar.gz](#) : Apple OS X binary package (for OS X v10.7 and above)

Obtaining and installing StringTie

The current version of StringTie can be downloaded as precompiled binary or as a source package:

- [stringtie-1.3.3b.tar.gz](#) : source package
- [stringtie-1.3.3b.Linux_x86_64.tar.gz](#) : Linux x86_64 binary package
- [stringtie-1.3.3b.OSX_x86_64.tar.gz](#) : Apple OS X binary package (for OS X v10.7 and above)

解析ツールの取得 (gffcompare)

<http://ccb.jhu.edu/software/stringtie/gff.shtml>

GFF utilities
Programs for processing GTF/GFF files

JOHNS HOPKINS UNIVERSITY
CENTER FOR COMPUTATIONAL BIOLOGY
CCB

CCB » CBCC » GFF utilities

- [GFF files](#)
- [The gffread utility](#)
 - [Extracting transcript sequences](#)
 - [Obtaining gffread](#)
- [The gffcompare utility](#)
 - [Example: evaluating transcript discovery accuracy](#)
 - [Obtaining gffcompare](#)

GFF files

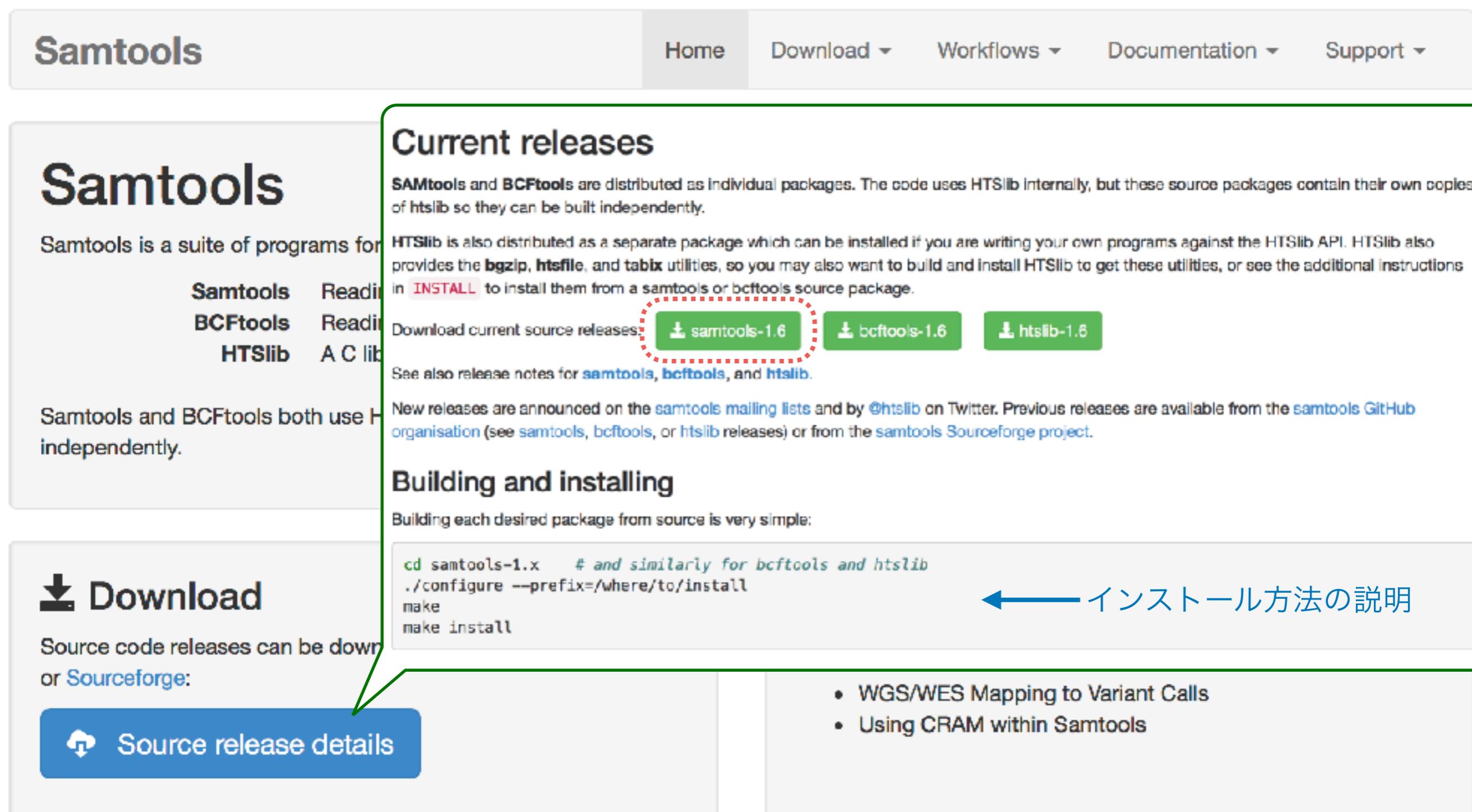
Many bioinformatics programs represent genes and transcripts in GFF format (**General Feature Format**) which simply describes the locations and the attributes of gene and transcript features on the genome (chromosome or scaffolds/contigs). GFF has many versions, but the two most popular that are **GTF2** (Gene Transfer Format, described [here](#)) and **GFF3** (defined [here](#)). Here are a few details about the way these formats are interpreted by StringTie and other bioinformatics programs.

Obtaining gffcompare

- project on Github: <https://github.com/gperteal/gffcompare>
- source package: [gffcompare-0.10.1.tar.gz](#)
- precompiled Linux x86_64 binary (statically built on CentOS 5) : [gffcompare-0.10.1.Linux_x86_64.tar.gz](#)
- precompiled Apple OS X binary (for OS X 10.7 and higher): [gffcompare-0.10.1.OSX_x86_64.tar.gz](#)

解析ツールの取得 (SAMtools)

<http://www.htslib.org/>



The screenshot shows the Samtools website with a green border around the 'Current releases' section. In this section, the 'samtools-1.6' download button is circled in red. A blue arrow points from the 'Source release details' button in the bottom left corner to this circled button. Another blue arrow points from the 'samtools-1.6' download button to the explanatory text on the right.

Samtools

Samtools is a suite of programs for reading, writing, and manipulating SAM/BAM/CRAM files.

Samtools	Reading and writing SAM/BAM/CRAM files.
BCFtools	Reading and writing BCF files.
HTSlib	A C library for reading and writing BAM/SAM/CRAM files.

Samtools and BCFtools both use HTSlib internally, but can be built independently.

Download

Source code releases can be downloaded from [Sourceforge](#):

Source release details

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:

- [!\[\]\(6c4743e17c4bf6045997b614f2b9cdb6_img.jpg\) samtools-1.6](#)
- [!\[\]\(854464efc125f8d92c8241542fc54ccf_img.jpg\) bcf-tools-1.6](#)
- [!\[\]\(6b489f60d58b9f51c48c12fd5ec11496_img.jpg\) htslib-1.6](#)

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](#).

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
```

← インストール方法の説明

- WGS/WES Mapping to Variant Calls
- Using CRAM within Samtools

SAMtoolsはソースファイルしか提供されていないため、makeコマンドを使ってコンパイル、インストールする必要がある。

インストールされたツールを確認

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 hisat2-2.1.0 stringtie-1.3.3b.Linux_x86_64
gffcompare-0.10.1.Linux_x86_64 samtools-1.6 Trimmomatic-0.36
```

例えば、HISAT2は「tool/hisat2-2.1.0」以下にインストールされている。

```
$ ls tool/hisat2-2.1.0
AUTHORS hisat2-build-s-debug hisatgenotype_hla_cyp.py
doc hisat2_extract_exons.py hisatgenotype_locus.py
example hisat2_extract_snps_haplotypes_UCSC.py hisatgenotype_modules
extract_exons.py hisat2_extract_snps_haplotypes_VCF.py hisatgenotype.py
extract_splice_sites.py hisat2_extract_splice_sites.py hisatgenotype_scripts
hisat2 hisat2-inspect LICENSE
hisat2-align-l hisat2-inspect-l MANUAL
hisat2-align-l-debug hisat2-inspect-l-debug MANUAL.markdown
hisat2-align-s hisat2-inspect-s NEWS
hisat2-align-s-debug hisat2-inspect-s-debug scripts
hisat2-build hisat2_simulate_reads.py TUTORIAL
hisat2-build-l hisatgenotype_build_genome.py VERSION
hisat2-build-l-debug hisatgenotype_extract_reads.py
hisat2-build-s hisatgenotype_extract_vars.py
```

*太字は本演習で用いるプログラム

インストールされたツールを確認

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/hisat2-2.1.0/hisat2 -h ..... -hオプションをつけてHISAT2を実行
```

HISAT2 version 2.1.0 by Daehwan Kim (infphilo@gmail.com, wwwccb.jhu.edu/people/infphilo)

Usage:

```
hisat2 [options]* -x <ht2-idx> {-1 <m1> -2 <m2> | -U <r> | --sra-acc <SRA accession number>} [-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).

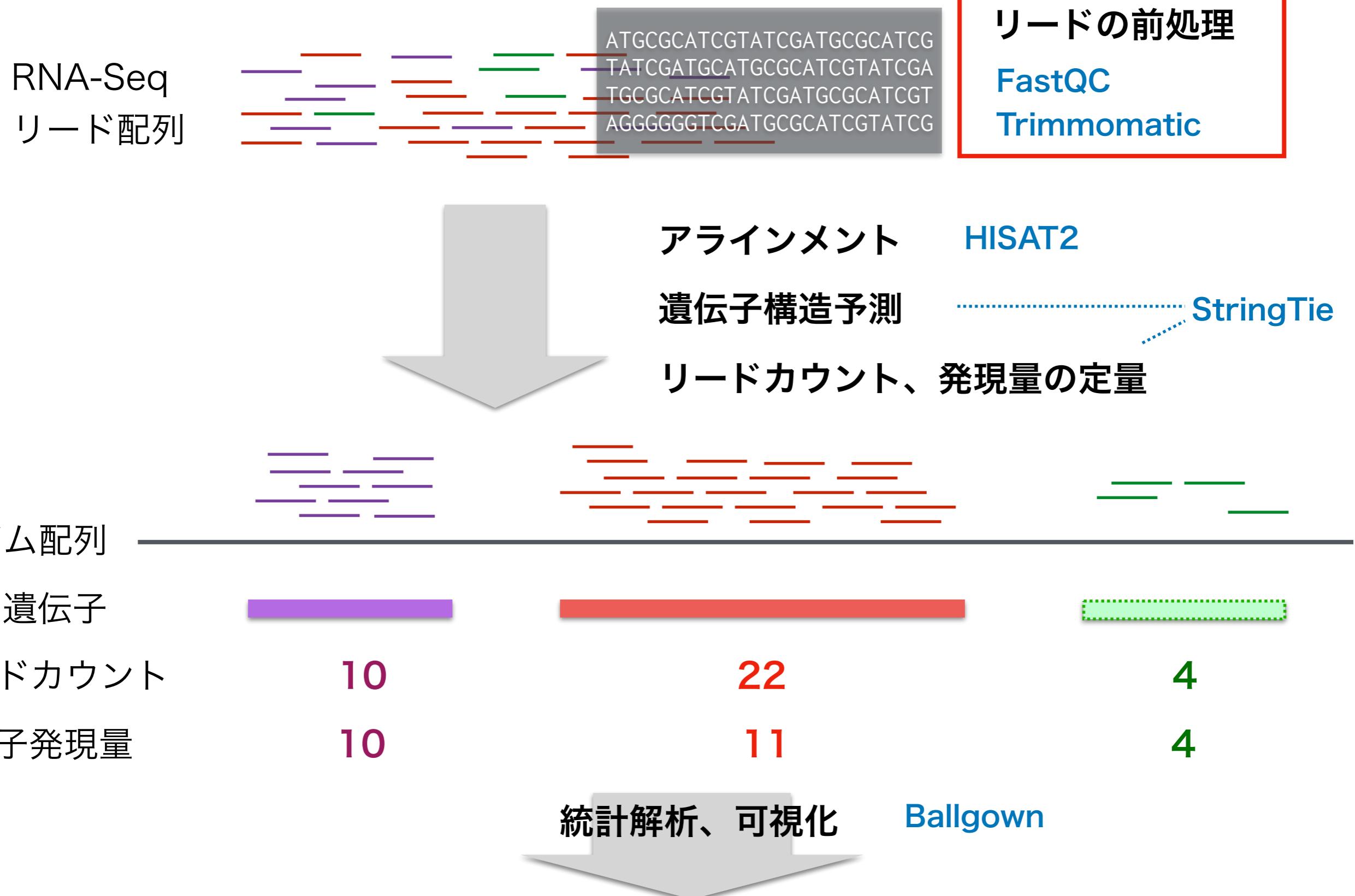
⋮ (省略)

`--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ことが多い。

遺伝子発現解析の流れ



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 2. 前処理 - FastQCなどによる配列データの確認

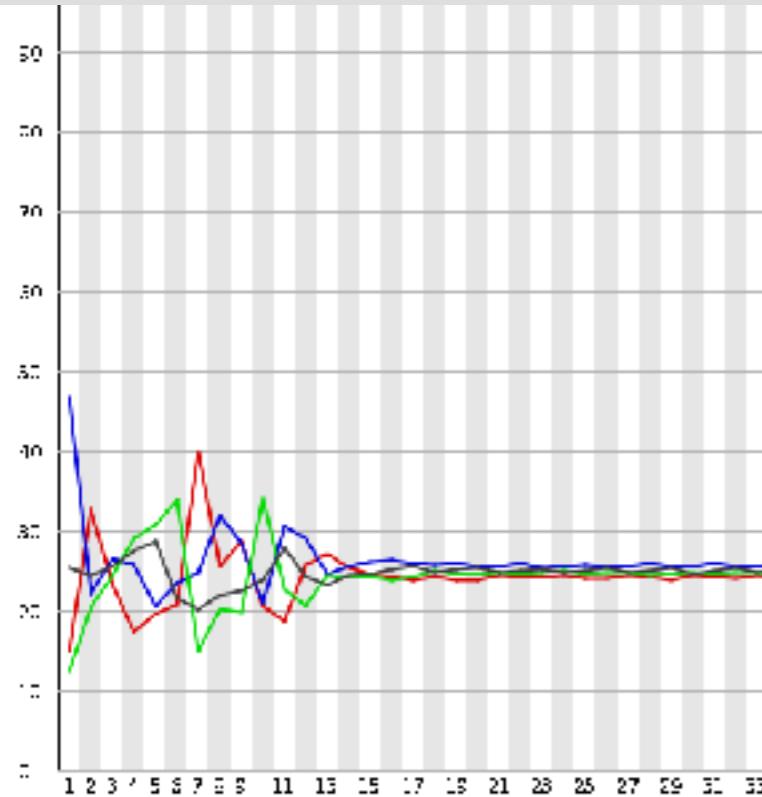
誤ったデータから正しい結果は得られない。ゲノムシークエンスデータと同様に、FastQCなどのツールを使い、リード数や解読塩基のクオリティ、バイアス、配列長などの事前チェックは重要！

RNA-Seqデータの特徴

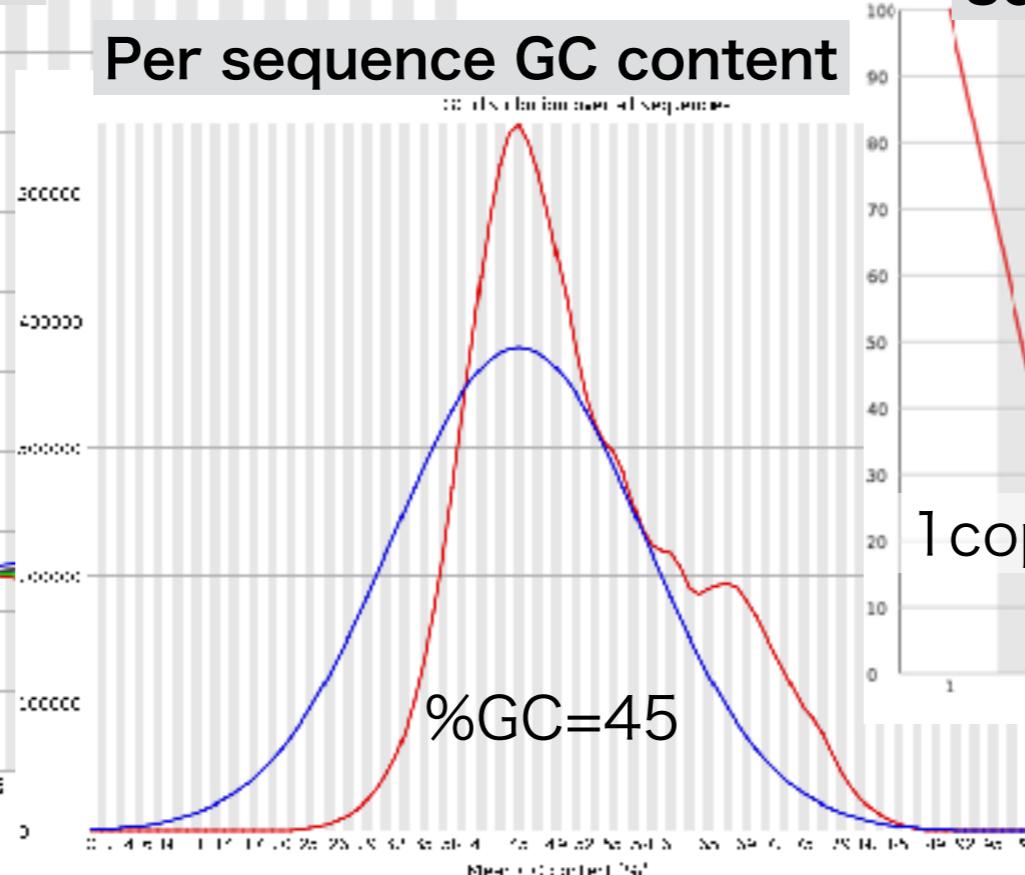
- ・先頭部分の塩基頻度のバイアス
- ・GCバイアス
- ・高発現遺伝子によると思われる
Duplication levelの高さ

これらの特徴はゲノムシークエンスデータとは
やや異なるので注意。

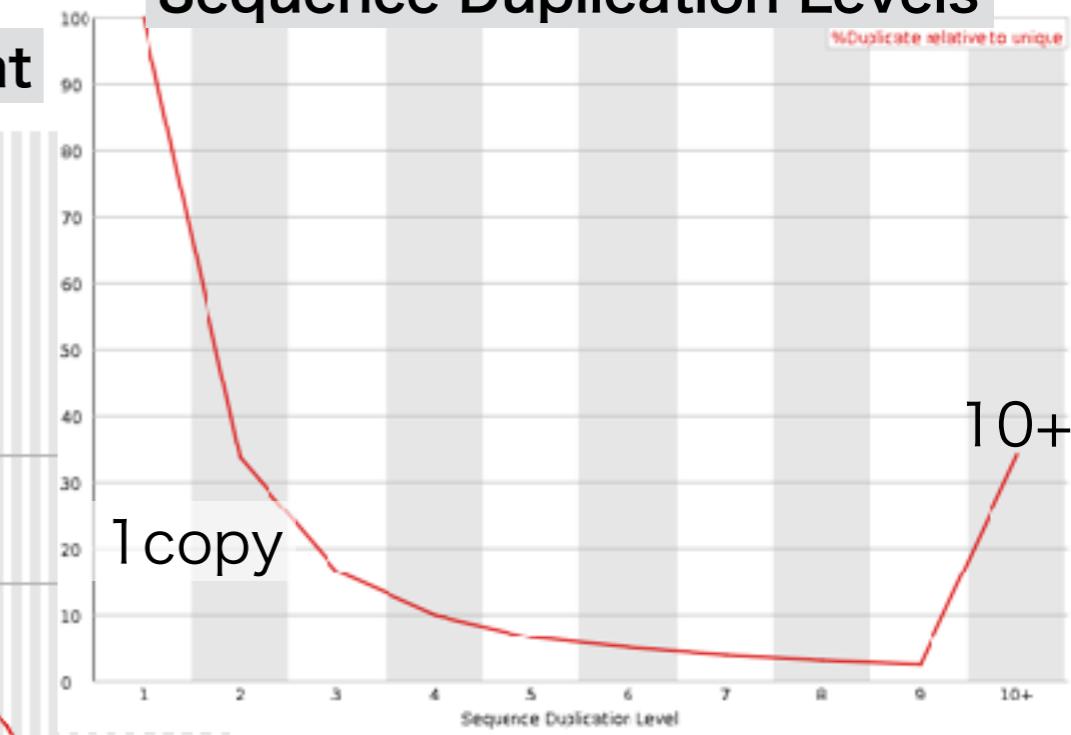
Per base sequence content



Per sequence GC content



Sequence Duplication Levels



(FastQCのレポートの一部)

Step 2. 前処理 - 配列データを綺麗にする

1. 低クオリティ塩基と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:4:15 MINLEN:30」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 (本演習ではやりません)

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]

これらの工程は必須ではないが、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問題がなかったかを確認するための指標になる。解析に用いる配列データに問題がないか把握しておくことは重要。

Step 2. リードの前処理

- リードの前処理をするためのシェルスクリプト

```
$ less step2_preprocessing.sh
```

- step2_preprocessing.shの前半部分 -

```
TOOL_HOME=./tool
FASTQC_HOME=$TOOL_HOME/FastQC
TRIMMOMATIC_HOME=$TOOL_HOME/Trimmomatic-0.36
export PATH=$FASTQC_HOME:$PATH
### Step.2: preprocessing by FastQC and Trimmomatic
FASTQC_OUTDIR_BEFORE=FastQC_before_preprocess
FASTQC_OUTDIR_AFTER=FastQC_after_preprocess
mkdir $FASTQC_OUTDIR_BEFORE $FASTQC_OUTDIR_AFTER
```

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。

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などが生じることを避けたり、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. 2: リードの前処理

- step2_preprocessing.shの後半部分 -

for関数による繰り返し師の指定。変数DATASETに
in以降のサンプル名が順番にセットされて繰り返される。

```
for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_D_rep4 rice_N_rep1 rice_N_rep2  
rice_N_rep3 rice_N_rep4
```

```
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 ${DATASET}_r*.org.fastq.gz
```

```
# Trimmomatic ----- Trimmomaticの実行
```

```
java -Xmx4G -Xms2G -jar $TRIMMOMATIC_HOME/trimmomatic-0.36.jar PE \  
-phred33 -trimlog Trimmomatic_${DATASET}.log \  
${DATASET}_r1.org.fastq.gz ${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_HOME/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 --nogroup --outdir $FASTQC_OUTDIR_AFTER --format fastq ${DATASET}_r*.pe.fastq.gz
```

```
done ----- 繰り返し終了
```

\ (バックスラッシュ) は
見た目を整えるために改行
する際に用いる。実行時
は無視され、ひと続きの
コマンドとして実行される。

- サンプル数が8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。
- 演習用のシェルスクリプトではサンプル1つ分のみ処理を実行するよう、for文の1行目が変更されている。

Step. 2: リードの前処理

- シェルスクリプトの実行

```
$ bash ./step2_preprocessing.sh 2> step2_preprocessing.stderr > \
step2_preprocessing.stdout
```

(実行時間：約8分、1サンプルのみだと：約1分)

- 出力ファイルの確認

```
$ ls -lh
drwxr-xr-x 2 guest01 guest01 4.0K 9月 27 13:45 2016 FastQC_after_preprocess
drwxr-xr-x 2 guest01 guest01 4.0K 9月 27 13:45 2016 FastQC_before_preprocess
...
...
-rw-r--r-- 1 guest01 guest01 11M 9月 26 15:56 2016 rice_D_rep1_r1.org.fastq.gz
-rw-r--r-- 1 guest01 guest01 11M 9月 27 13:43 2016 rice_D_rep1_r1.pe.fastq.gz
-rw-r--r-- 1 guest01 guest01 20 9月 27 13:43 2016 rice_D_rep1_r1.unpe.fastq.gz
-rw-r--r-- 1 guest01 guest01 11M 9月 26 15:56 2016 rice_D_rep1_r2.org.fastq.gz
-rw-r--r-- 1 guest01 guest01 11M 9月 27 13:43 2016 rice_D_rep1_r2.pe.fastq.gz
-rw-r--r-- 1 guest01 guest01 20 9月 27 13:43 2016 rice_D_rep1_r2.unpe.fastq.gz
...
```

FastQC_before_preprocess:

前処理前のリードのFastQCの結果

FastQC_after_preprocess:

前処理後のリードの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_*.org.fastq.gz:

オリジナルのリード配列ファイル

● 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出力に書き出されており、以下のように確認できる。

```
$ less step2_preprocessing.stderr
```

...

```
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%)
```

...

*この例は演習用データではなく、その元となった全データによる結果。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ものだけを抽出している。

- ・ 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。

前処理で見つかる問題と対応策

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。

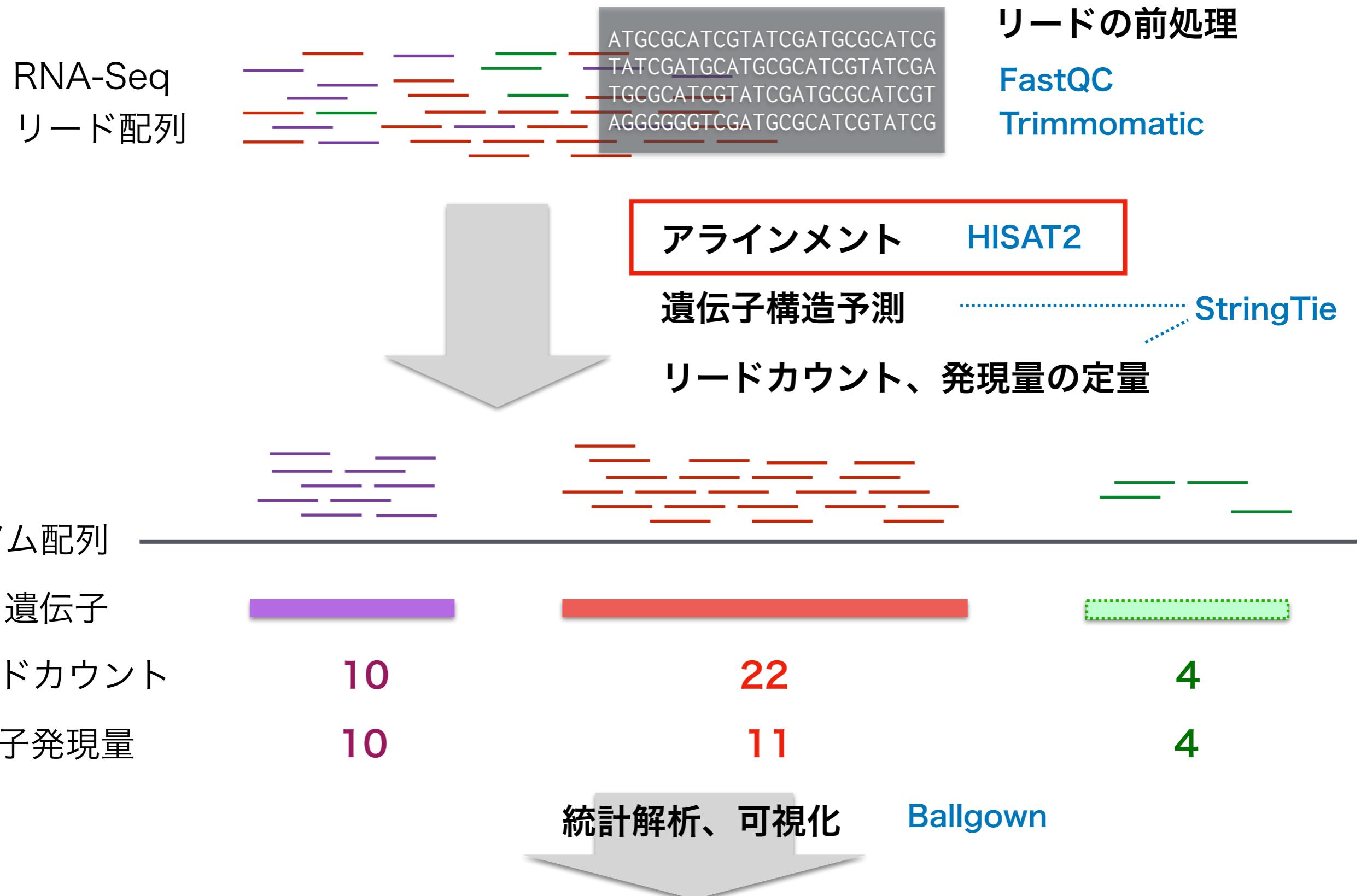
4. ゲノム、転写産物配列へのマップ率が低い

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）リードもあれば十分だと思う。

遺伝子発現解析の流れ



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

HISAT、StringTie、Ballgownを使ったRNA-Seq解析のプロトコル

Pertea et al. Nature Protocol 2016 11(9)

PROTOCOL

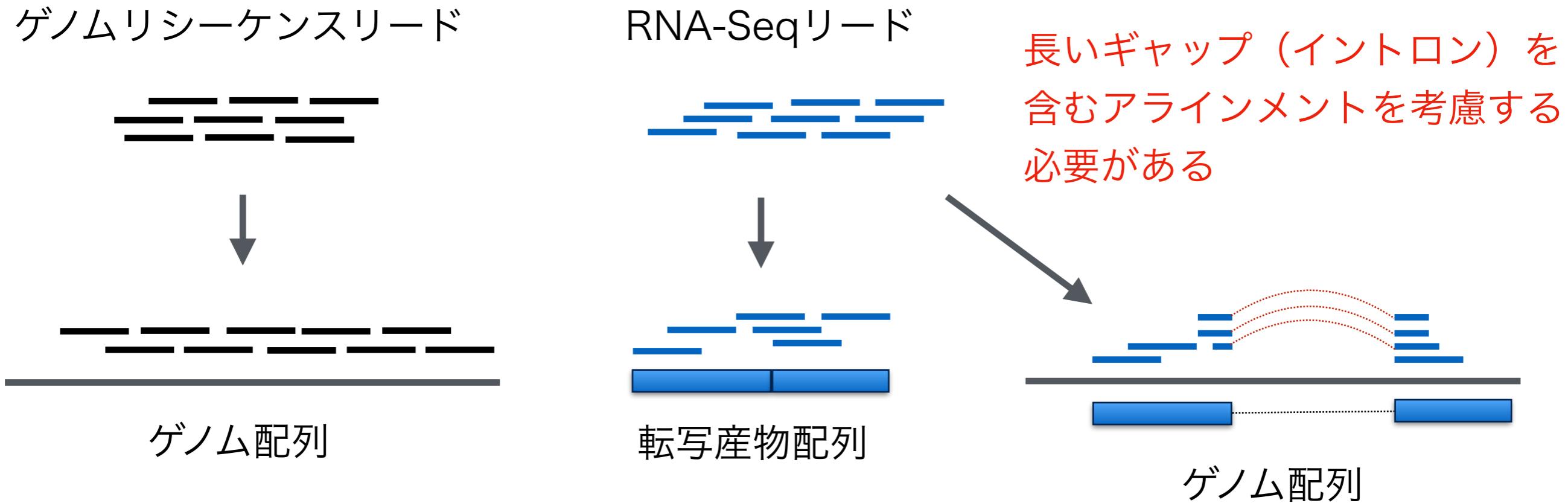
Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown

Mihaela Pertea^{1,2}, Daehwan Kim¹, Geo M Pertea¹, Jeffrey T Leek³ & Steven L Salzberg¹⁻⁴

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

- ・ 1年前に公開されたプロトコルのため、最新版のツールの使い方とは異なる部分もある。
- ・ ツールは更新され続けているため、論文や書籍に書かれている静的な情報はどんどん古くなっていく。

HISAT2によるsplice-awareアラインメント



- ・ ゲノムリシーケンス解析でよく使われるBWAやBowtie、Novoalign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 (HISAT、TopHat、STAR、GSNAPなど) が開発されている。
- ・ HISAT2はGlobal SearchとLocal Searchのための2種類のインデックス (hierarchical indexing) を作成しておき、GlobalとLocal Search、Extensionを切り替えながら高速に最適なアラインメントを探索する (Kim, D., Langmead, B., & Salzberg, S. L. (2015) Nature Methods, 12(4)) 。

Step 3. 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 step3_HISAT2-build.sh
```

- step3_HISAT2-build.sh -

```
TOOL_HOME=./tool
HISAT2_HOME=$TOOL_HOME/hisat2-2.1.0
SAMTOOLS_HOME=$TOOL_HOME/samtools-1.6
```

各ツールのコマンドが置かれている
ディレクトリへのパスを変数に格納

```
export PATH=$HISAT2_HOME:$SAMTOOLS_HOME:$PATH
```

exportコマンドで各ツールの
ディレクトリをPATHに追加。

```
### Step.3: Build index of the reference genome sequence by hisat2-build and samtools
# build index for HISAT2
hisat2-build --ss ss.tab --exon exon.tab genome.fa genome
# build index for IGV etc.
samtools faidx genome.fa
```

HISAT2のためのインデックス
の作成

IGVなどによる可視化に必要なゲノム配列のインデックス作成

- HISAT2用のhierarchical indexingの作成時に、既知のスプライスサイトやExonの位置情報 (ss.tab) を与えている（既存のアノテーションがなければ不要）。
- ss.tabやexon.tabは遺伝子アノテーション情報 (GTFファイル) から作成可能（後述）。

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

```
$ bash ./step3_HISAT2-build.sh 2> step3_HISAT2-build.stderr > \
step3_HISAT2-build.stdout
```

(実行時間：約3分)

- ・ 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.fa  
genome.2.ht2  genome.4.ht2  genome.6.ht2  genome.8.ht2  
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などでの可視化に必要なインデックス

Step. 3: HISAT2のためのインデックスの作成（補足）

- ・スプライスサイトとExonの位置情報の作成

```
$ python tool/hisat2-2.1.0/extract_splice_sites.py annotation.gtf > ss.tab
```

```
$ python tool/hisat2-2.1.0/extract_exons.py annotation.gtf > exon.tab
```

annotation.gtf : 遺伝子アノテーション情報 (GTF)

- ・既知のスプライスサイトなどの情報をインデックス作成に含めることでアラインメントの精度と効率を良くする。
- ・遺伝子アノテーション情報 (GTF) があれば、hisat2のディレクトリ中にあるPythonスクリプトを用いて作成可能。

*演習用のサーバにインストールされているpythonのバージョンが古く、これらのスクリプトは動作しないので注意。

遺伝子アノテーションファイル (GTF2/GFF3)

RAP-DBからイネの代表転写産物のアノテーション情報 (GFF3) がダウンロードできる。

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

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, 13.6MB)
- Gene sequences (CDS + UTRs + introns) in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 39.5MB)
- Transcript sequences (CDS + UTRs) in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 20.8MB)
- CDS sequences in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 12.4MB)
- Protein sequences (translated CDSs) in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 7.7MB)
- 1 kb upstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 13.7MB)
- 2 kb upstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 26.9MB)
- 3 kb upstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 39.5MB)
- 1 kb downstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 13.5MB)
- 2 kb downstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 26.4MB)
- 3 kb downstream sequences of genes in FASTA format.
[\[DOWNLOAD\]](#) (gz file, 38.8MB)

- 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。

色々な生物のアノテーションファイルを提供しているサイト

Illumina社 iGenomes

https://support.illumina.com/sequencing/sequencing_software/igenome.html

EnsemblPlants

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

Phytozome

<http://phytozome.jgi.doe.gov/pz/portal.html>

HISAT2やStringTieで使う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 annotation.gtf
chr01    irgsp1_rep    transcript    152853  156449  .      +      .      gene_id
"0s01g0102800"; transcript_id "0s01t0102800-01"; gene_name "CSB, OsCBS"; note "Similar
to chromatin remodeling complex subunit.";
chr01    irgsp1_rep    exon     152853  153025  .      +      .      gene_id
"0s01g0102800"; transcript_id "0s01t0102800-01";
chr01    irgsp1_rep    exon     153178  154646  .      +      .      gene_id
"0s01g0102800"; transcript_id "0s01t0102800-01";
chr01    irgsp1_rep    exon     155010  155450  .      +      .      gene_id
"0s01g0102800"; transcript_id "0s01t0102800-01";
chr01    irgsp1_rep    exon     155543  156449  .      +      .      gene_id
"0s01g0102800"; transcript_id "0s01t0102800-01";
```

- ・ RAP-DBから提供されているGFF3ファイルをGTF形式に整形する必要がある。
- ・ 「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、結果ファイルに出力されるので便利。

Step 4. HISAT2によるRNA-Seqリードのアラインメント

- 8サンプル（2条件、4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 step4_HISAT2.sh
```

- step4_HISAT2.sh -

```
### Step.4: Align reads to the reference genome by HISAT2
HISAT2_COMMON_PARAM="--min-intronlen 20 --max-intronlen 10000 --downstream-
transcriptome-assembly --rna-strandness RF -x genome"           全サンプルに共通のパラメータを
                                                               変数に格納

for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_D_rep4 rice_N_rep1 rice_N_rep2
rice_N_rep3 rice_N_rep4 ..... for関数による繰り返し師の指定。変数DATASETに
do ..... in以降のサンプル名が順番にセットされて繰り返される。
      繰り返し開始

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

    samtools sort -o ${DATASET}.bam ${DATASET}.sam
    samtools index ${DATASET}.bam
    rm ${DATASET}.sam                                         SAM形式で出力されたアラインメントを
                                                               BAM形式に変換し、インデックスを作成

done ..... 繰り返し終了
```

Step 4. 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

```
$ bash ./step4_HISAT2.sh 2> step4_HISAT2.stderr > step4_HISAT2.stdout
```

(実行時間：約3分)

- ・ 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_rep4.bam      rice_N_rep3.bam
rice_D_rep1.bam.bai  rice_D_rep4.bam.bai  rice_N_rep3.bam.bai
rice_D_rep2.bam      rice_N_rep1.bam      rice_N_rep4.bam
rice_D_rep2.bam.bai  rice_N_rep1.bam.bai  rice_N_rep4.bam.bai
rice_D_rep3.bam      rice_N_rep2.bam
rice_D_rep3.bam.bai  rice_N_rep2.bam.bai
```

- ・ *.bamファイルはHISAT2によるアラインメントファイル
- ・ *.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ている。

```
$ less step4_HISAT2.stderr
```

16992068 reads; of these:

16992068 (100.00%) were paired; of these:

489987 (2.88%) aligned concordantly 0 times

14716422 (86.61%) aligned concordantly exactly 1 time

1785659 (10.51%) aligned concordantly >1 times

489987 pairs aligned concordantly 0 times; of these:

239950 (48.97%) aligned discordantly 1 time

250037 pairs aligned 0 times concordantly or discordantly; of these:

500074 mates make up the pairs; of these:

343334 (68.66%) aligned 0 times

103649 (20.73%) aligned exactly 1 time

53091 (10.62%) aligned >1 times

98.99% overall alignment rate

*この例は演習用データではなく、その元となった全データによる結果。

- ・ 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、何も問題がなければ90%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

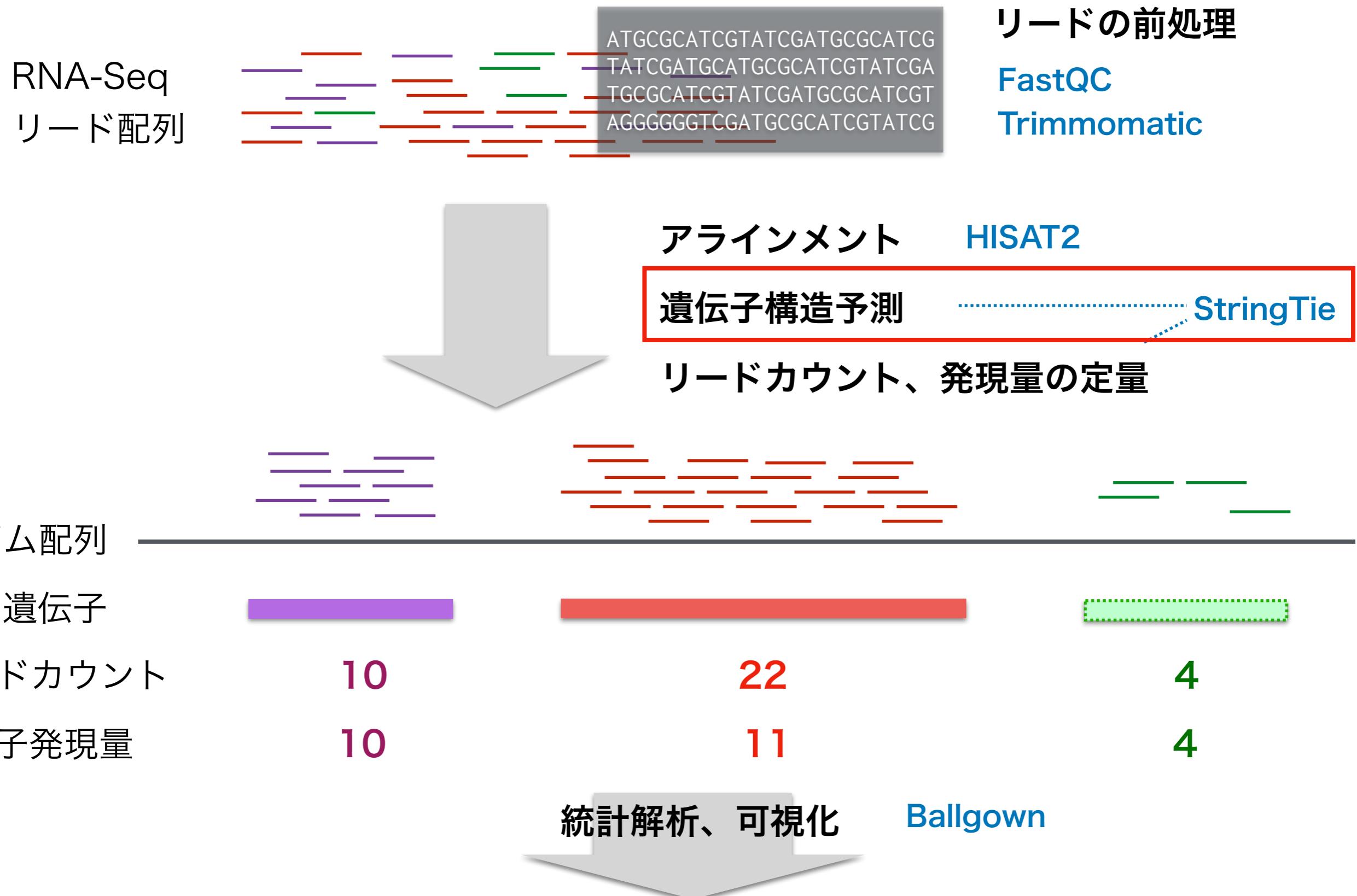
SAMtoolsを使い、個々のリードのアラインメント情報を見ることが出来る。

```
$ tool/samtools-1.6/samtools view ./rice_D_rep1.bam | less
```

MG00HS14:530:C6M7YACXX:8:2208:4438:63887	161	chr02	14245740	60
67M = 31527665	17282026			
TGAACGCTGGCGGCATGCTAACACATGCAAGTCGAACGGGAAGTGGTGTTCCAGTGGCGAACGGG				
@@@A7DDDD811@EAFGEB9?BAD38B>FDFDFFEIFA@BEB>E==@@;?=>AADCABABBBB'05B			AS:i:0	XN:i:0
XM:i:0 X0:i:0 XG:i:0 NM:i:0 MD:Z:67 YS:i:-2 YT:Z:DP XS:A:+ NH:i:1				
MG00HS14:530:101M = CIGAR 75M 112N 21M		chr02	23112840	60
CATCCCTGTCCC リード ——————→ 1S			ATGTGTTCTCCATCACCGGTCGTGGTACCGTTG	
CCACTGGCCGTA ゲノム配列 ——————			==CC>@7==@@E=;?=?	
DCBB;@(;@CCC S:i:0 XN:i:0 XM:i:0 X0:i:0			XS:i:0	
XG:i:0 NM:i:0 + NH:i:1				
MG00HS14:510:C6M7YACXX:8:1103:1603:42247	113	chr02	23845634	60
75M112N21M1S = 31055654 7210007				
GTCAGCTTCGCTGCCGCCGTGCTGGGGCTCCTCGCAGCCATCCTCGGGTCGCGGAAGGCGCCAAGTCCAATTGTTCGTGC				
GTTCGACGGGG C@>CB>><BDB7DDDBDDC?8(B?BA?B@>;DDDCC@A?<DDDDDDDB@B??				
<FFFHHGEJJIIIIJIIIEFJIHFIIIHGGJIGAFHHFFFFFCCC AS:i:-14 XN:i:0 XM:i:0				
X0: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ゲノム解析の場合と同じ（詳しくは、<http://samtools.github.io/hts-specs/SAMv1.pdf>などを参照）。
- ・ 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の意味。

遺伝子発現解析の流れ



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 5. StringTieによる遺伝子構造のアセンブル

- 8サンプル（2条件、4反復）のそれぞれのアラインメント結果を元に遺伝子構造をアセンブルするためのシェルスクリプト

```
$ less step5_StringTie-assemble.sh
```

- step5_StringTie-assemble.sh -

```
### Step.5: Assemble transcript structures by StringTie
STRINGTIE_COMMON_PARAM="" ----- 全サンプルに共通のパラメータを
for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_D_rep4 rice_N_rep1 rice_N_rep2
rice_N_rep3 rice_N_rep4
do
    stringtie $STRINGTIE_COMMON_PARAM -o ${DATASET}.gtf -l ${DATASET} ${DATASET}.bam ----- 変数に格納。今回は未設定。
done
```

出力ファイルとラベル、アラインメント
ファイルを指定してStringTieを実行

Step 5. StringTieによる遺伝子構造のアセンブル

- ・ シェルスクリプトの実行

```
$ bash ./step5_StringTie-assemble.sh 2> step5_StringTie-assemble.stderr > \
step5_StringTie-assemble.stdout
```

(実行時間：約1分)

- ・ 作成された遺伝子構造データ（GTFファイル）

```
$ ls *.gtf
annotation.gtf    rice_D_rep3.gtf    rice_N_rep2.gtf
rice_D_rep1.gtf   rice_D_rep4.gtf    rice_N_rep3.gtf
rice_D_rep2.gtf   rice_N_rep1.gtf    rice_N_rep4.gtf
```

- ・ rice_*.gtfが、個々のサンプルのアラインメントデータを元にアセンブルされた遺伝子構造情報
- ・ annotation.gtfはリファレンスの遺伝子構造

StringTieによる遺伝子構造のアセンブル結果の確認

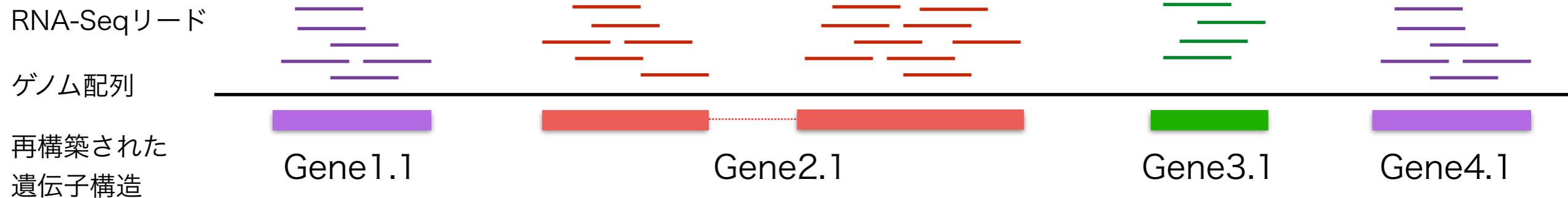
- 作成された遺伝子構造データ (GTF) の確認

```
$ less rice_D_rep1.gtf
# stringtie -o rice_D_rep1.gtf -l rice_D_rep1 rice_D_rep1.bam
# StringTie version 1.3.3b
chr02  StringTie      transcript    29999629      30001439      1000      -
.        gene_id "rice_D_rep1.1"; transcript_id "rice_D_rep1.1.1"; cov "9.875343"; FPKM
"359.762177"; TPM "593.122803";
chr02  StringTie      exon        29999629      29999679      1000      -
.        gene_id "rice_D_rep1.1"; transcript_id "rice_D_rep1.1.1"; exon_number "1"; cov
"2.408823";
chr02  StringTie      exon        29999768      29999884      1000      -
.        gene_id "rice_D_rep1.1"; transcript_id "rice_D_rep1.1.1"; exon_number "2"; cov
"9.495370";
chr02  StringTie      exon        30000045      30000174      1000      -
.        gene_id "rice_D_rep1.1"; transcript_id "rice_D_rep1.1.1"; exon_number "3"; cov
"12.716667";
chr02  StringTie      exon        30000373      30000536      1000      -
.        gene_id "rice_D_rep1.1"; transcript_id "rice_D_rep1.1.1"; exon_number "4"; cov
"8.900915";
...
...
```

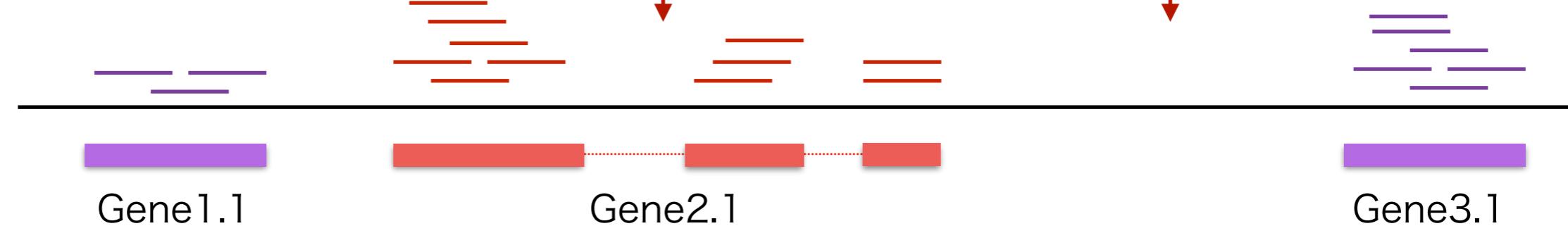
アセンブルされた遺伝子のExonの位置情報、リードのcoverage情報、発現量 (RPKMとTPM) などが記載されている。

サンプルごとに再構築された遺伝子構造を1つにまとめる

サンプル1

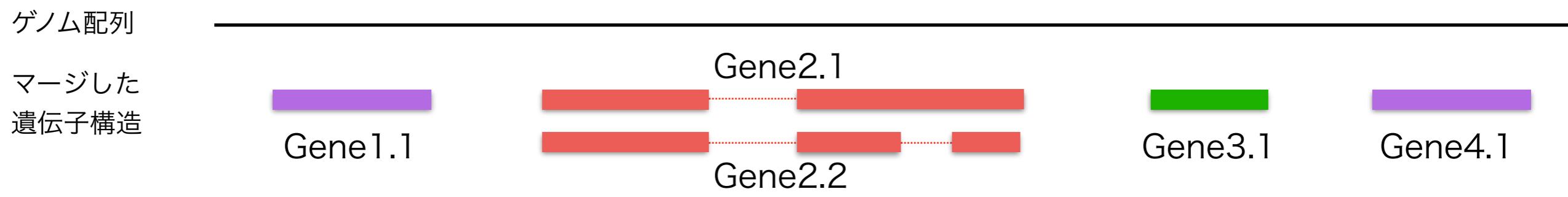


サンプル2



||

サンプル1+2



Step 6. StringTieによる遺伝子構造の統合

- サンプルごとの遺伝子構造を1つの遺伝子構造に統合するためのスクリプト

```
$ less step6_StringTie-merge.sh
```

```
### Step.6: Merge assembled transcript structures by StringTie
# make the list of assembled transcripts files
ls ./rice_*.gtf > assemblies.txt ----- 統合する遺伝子構造 (GTF) のリストを作成

# merge assembled transcripts
stringtie --merge -G annotation.gtf -o stringtie_merged.gtf assemblies.txt
                                         ↘ リファレンスアノテーション中の遺伝子構造も
                                         合わせて統合する。
```

-Gオプションでリファレンスアノテーション情報を与えることで、既存の遺伝子構造も合わせて統合することも可能。

- シェルスクリプトの実行

```
$ bash ./step6_StringTie-merge.sh 2> step6_StringTie-merge.stderr > \
step6_StringTie-merge.stdout
```

(実行時間：約10秒)

Step 7. リファレンスとRNA-Seqデータに基づいた遺伝子構造の比較

- RNA-Seqデータに基づいてアセンブルした遺伝子構造とリファレンスアノテーション中の遺伝子構造を比較するシェルスクリプト

```
$ less step7_gffcompare.sh
```

- step7_gffcompare.sh -

```
# ### Step.7 : Examine how the transcripts compare with the reference annotation
gffcompare -r annotation.gtf -o gffcmp stringtie_merged.gtf
```

リファレンスアノテーション中の遺伝子構造も
合わせて統合した遺伝子構造セットの検証

リファレンスアノテーションとの比較により、作成した遺伝子構造セットの精度を検証することが可能。

- シェルスクリプトの実行

```
$ bash ./step7_gffcompare.sh 2> step7_gffcompare.stderr > \
step7_gffcompare.stdout
```

(実行時間：約10秒)

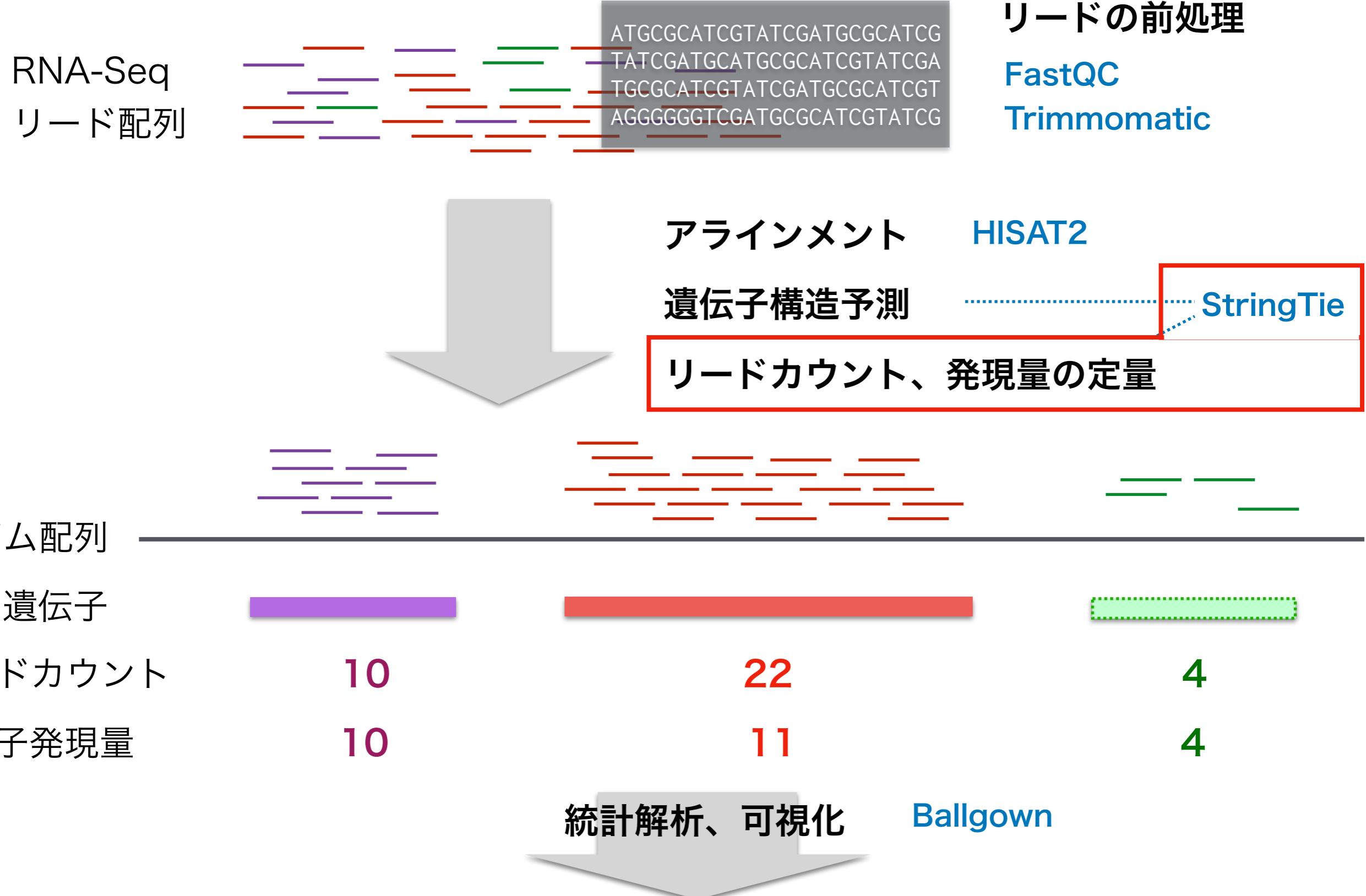
Step. 7: リファレンスとRNA-Seqデータに基づいた遺伝子構造の比較

比較結果の統計情報

```
$ less gffcmp.stats
# gffcompare v0.10.1 | Command line was:
#gffcompare -r annotation.gtf -o gffcmp stringtie_merged.gtf
#
#= Summary for dataset: stringtie_merged.gtf
#   Query mRNAs : 689 in 393 loci (508 multi-exon transcripts)
#       (114 multi-transcript loci, ~1.8 transcripts per locus)
# Reference mRNAs : 347 in 295 loci (237 multi-exon)
# Super-loci w/ reference transcripts: 281
#-----| Sensitivity | Precision |
#      Base level: 100.0 | 76.2 |
#      Exon level: 99.5 | 66.5 |
#      Intron level: 100.0 | 71.1 |
# Intron chain level: 97.5 | 45.5 |
# Transcript level: 98.3 | 49.5 |
# Locus level: 100.0 | 71.8 |
#
# Matching intron chains: 231
# Matching transcripts: 341
# Matching loci: 295
#
# Missed exons: 0/1373 ( 0.0%)
# Novel exons: 405/2237 ( 18.1%)
# Missed introns: 0/996 ( 0.0%)
# Novel introns: 290/1401 ( 20.7%)
# Missed loci: 0/295 ( 0.0%)
# Novel loci: 109/393 ( 27.7%)
#
# Total union super-loci across all input datasets: 393
# 689 out of 689 consensus transcripts written in gffcmp.annotated.gtf (0 discarded as redundant)
```

今回は遺伝子構造をマージする際にリファレンス情報も含めているため、Sensitivity（感度）はほぼ100%となっている。その一方、Precision（適合率）から、リファレンスにない新規の転写産物も少なからず検出されていることが分かる。

遺伝子発現解析の流れ



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 8. StringTieによる遺伝子ごとのリードのCoverageを計算

統合した遺伝子モデル（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のCoverage（発現量と比例）を計算するスクリプト。

```
$ less step8_StringTie-abundance_estimation.sh
```

- step8_StringTie-abundance_estimation.sh -

```
# ### Step.8: Estimate transcript abundances
STRINGTIE_COMMON_PARAM="-e -B"
for DATASET in rice_D_rep1 rice_D_rep2 rice_D_rep3 rice_D_rep4 rice_N_rep1 rice_N_rep2
rice_N_rep3 rice_N_rep4
do
    stringtie $STRINGTIE_COMMON_PARAM -G stringtie_merged.gtf -o ballgown/${DATASET}/${
${DATASET}}.gtf ${DATASET}.bam
done
```

統合した遺伝子モデルと出力ファイル、アラインメントデータを指定して、StringTieを実行し、coverageを計算。

Step 8. StringTieによる遺伝子発現量等の定量

- シェルスクリプトの実行

```
$ bash ./step8_StringTie-abundance_estimation.sh \
2> step8_StringTie-abundance_estimation.stderr \
> step8_StringTie-abundance_estimation.stdout
```

(実行時間：約1分)

- 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

Step 8. StringTieによる遺伝子発現量等の定量

- 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	t_name	num_exons	length	gene_id
gene_name		cov			FPKM			
...								
6	chr02	+	30011066		30015609		0s02t0722700-01	7
MSTRG.4	-	96.389221		3590.763916				1436
7	chr02	+	30011085		30015661		MSTRG.4.2	6
MSTRG.4	.	10.439097		388.885132				1320
8	chr02	+	30011089		30015638		MSTRG.4.3	6
MSTRG.4	.	8.876240		330.664398				3122
9	chr02	-	30015998		30025935		0s02t0722800-01	13
MSTRG.5	OsWD40-53		9.263726		345.099335			4485
10	chr02	-	30016149		30018289		0s02t0722800-02	6
MSTRG.5	-	17.567350		654.432251				1316
11	chr02	-	30044110		30048064		MSTRG.6.1	3
MSTRG.6	.	2.353367		87.669411				1381
12	chr02	-	30046134		30048041		MSTRG.6.2	2
MSTRG.6	.	1.070569		39.881645				1828

- 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。
- 赤点線 ----- は遺伝子座の区切り。

遺伝子発現量の指標 (RPKM/FPKM)

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／遺伝子の長さ(kb)／総リード数(M)

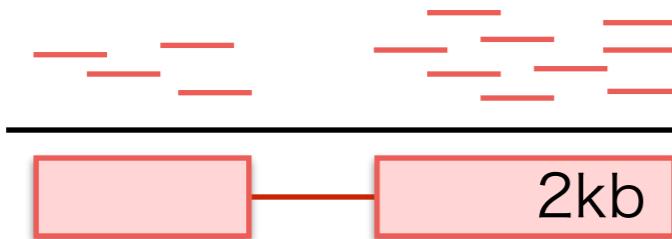
RPKM: reads per kilobase of exon model per million mapped reads

FPKM: fragments per kilobase of transcript per million fragments mapped

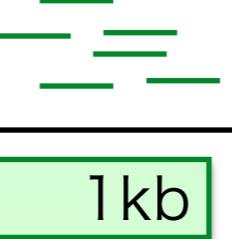
条件 1

リードカウント (≠発現量)

13リード

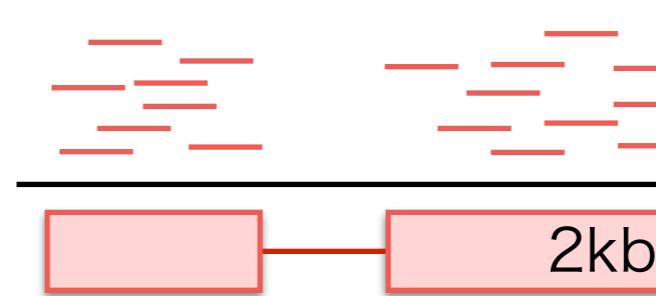


6リード

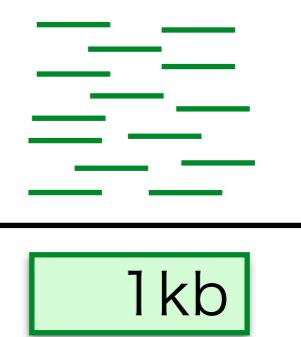


条件 2

18リード



14リード



RPKM: $13/2/19=0.34$ $6/1/19=0.32$

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。

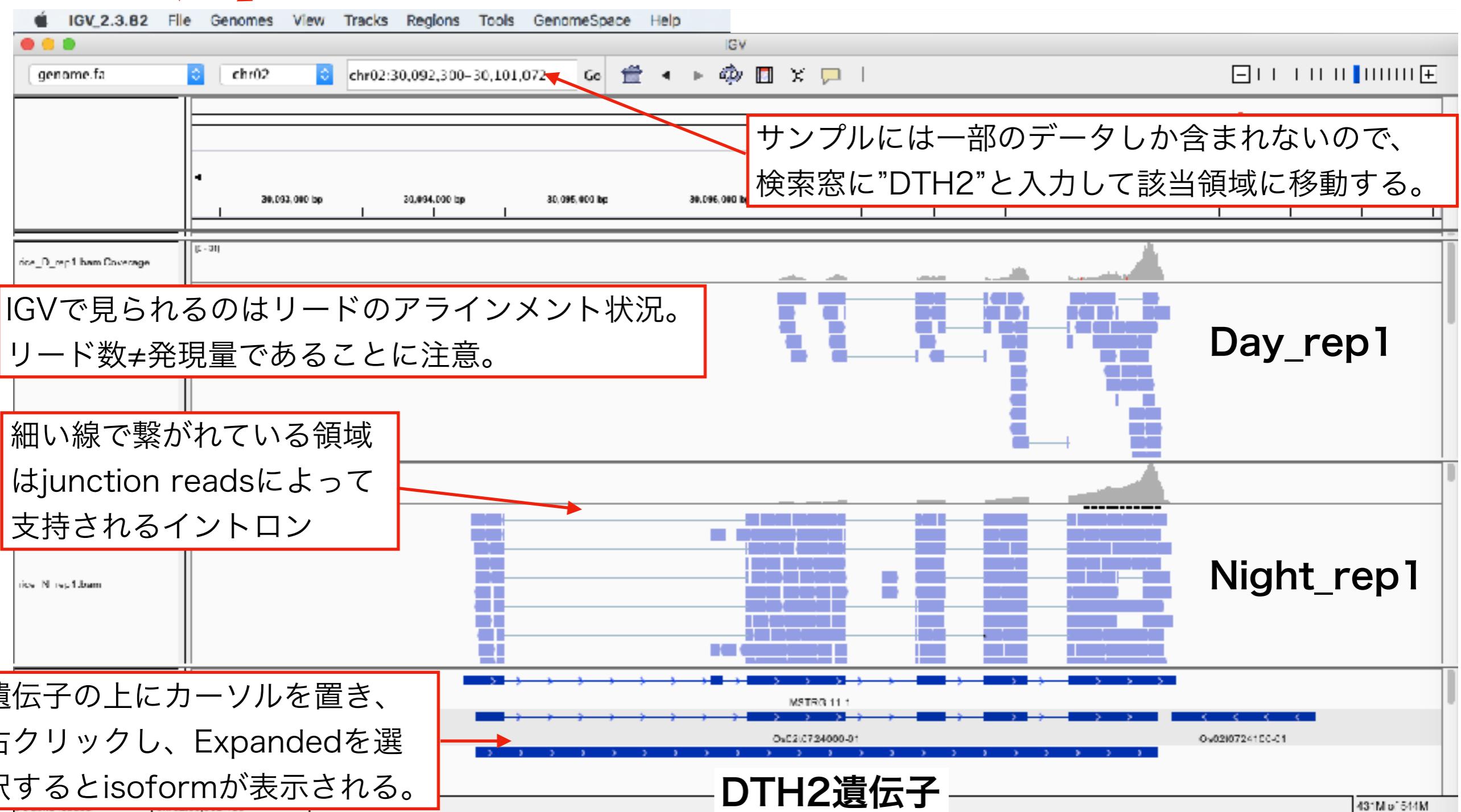
本演習で取り組む課題

1. リファレンス情報を用いた遺伝子発現解析（Linuxサーバ）
2. IGVやRなどを用いた発現比較解析と可視化（ローカルPC）

IGVによるRNA-Seqデータのアラインメント、遺伝子構造の可視化

デスクトップ上の「workshop/RNA-Seq_analysis/alignment」中のリファレンスゲノムとアノテーション、HISAT2やStringTieが出力するBAMファイルやGTF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。



ballgownによる統計解析と可視化

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。

R

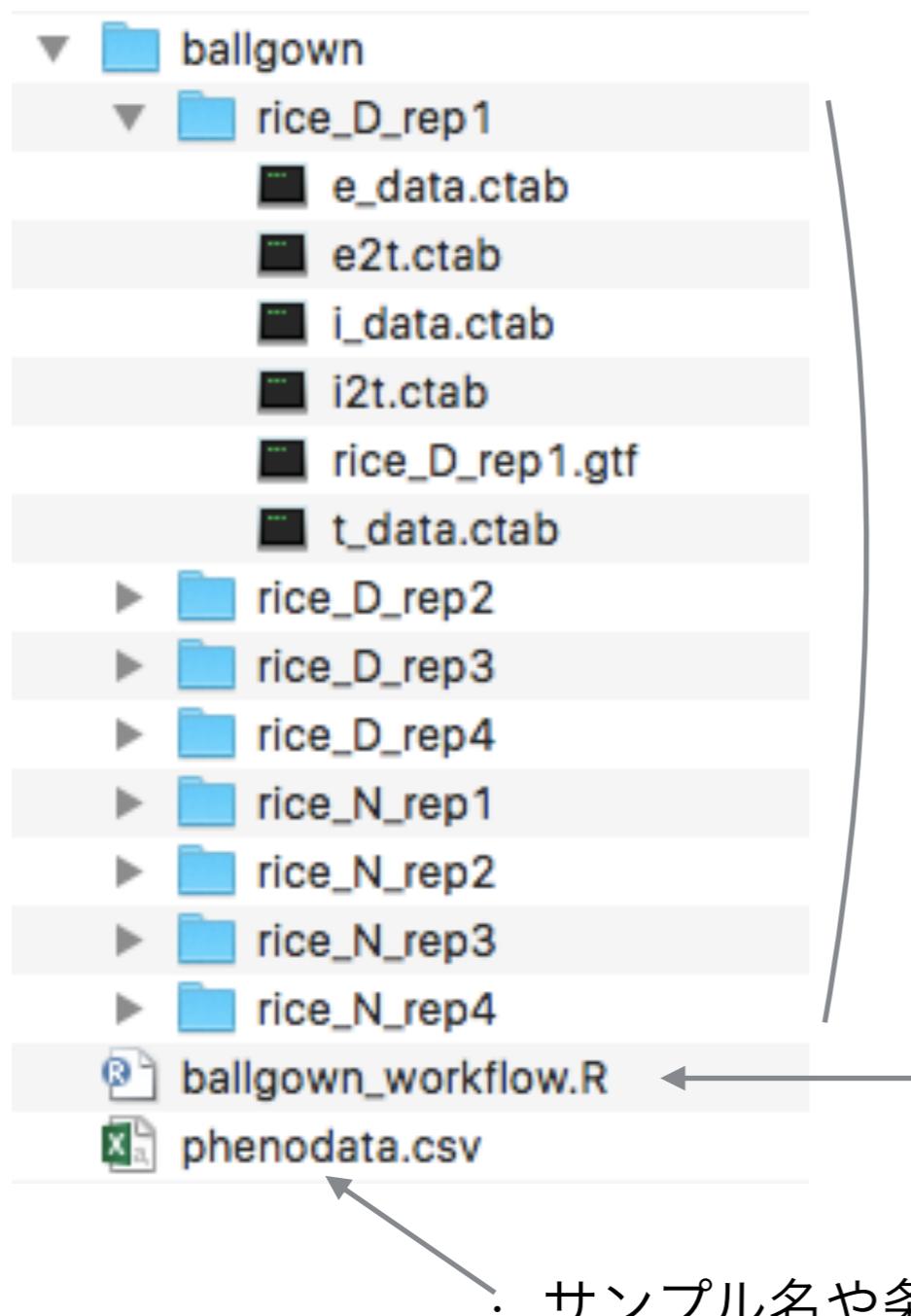
<https://www.r-project.org/>

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「workshop/RNA-Seq_analysis」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。

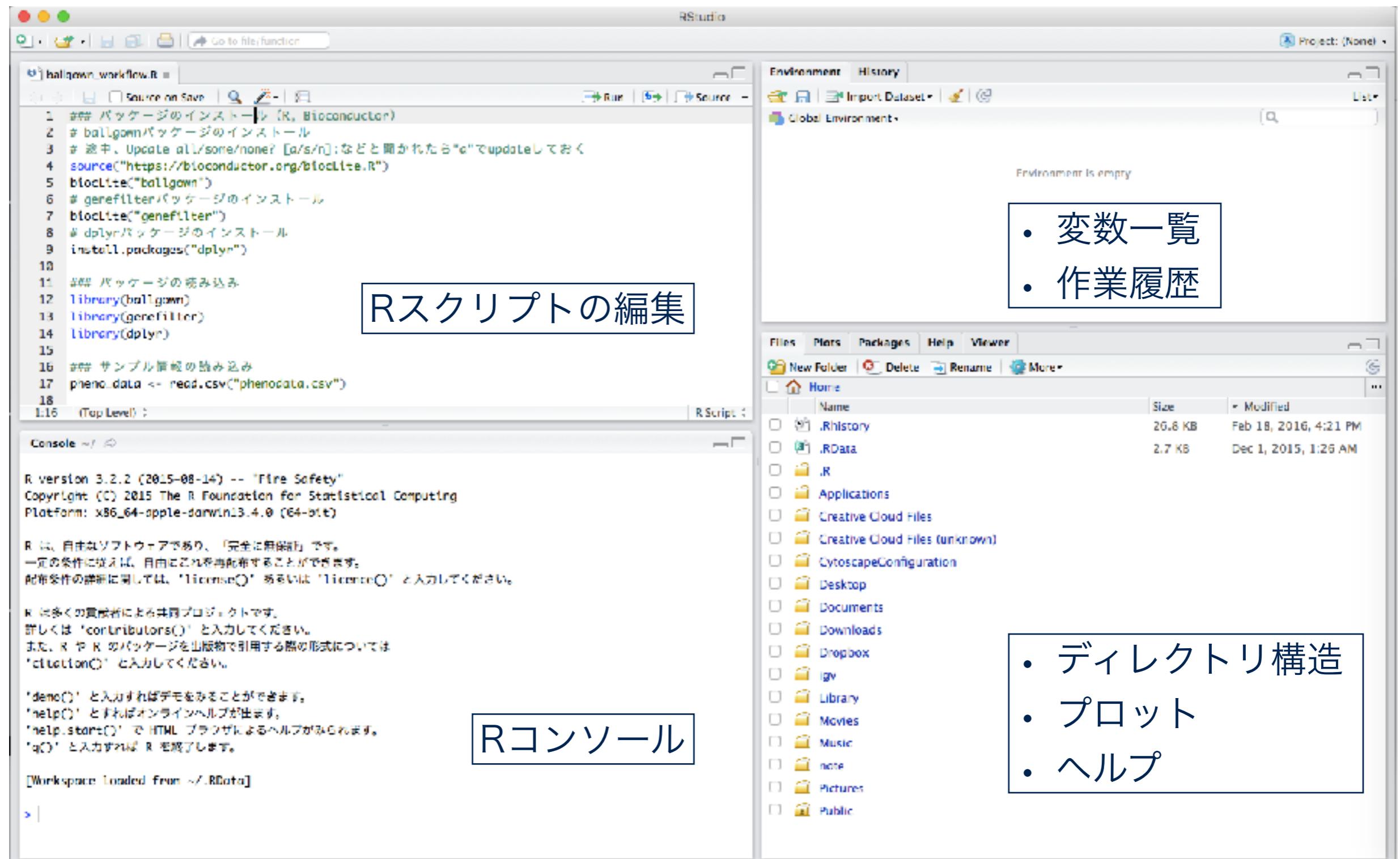


「ballgown」ディレクトリ中にある8サンプル分
(2条件、4反復) のStringTieの結果

- 実行するコマンドを記載したRのスクリプト
(ballgown_workflow.R)
- 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)

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



作業ディレクトリの指定

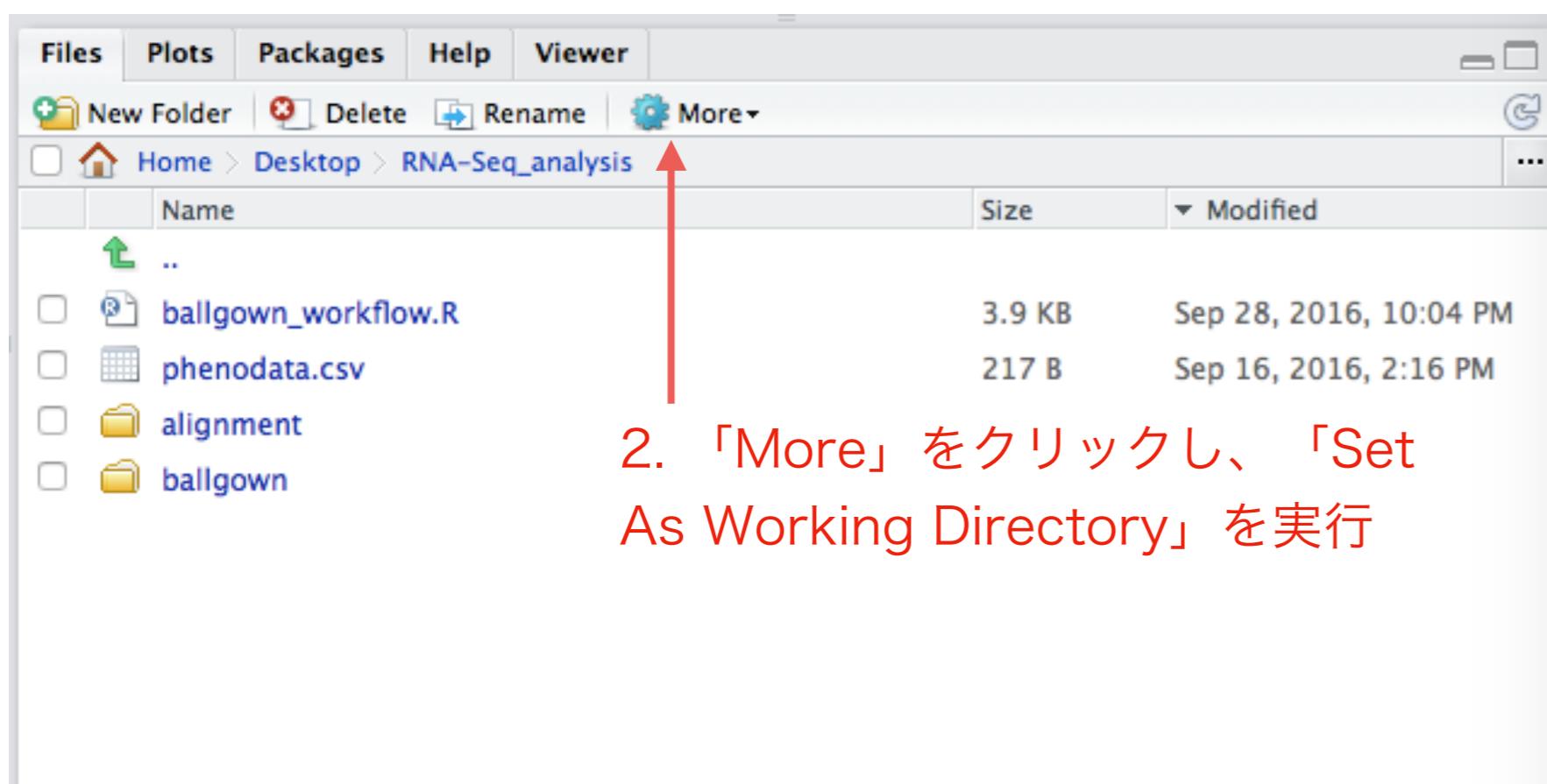
1. Rコンソール（左下のペイン）でコマンドを実行

```
> setwd("c:/Users/user/Desktop/workshop/RNA-Seq_analysis/")
```

ballgownディレクトリがある場所を指定する。

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



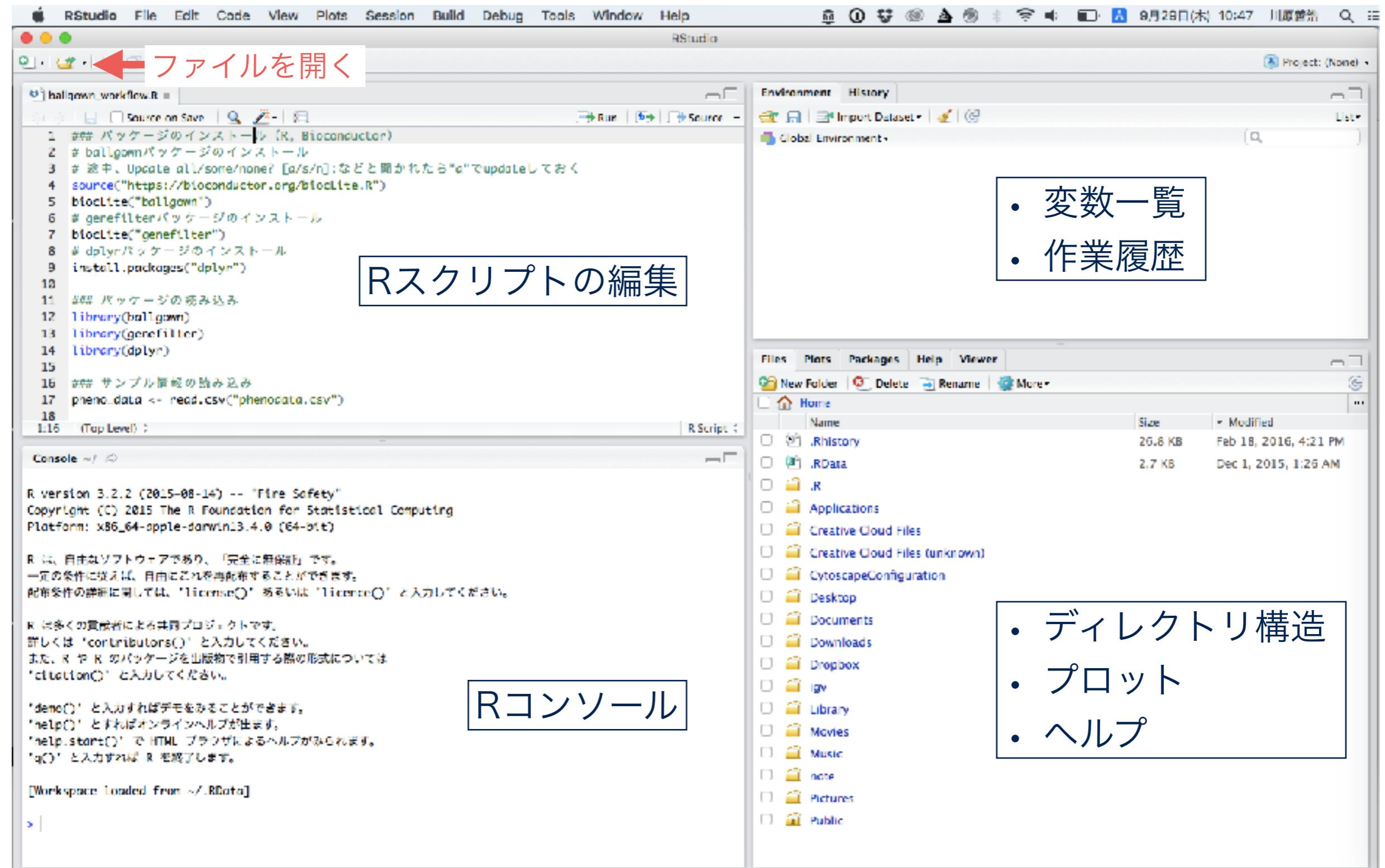
1. 「...」をクリックし、「RNA-Seq_analysis」ディレクトリを選択

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

演習用のRスクリプトを開く

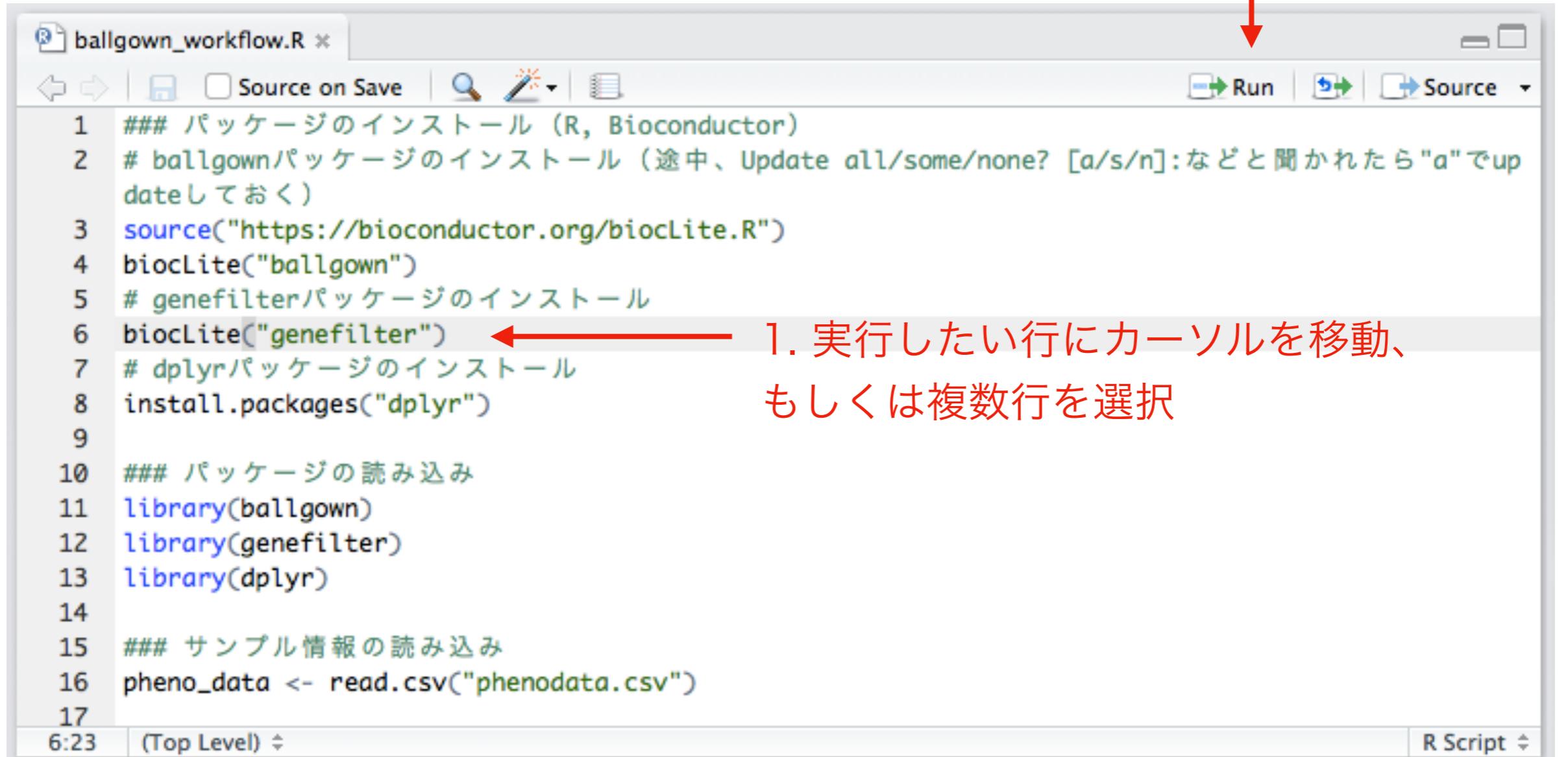
ファイルを開く、もしくは右下のペインでballgown_workflow.Rを選択して開く。

*日本語のコメント部分が文字化けするようなら、File -> Reopen with Encoding…から「UTF-8」を選択する。



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



```

1  ### パッケージのインストール (R, Bioconductor)
2 # ballgownパッケージのインストール (途中、Update all/some/none? [a/s/n]:などと聞かれたら "a" でup
  dateしておく)
3 source("https://bioconductor.org/biocLite.R")
4 biocLite("ballgown")
5 # genefilterパッケージのインストール
6 biocLite("genefilter") ← 1. 実行したい行にカーソルを移動、
7 # dplyrパッケージのインストール
8 install.packages("dplyr")
9
10 ### パッケージの読み込み
11 library(ballgown)
12 library(genefilter)
13 library(dplyr)
14
15 ### サンプル情報の読み込み
16 pheno_data <- read.csv("phenodata.csv")
17

```

1. 実行したい行にカーソルを移動、
もしくは複数行を選択

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パッケージのインストール

```
> source("https://bioconductor.org/biocLite.R")
> biocLite("ballgown")
```

- ・ ballgownはR/Bioconductorのパッケージとして提供されており、biocLite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。

さらに演習で使う2つのパッケージをインストールし、最後に3つのパッケージを読み込む。

```
> biocLite("genefilter")
> install.packages("dplyr")
>
> library(ballgown)
> library(genefilter)
> library(dplyr)
```

← ballgownと同様にR/Bioconductorのパッケージ
 ← Rのパッケージのため、install.packages関数を用いてインストールする。

*演習ではパッケージのインストールは完了しているため、最後の3行のパッケージの読み込みだけを行えばOK（なはず）。

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 Oct  3 06:41:22 2016
Mon Oct  3 06:41:22 2016: Reading linking tables
Mon Oct  3 06:41:24 2016: Reading intron data files
Mon Oct  3 06:41:33 2016: Merging intron data
Mon Oct  3 06:41:34 2016: Reading exon data files
Mon Oct  3 06:41:54 2016: Merging exon data
Mon Oct  3 06:41:56 2016: Reading transcript data files
Mon Oct  3 06:42:02 2016: Merging transcript data
Wrapping up the results
Mon Oct  3 06:42:03 2016
```

ballgownオブジェクトを指定してみると

```
> bg
ballgown instance with 44586 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。

サンプル情報 (phenodata.csv)

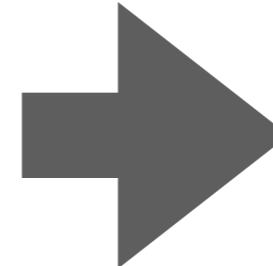
サンプル情報の読み込み

```
> 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)。

Rのデータフレームとして読み込まれる

```
"ids","DN","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"
```



```
> pheno_data
      ids    DN 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
```

サンプル間で遺伝子発現量が変化した遺伝子（DEG）の検出

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", genomesubset=TRUE)
> bg_filtered
ballgown instance with 21190 transcripts and 8 samples
```

- ・ 解析対象が21,190 transcriptsに絞り込まれている。

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="DN", getFC=TRUE, meas="FPKM")
```

- ・ covariateで比較対象の列名、measで利用する発現量の指標、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。

サンプル間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 0.8970859 0.4211202313 0.6233697
2 transcript 6 0.8805537 0.2824973863 0.5005954
3 transcript 7 0.9743486 0.7936585074 0.8856440
4 transcript 9 0.4147634 0.1254622163 0.3237193
5 transcript 10 0.9033641 0.2940974411 0.5127468
6 transcript 11 1.3815087 0.0005690435 0.0338399
```

- 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なうとよい。

サンプル間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の閾値で切った遺伝子を抽出する。

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

	geneNames	geneIDs	transcriptIDs	feature	id	fc	pval	qval
1	- Os02g0623932	Os02t0623932-00	transcript	9347	0.1317284	3.432923e-07	0.004784124	
2	LHY	Os04g0583900	Os04t0583900-01	transcript	19641	204.5558549	5.307114e-07	0.004784124
3	HAG702	Os10g0415900	Os10t0415900-01	transcript	38011	0.5120818	8.358134e-07	0.004784124
4	- Os06g0660800	Os06t0660800-01	transcript	27355	6.4905427	1.068159e-06	0.004784124	
5	- Os08g0360100	Os08t0360100-01	transcript	32883	0.1574917	1.128864e-06	0.004784124	
6	PsbP	Os03g0279950	Os03t0279950-01	transcript	13109	24.3057253	1.853363e-06	0.006545459
7	DTH2	Os02g0724000	Os02t0724000-01	transcript	10165	26.9011427	2.219905e-06	0.006719969
8	- Os01g0227500	Os01t0227500-01	transcript	1172	0.2241632	3.253153e-06	0.008616789	

遺伝子発現量の分布の可視化

DayとNightサンプルをプロットする際に用いる色を指定する。

```
> mycolor=c("orange", "midnightblue")
> palette(mycolor)
```

全遺伝子のFPKM値を抽出する

```
> fpkm <- log2(expr(bg, meas="FPKM") + 1)
```

変数 fpkm の中を見てみると、サンプルごとに全FPKM値が格納されている。

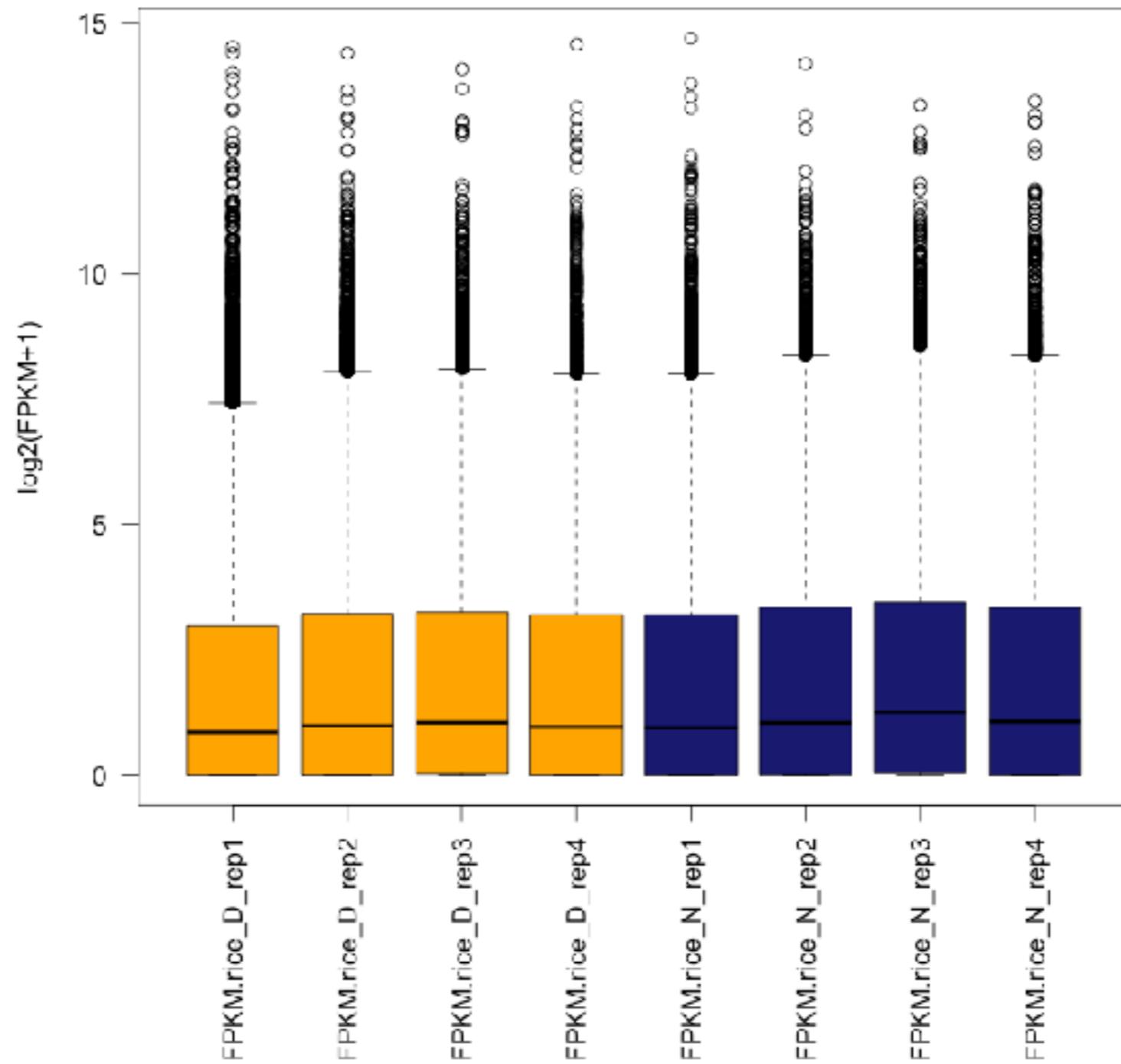
```
> head(fpkm)
   FPKM.rice_D_rep1 FPKM.rice_D_rep2 FPKM.rice_D_rep3 FPKM.rice_D_rep4 FPKM.rice_N_rep1
1  3.65146828      3.928447      4.156322      4.24507492      3.97624072
2  0.33459345      0.000000      0.000000      0.03923387      0.51078575
3  0.05678327      0.000000      0.000000      0.00000000      0.08605001
4  1.67573420      1.616305      1.325968      0.78915331      1.85274197
5  0.00000000      0.000000      0.000000      0.00000000      0.00000000
6  3.75818281      3.817220      4.126931      3.98753462      3.66980756
   FPKM.rice_N_rep2 FPKM.rice_N_rep3 FPKM.rice_N_rep4
1  4.2783960       4.380123      3.817899
2  0.3145248       0.245408      0.000000
3  0.0000000       0.000000      0.000000
4  1.1225521       1.940135      1.338877
5  0.0000000       0.000000      0.000000
6  3.8808186       4.393257      4.165557
```

遺伝子発現量の分布の可視化

グラフの余白を設定し、箱ひげ図を描く。

```
> par(mar=c(8,4,1,1))
> boxplot(fpkm, col=as.numeric(pheno_data$DN), las=2, ylab="log2(FPKM+1)")
```

*boxplot関数はRの標準的な関数



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)
```

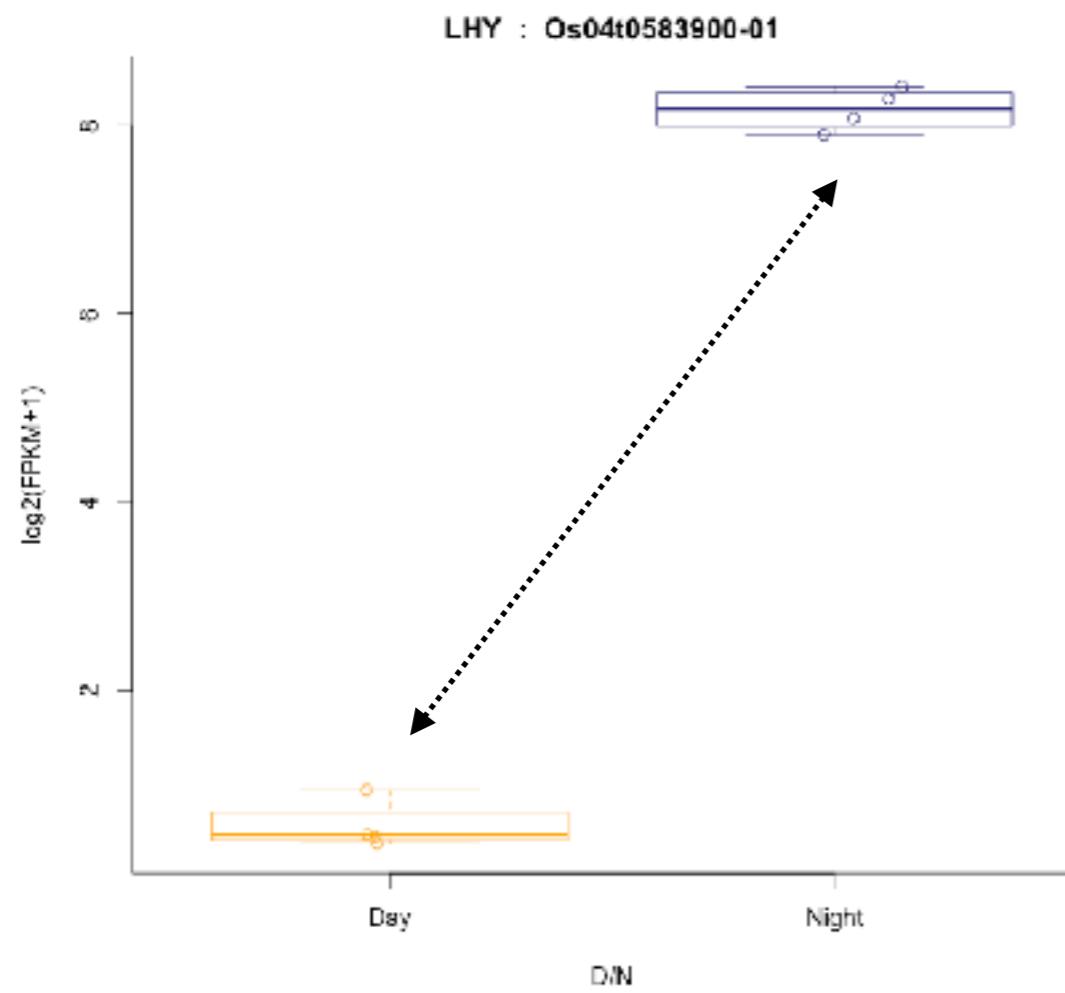
	geneNames	geneIDs	transcriptIDs	feature	id	fc	pval	qval
1	-	Os02g0623932	Os02t0623932-00	transcript	9347	0.1317284	3.432923e-07	0.004784124
2	LHY	Os04g0583900	Os04t0583900-01	transcript	19641	204.5558549	5.307114e-07	0.004784124
3	HAG702	Os10g0415900	Os10t0415900-01	transcript	38011	0.5120818	8.358134e-07	0.004784124
4	-	Os06g0660800	Os06t0660800-01	transcript	27355	6.4905427	1.068159e-06	0.004784124
5	-	Os08g0360100	Os08t0360100-01	transcript	32883	0.1574917	1.128864e-06	0.004784124
6	PsbP	Os03g0279950	Os03t0279950-01	transcript	13109	24.3057253	1.853363e-06	0.006545459
7	DTH2	Os02g0724000	Os02t0724000-01	transcript	10165	26.9011427	2.219905e-06	0.006719969
8	-	Os01g0227500	Os01t0227500-01	transcript	1172	0.2241632	3.253153e-06	0.008616789

- ballgown id: 19641 = LHY = Os04t0583900-01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。

```
> par(mar=c(4, 4, 2, 1)) ← 余白の設定
> plot(fpkm[19641, ] ~ pheno_data$DN, border=c(1, 2),
main=paste(ballgown::geneNames(bg)[19641], ' : ', ballgown::transcriptNames(bg)
[19641]), pch=19, xlab="D/N", ylab='log2(FPKM+1)')
> points(fpkm[19641, ] ~ jitter(as.numeric(pheno_data$DN)),
col=as.numeric(pheno_data$DN))
```

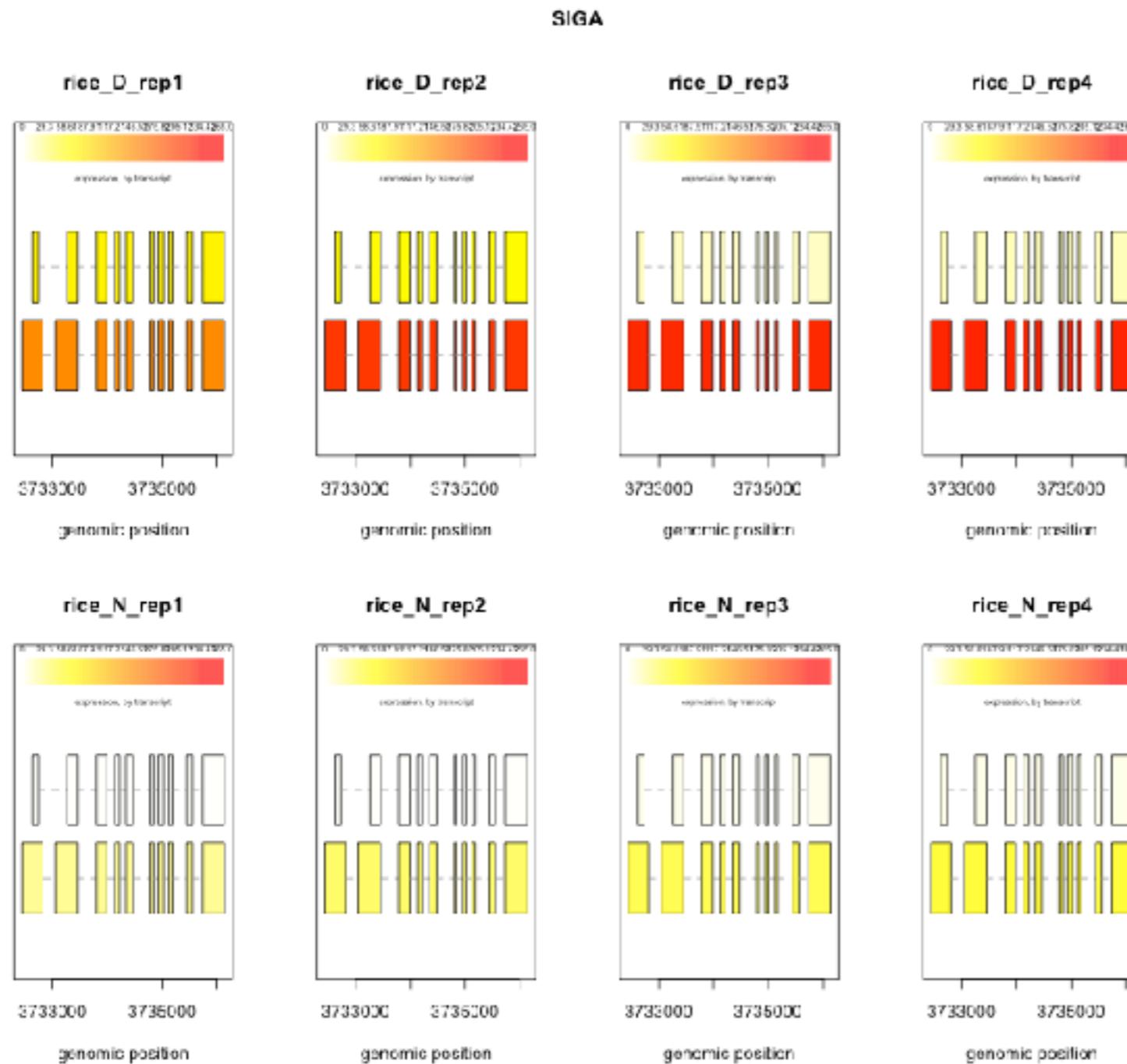


イネの時計遺伝子LHYの発現量は昼は低く、夜は高いパターンを示し、その差は有意である (q-value = 0.004784124)。

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'))
```



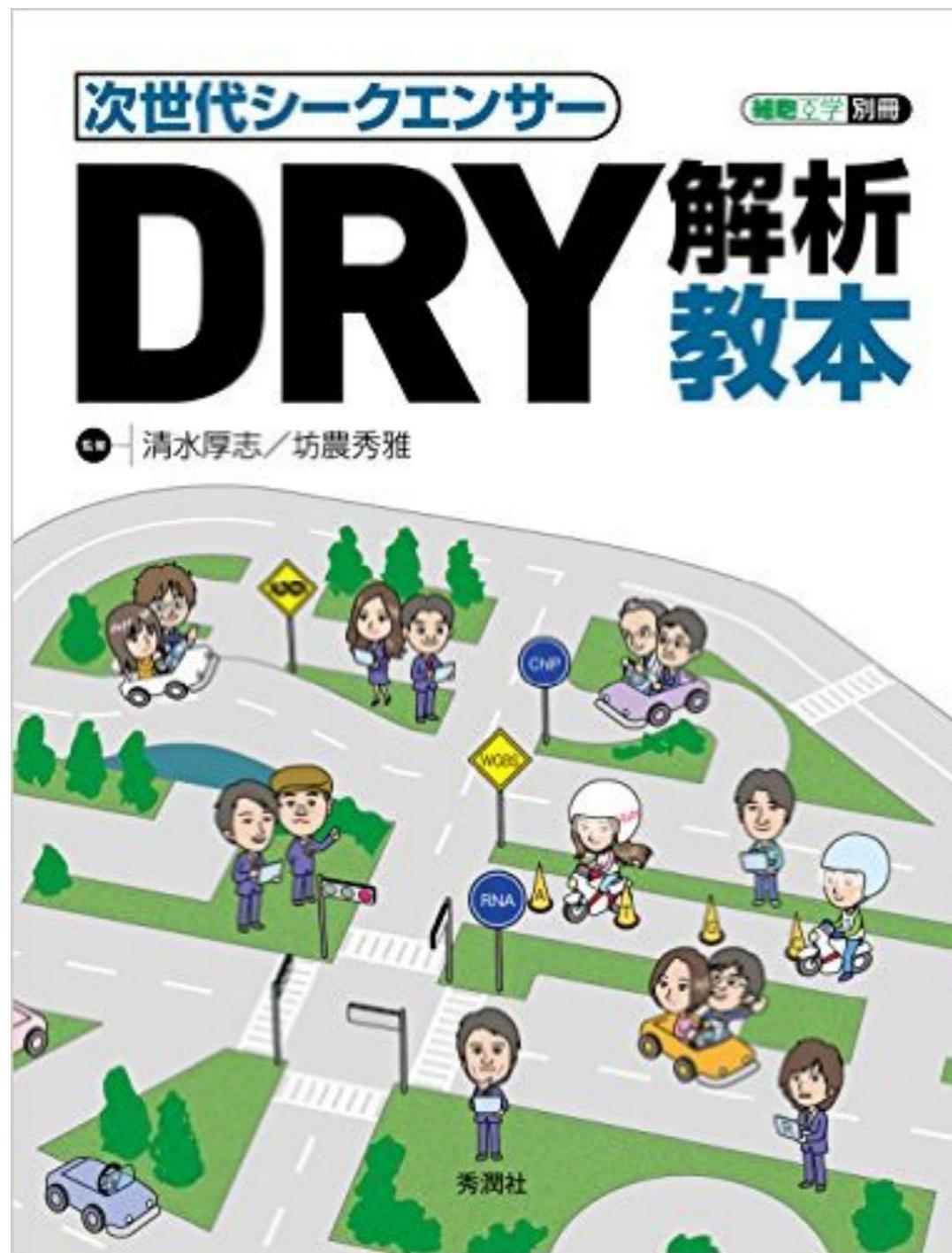
転写調節に関わるイネのsigA遺伝子の発現量は昼に高く、夜に低いパターンを示し、2つあるsplicing isoformうちの一方が主に発現していることが分かる。

参考書籍

次世代シークエンサーDRY解析教本 (細胞工学別冊)

(監修) 清水 厚志, 坊農 秀雅

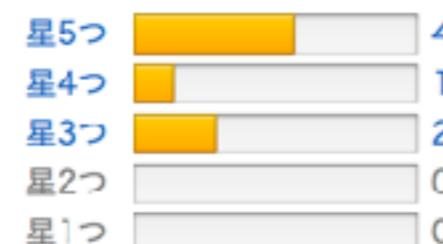
発売日: 2015/10/15



- NGSデータの解析環境の構築から実行コマンドまで非常に細かく書かれている。

カスタマーレビュー

★★★★★ 7
5つ星のうち4.3



★★★★★ 役立ってます

投稿者 Amazon カスタマー 2017年6月14日

★★★★★ 非常にいい

投稿者 kz 2017年4月2日

★★★★★ マストアイテム

投稿者 MOAOL 2015年12月7日

★★★★☆ 良い意味で泥臭い本

投稿者 yasai 2016年3月1日

★★★★★ 現時点におけるNGS解析最強の入門本

投稿者 DRY初心者 2016年2月11日

★★★★☆ 初心者には良い参考書ですが、動かないコードがあります。

投稿者 Amazon カスタマー 2015年12月9日

★★★★☆ 改訂が必要

投稿者 扇一 2016年7月29日

参考書籍

Dr. Bonoの生命科学データ解析 発売日： 2017/9/29
坊農秀雅 (著)



- バイオインフォマティクス全般についてかかれた入門書
- 解析でよく使われるデータベース、ツール、ファイル形式、専門用語などについて書かれている。

参考書籍

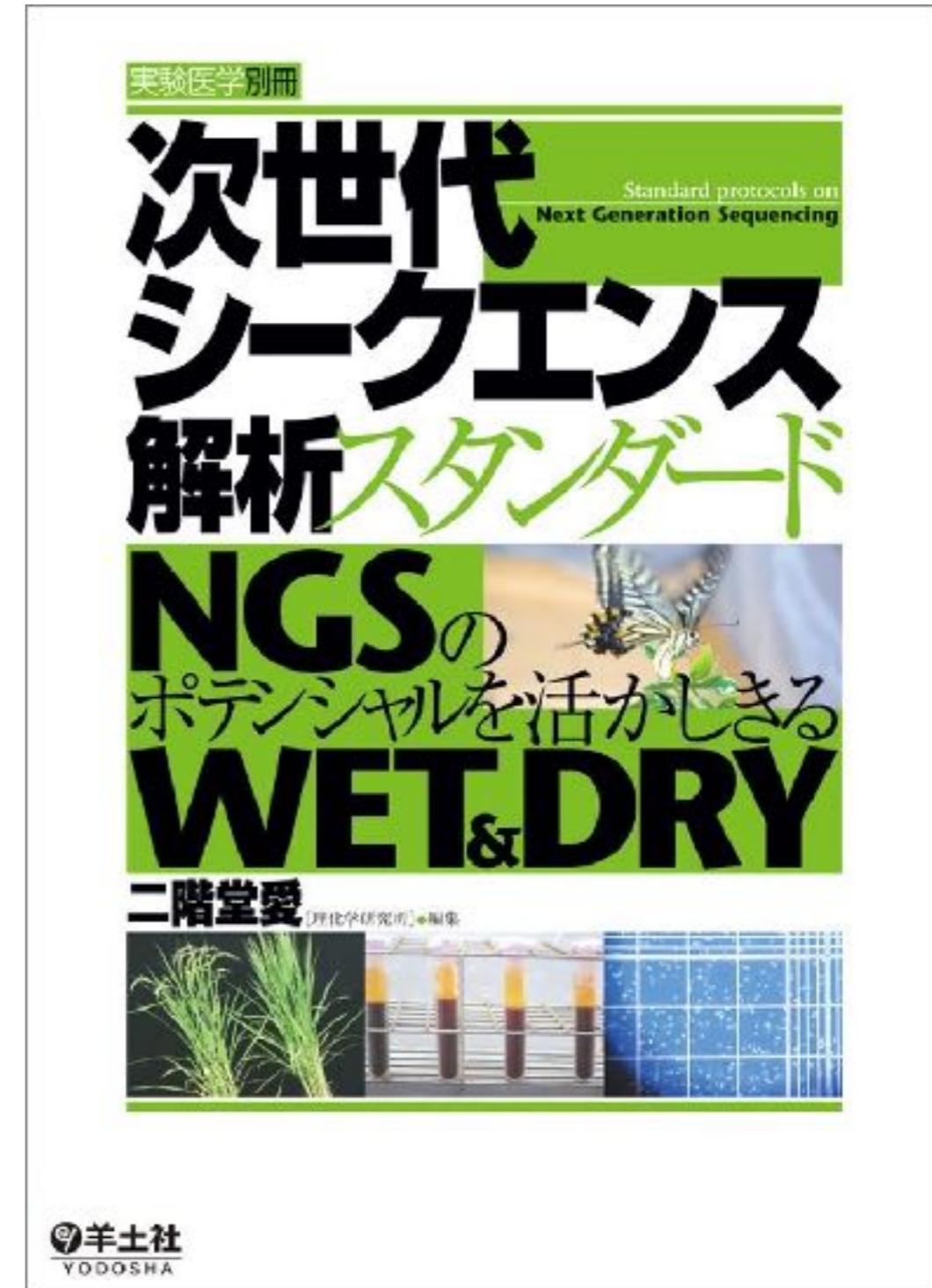
NGSアプリケーション RNA-Seq実験ハンドブック～発現解析からncRNA、シングルセルまであらゆる局面を網羅! (実験医学別冊)

鈴木 穂 (編集) 発売日： 2016/3/28



次世代シークエンス解析スタンダード～NGSのポテンシャルを活かしきる WET&DRY (実験医学別冊)

二階堂 愛 (編集) 発売日： 2014/8/23



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。

まずはGoogleで検索！

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>