



第3部:NGS解析(中～上級)

～クラウド環境との連携、ロングリードデータの解析～

東京大学・大学院農学生命科学研究科
アグリバイオインフォマティクス教育研究プログラム

門田幸二(かどた こうじ)

kadota@iu.a.u-tokyo.ac.jp

<http://www.iu.a.u-tokyo.ac.jp/~kadota/>



利用プログラムの簡単な解説

- DDBJ Pipeline (Nagasaki et al., *DNA Res.*, 2013)
 - マッピングや*de novo*アセンブリを行ってくれるウェブツール(クラウド解析環境)
- FaQCs (Lo and Chain, *BMC Bioinformatics*, 2014)
 - Quality Control用プログラム。クオリティフィルタリングやアダプター除去が主目的
- FastQC
 - Quality Control用プログラム。アダプターの混入などNGSデータのクオリティチェックが主目的
- HGAP (Chin et al., *Nat Methods*, 2013)
 - PacBio用*de novo*ゲノムアセンブリ。.bax.h5ファイルのみを入力として受け付ける
- KmerGenie (Chikhi and Medvedev, *Bioinformatics*, 2014)
 - *de novo*ゲノムアセンブリ時に用いる最適なk値を算出してくれる。推定ゲノムサイズも返す
- Platanus (Kajitani et al., *Genome Res.*, 2014)
 - *de novo*ゲノムアセンブリ。複数のk値を利用するので、KmerGenieとは無関係
- SRA Toolkit
 - sra形式ファイル処理用のプログラム群。FASTQに変換するfastq-dump利用が主目的
- Velvet (Zerbino and Birney, *Genome Res.*, 2008)
 - *de novo*ゲノムアセンブリ。単一のk値を利用するので、KmerGenieと関係あり

おさらい

http://www.ncbi.nlm.nih.gov/pubmed/25879859 Complete genome sequ... Sign in to NCBI

NCBI Resources How To Search Help

PubMed Advanced

Abstract Send to:

BMC Genomics. 2015 Mar 25;16:240. doi: 10.1186/s12864-015-1435-2.

Complete genome sequence and analysis of *Lactobacillus hokkaidonensis* LOOC260(T), a psychrotrophic lactic acid bacterium isolated from silage.

Tanizawa Y^{1,2}, Tohno M³, Kaminuma E⁴, Nakamura Y⁵, Arita M^{6,7}.

Author information

Abstract

BACKGROUND: *Lactobacillus hokkaidonensis* is an obligate heterofermentative lactic acid bacterium, which is isolated from Timothy grass silage in Hokkaido, a subarctic region of Japan. This bacterium is expected to be useful as a silage starter culture in cold regions because of its remarkable psychrotolerance; it can grow at temperatures as low as 4°C. To elucidate its genetic background, particularly in relation to the source of psychrotolerance, we constructed the complete genome sequence of *L. hokkaidonensis* LOOC260(T) using PacBio single-molecule real-time sequencing technology.

RESULTS: The genome (1) comprises one circular chromosome (2.28 Mbp) and two circular plasmids: pLOOC260-1 (81.6 kbp) and pLOOC260-2 (41.0 kbp). We identified diverse mobile genetic elements, such as prophages, integrated and conjugative elements, and conjugative plasmids, which may reflect adaptation to plant-associated niches. Comparative genome analysis also detected unique genomic features, such as genes involved in pentose assimilation and NADPH generation.

CONCLUSIONS: This is the first complete genome in the *L. vaccinostercus* group, which is poorly characterized, so the genomic information obtained in this study provides insight into the genetics and evolution of this group. We also found several factors that may contribute to the ability of *L. hokkaidonensis* to grow at cold temperatures. The results of this study will facilitate further investigation for the cold-tolerance mechanism of *L. hokkaidonensis*.

Full text links
Read free full text at BioMed Central
PMC Full text FREE

Save items Add to Favorites

Similar articles

Lactobacillus hokkaidonensis sp. nc [Int J Syst Evol Microbiol. 2013]
Insights into the completely annotated ger [J Biotechnol. 2012]
Lactobacillus wasatchensis sp. nc [Int J Syst Evol Microbiol. 2015]
Review Genomic organization of [Antonie Van Leeuwenhoek. 1996]
Review Low-redundancy [Antonie Van Leeuwenhoek. 1999]

See reviews... See all...

Related information

1 2

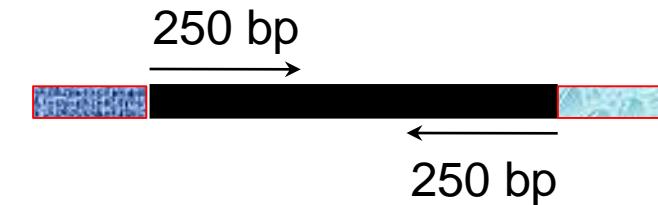
おさらい

PacBio RS IIデータ(DRR024500)

- DRR024500は登録内容に問題があったことが判明し消滅
- 4セル分のデータ。DRR054113–054116に差し替えられている
- セルあたり約15万リード。4セル分なので約60万リード

Illumina MiSeqデータ(DRR024501)

- paired-endゲノムデータ
- リード長は、forward側とreverse側共に250 bp
- オリジナルは2,971,310リード。最初の300,000リードを解析
- forward側(DRR024501sub_1.fastq.gz)
- reverse側(DRR024501sub_2.fastq.gz)



FaQCs実行結果(第6回W5–4)

- 300,000リード → 297,633リード (W5–2)
- forward側(QC.1.trimmed.fastq.gz)
- reverse側(QC.2.trimmed.fastq.gz)
- ファイルの場所: ~/Documents/DRR024501/result



①乳酸菌ゲノム配列決定論文は、2種類のNGS機器から得られたデータを併用している。第6回はIllumina MiSeqデータ(DRR024501)を、そして第7回はPacBioデータを取り扱っている

第6回原稿PDFのp45

尚、このデータの正解は、配列数が3 (1 chromosome + 2 plasmids)、2,400,586 bp (約 2.4MB) である⁴⁾。k値の選択の重要性がよくわかる例といえよう。

通常、Velvetを実行する場合は複数の異なるk値を用いてアセンブルを行い、それらの結果を眺める[W10]。ここでは、計10個のk値 (k=31, 61, 91, 111, 121, 131, 151, 171, 181, 191) で実行した結果を眺め、主に配列数の観点から、k=171周辺の結果が一番よさそうだと解釈する。もちろんこのデータの場合は、「真のゲノムサイズは約2.4MB」だという答えがわかった状態でアセンブリ結果の評価を行っていることになるが、実際には近縁種との比較により妥当と考えられるゲノムサイズを検討する。ここではそのような情報が得られなかつたと仮定して「ゲノムサイズ推定」を行い、アセンブリ結果の評価を行う。

ゲノムサイズ推定



ゲノムサイズの推定は、フローサイトメトリー (flow cytometry) という手法を用いて実験的に求めるやり方

複数のk値でVelvetアセンブリを行い、主観でk=171がいいと判断したところまでが2016.08.02の内容。今日は、Velvetなど単一のk値を指定してアセンブリを実行する際に、客観的に最もよいと思われるk値を出力してくれるKmerGenieのインストールの話からスタート。KmerGenieは、makeコマンドを利用してインストールする2例目として、また最適k値と同時に出力する①ゲノムサイズ推定周辺を主目的として紹介。KmerGenie出力結果の一部には、2016.07.20で眺めたk-mer出現頻度分布もある。②はゲノムサイズ推定の意義について…



Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



ゲノムサイズ推定

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランскриプトーム、正規化、発現変動、統計、モラフニン

(last modified 2016/06/03, since 2011)

- ・書籍 | 日本乳酸菌学会誌 | 第3回Linux環境構築からNGSデータ取得まで (last modified 2015/12/15)
- ・書籍 | 日本乳酸菌学会誌 | 第4回クオリティコントロールとプログラムのインストール (last modified 2016/06/13)
- ・書籍 | 日本乳酸菌学会誌 | 第5回アセンブル、マッピング、そしてQC (last modified 2016/06/13)
- ・書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ (last modified 2016/06/15) NEW
- ・書籍 | 日本乳酸菌学会誌 | 第7回ロングリードアセンブリ (last modified 2016/05/12)
- ・イントロ
- ・イントロ
- ・イントロ
- ・イントロ

書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ NEW

日本乳酸菌学会誌の第6回分です。Linuxコマンドのリンク先は主に日経BP社様です。原稿PDFの「はじめに」項目のところにtypoがあります。誤:するであれば、正:する"の"であれば、ですねm()m。また、「配列長によるフィルタリング」項目の最後の文章「これらについては、第7回で詳述する予定である。」についてですが、これは「これらについては、"第8回以降"で詳述する予定である。」と読み替えてください。第7回ドラフト原稿作成時点。

加)。

- ・原稿PDF
- ・ウェブ資料PDF
 - Windows用(2016.03.29版)
 - Macintosh用
- ・(共有フォルダ設定情報を含む)追
 - Windows用(2015.12.28版)
 - Macintosh用(2015.12.28版)



ゲノムサイズ推定

- ・ゲノムサイズの調べ方(OKWAVE)
 - Jerryfish: Marçais and Kingsford, Bioinformatics, 2011
 - KmerGenie: Chikh and Medvedev, Bioinformatics, 2014
 - KmerStream: Melsted and Halldórsson, Bioinformatics, 2014
- ・KmerGenieダウンロードと解凍[W11-2]

```
cd ~/Downloads
ls
wget -cq http://kmergenie.bx.psu.edu/kmergenie-1.6982.tar.gz
#cp ~/Desktop/backup/kmergenie-1.6982.tar.gz .
ls -l kmer*
tar -zvxf kmergenie-1.6982.tar.gz
ls -ld kmer*
```

W11-1 : KmerGenie

Software: KmerGenie

KmerGenie estimates the best k-mer length for genome de novo assembly. Given a set of reads, KmerGenie first computes the k-mer abundance histogram for many values of k. Then, for each value of k, it predicts the number of distinct genomic k-mers in the dataset, and returns the k-mer length which maximizes this number. Experiments show that KmerGenie's choices lead to assemblies that are close to the best possible over all k-mer lengths.

KmerGenie predictions can be applied to single-k genome assemblers (e.g. Velvet, SOAPdenovo 2, ABySS, Minia). However, multi-k genome assemblers (e.g. SPAdes, IDBA) generally perform better with default parameters (using multiple k values), rather than the single best k predicted by KmerGenie.

See a [sample report](#) generated by KmerGenie from a dataset of bacterial reads.

Download



Download KmerGenie sources here: [kmergenie-1.6982.tar.gz](#)

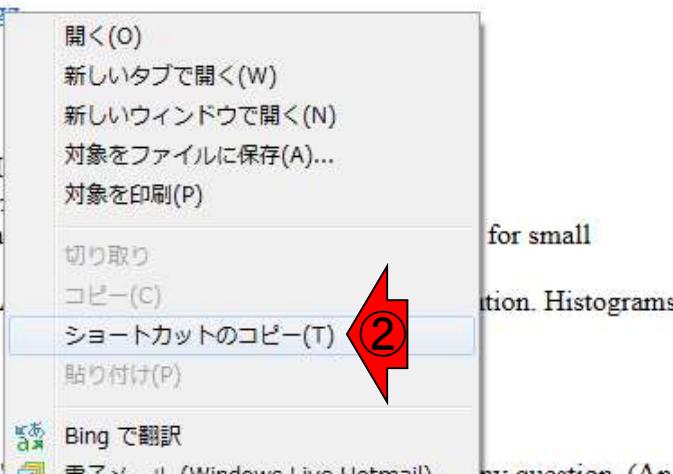
You will need Python and R.

Latest [README](#) and [CHANGELOG](#). Major changes since

- 1.6213 (1/30/14): Advanced Help section in the HTML
- 1.5621 (8/02/13): HTML report for easier results examination. Histograms are automatically plotted.
- 1.5378 (6/25/13): Suitable histogram resolution (-e parameter) for small (bacterial) genomes.
- 1.5260 (5/26/13): Improved model R code (thanks to ... are automatically plotted).

Support

Please use Biostars (dynamic FAQ system) click "New Post"



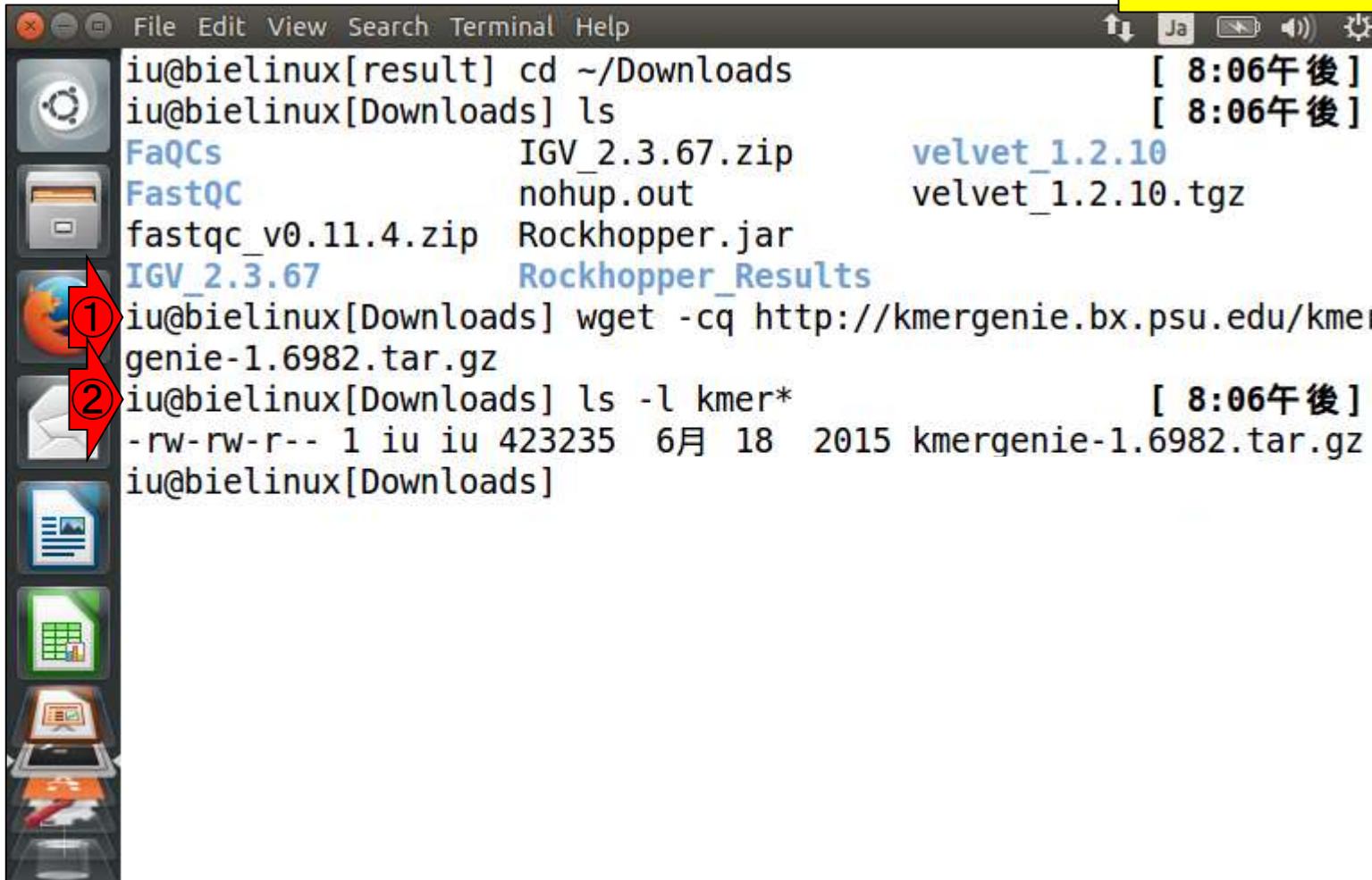
2016年6月15日現在のKmerGenieのバージョンは1.7016だが、第6回ウェブ資料作成当時(2016年1月5日)のKmerGenie (ver. 1.6982)をダウンロードできるのでそれで説明する。

①で右クリックし、②ショートカットのコピーで、wgetでダウンロードするときに必要なURL情報を取得できる。講習会ではver. 1.6982のtar.gzファイルをダウンロード済みですが、サイズは約400KBなので、時間をずらして休憩時間に最新版をダウンロードしてみてもよい

W11-2:ダウンロードと解凍

①ダウンロードと解凍。講習会ではver. 1.6982のtar.gzファイルをダウンロード済み。実際に行うのは、②の確認のみ

```
File Edit View Search Terminal Help [ 8:06午後 ]  
iu@bielinux[ result ] cd ~/Downloads  
iu@bielinux[ Downloads ] ls [ 8:06午後 ]  
FaQCs IGV_2.3.67.zip velvet_1.2.10  
FastQC nohup.out velvet_1.2.10.tgz  
fastqc_v0.11.4.zip Rockhopper.jar  
IGV_2.3.67 Rockhopper_Results  
iu@bielinux[ Downloads ] wget -cq http://kmergenie.bx.psu.edu/kmergenie-1.6982.tar.gz [ 8:06午後 ]  
iu@bielinux[ Downloads ] ls -l kmer*  
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz  
iu@bielinux[ Downloads ]
```



ゲノムサイズ推定

もし、.tar.gzファイルを間違って消してしまったり、その後のwgetでしくじった場合は、①でリカバリしてください

ゲノムサイズ推定

- ・[ゲノムサイズの調べ方\(OKWAVE\)](#)
- ・[Jerryfish: Marçais and Kingsford, Bioinformatics, 2011](#)
- ・[KmerGenie: Chikh and Medvedev, Bioinformatics, 2014](#)
- ・[KmerStream: Melsted and Halldórsson, Bioinformatics, 2014](#)
- ・[KmerGenie](#)ダウンロードと解凍[W11-2]

```
cd ~/Downloads  
ls  
wget -cq http://kmergenie.bx.psu.edu/kmergenie-1.6982.tar.gz  
#cp ~/Desktop/backup/kmergenie-1.6982.tar.gz .  
ls -l kmer*  
tar -zxvf kmergenie-1.6982.tar.gz  
ls -ld kmer*
```



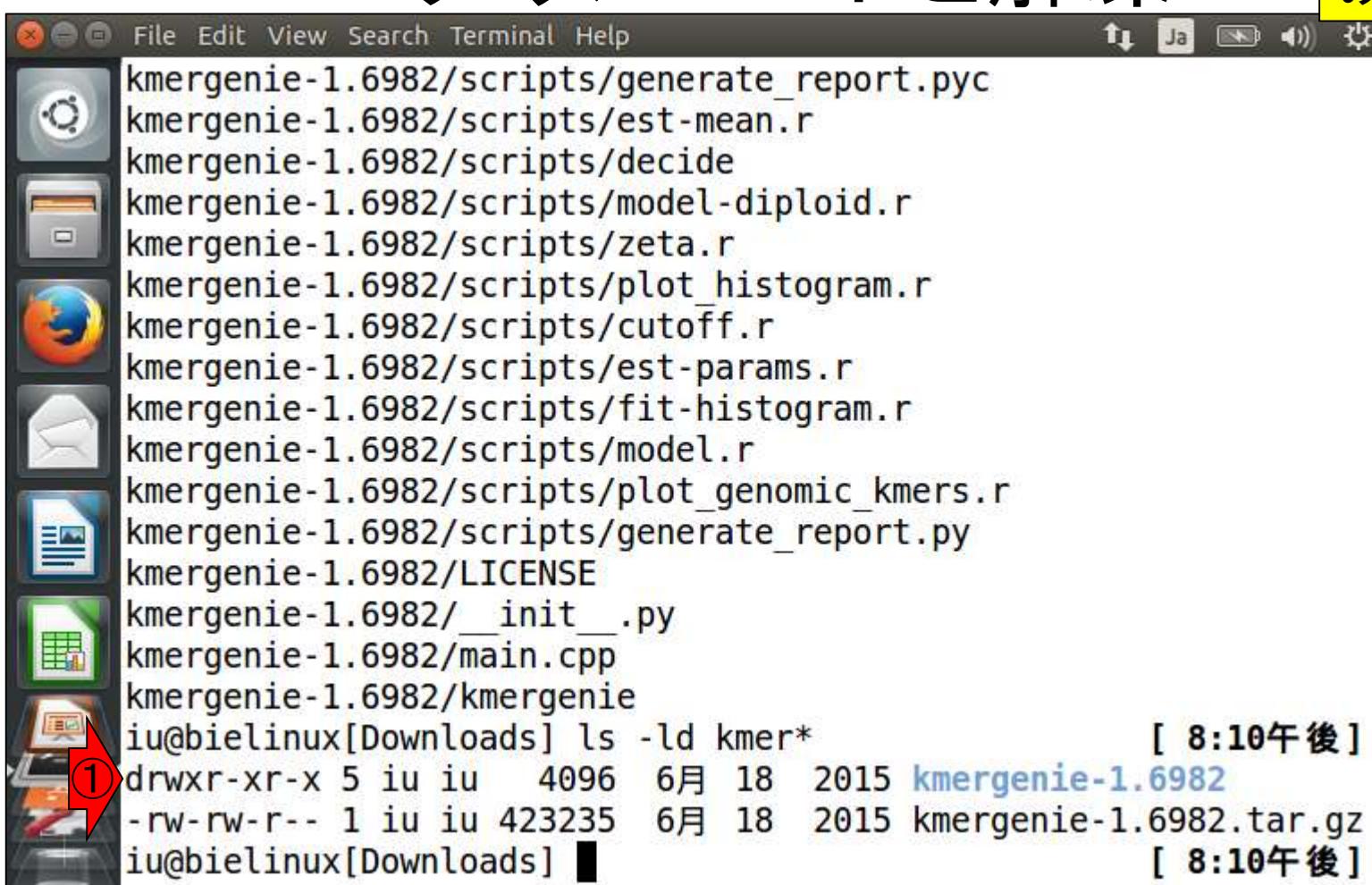
W11-2:ダウンロードと解凍

①解凍。.tar.gzファイルなので、解凍コマンドはW9-3と同じ。
ここから手を動かす

```
File Edit View Search Terminal Help [ 3:54 午後 ]  
iu@bielinux[Downloads] cd ~/Downloads  
iu@bielinux[Downloads] ls [ 3:55 午後 ]  
boost_1_61_0.tar.bz2 master.zip  
Bridger_r2014-12-01.tar.gz nohup.out  
FAQCs Rockhopper.jar  
FastQC Rockhopper_Results  
fastqc_v0.11.4.zip sratoolkit.2.6.3-ubuntu64.tar.gz  
IGV_2.3.67 v2.2.0.tar.gz  
IGV_2.3.67.zip velvet_1.2.10  
kmergenie-1.6982.tar.gz velvet_1.2.10.tgz  
iu@bielinux[Downloads] ls -l kmer* [ 3:55 午後 ]  
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz  
iu@bielinux[Downloads] tar -zxvf kmergenie-1.6982.tar.gz
```

W11-2:ダウンロードと解凍

①解凍が無事終わると、
.tar.gzを除いた部分の名前の
ディレクトリが作成される



The screenshot shows a Linux desktop environment with a terminal window open. The terminal displays the contents of a directory named 'kmergenie-1.6982'. The files listed include various scripts in Python (.py), R (.r), and C/C++ (.cpp), along with a LICENSE file. A red arrow points to the number '1' in the bottom left corner of the terminal window, indicating the step of extracting the tar.gz file.

```
kmergenie-1.6982/scripts/generate_report.pyc
kmergenie-1.6982/scripts/est-mean.r
kmergenie-1.6982/scripts/decide
kmergenie-1.6982/scripts/model-diploid.r
kmergenie-1.6982/scripts/zeta.r
kmergenie-1.6982/scripts/plot_histogram.r
kmergenie-1.6982/scripts/cutoff.r
kmergenie-1.6982/scripts/est-params.r
kmergenie-1.6982/scripts/fit-histogram.r
kmergenie-1.6982/scripts/model.r
kmergenie-1.6982/scripts/plot_genomic_kmers.r
kmergenie-1.6982/scripts/generate_report.py
kmergenie-1.6982/LICENSE
kmergenie-1.6982/_init_.py
kmergenie-1.6982/main.cpp
kmergenie-1.6982/kmergenie
iu@bielinux[Downloads] ls -ld kmer*
drwxr-xr-x 5 iu iu 4096 6月 18 2015 kmergenie-1.6982
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz
iu@bielinux[Downloads] [ 8:10 午後 ]
```

W11-3: README

```
File Edit View Search Terminal Help  
iu@bielinux[Downloads] pwd [ 8:23 午後 ]  
/home/iu/Downloads  
iu@bielinux[Downloads] ls -ld kmer* [ 8:24 午後 ]  
drwxr-xr-x 5 iu iu 4096 6月 18 2015 kmergenie-1.6982  
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz  
iu@bielinux[Downloads] cd kmergenie-1.6982 [ 8:24 午後 ]  
iu@bielinux[kmergenie-1.6982] pwd [ 8:24 午後 ]  
/home/iu/Downloads/kmergenie-1.6982  
iu@bielinux[kmergenie-1.6982] ls [ 8:24 午後 ]  
CHANGELOG LICENSE minia third_party  
__init__.py main.cpp README wrapper.py  
kmergenie makefile scripts  
iu@bielinux[kmergenie-1.6982] less README [ 8:24 午後 ]
```

基本的には、①の場所でmakeと打てばいいが、今一度マニュアル(README)を②lessで眺める。lessの利用法は、第3回W14-6。(スペースキーを押してページスクロール、qで抜ける)



①

②

W11-3: README

```
File Edit View Search Terminal Help
KmerGenie
-----
installation:

 1) Prerequisite: Python >= 2.5 and RScript (included in R).
 2) Type `make` in the KmerGenie directory
 3) Optionally, `make install` will create a symbolic link of
    kmergenie to /usr/local/bin (yet, do not delete the original in
    stall folder)

usage:
./kmergenie reads_file
 ①
reads_file is either a FASTA, FASTQ, FASTA.gz, FASTQ.gz file or
a list of file names, one per line.
 ②
README
```

W11-3: README

File Edit View Search Terminal Help
mmary of the results, and contains an Advanced help analysis.

前のスライドからスペースキーを2回押したあたりの画面。①の赤枠部分に注目。デフォルトでは、KmerGenieは②k=121までしか探索しない。探索の上限を200まで引き上げたい場合は、「make clean」と打ったのち、「make k=200」と打て、と書いてある。③qと打ってlessを終了

* There is no need to use Kmergenie for a multi-k assembler, like SPAdes. Default parameters of multi-k assemblers are generally better than a single best k.

* To support even larger values of k, e.g. up to 200, type `make clean ; make k=200`. By default, Kmergenie is compiled to support values of k up to 121.

* By default, KmerGenie will perform another pass to estimate k more precisely. You may skip it by using the "--one-pass" option (roughly 2x faster).

* To run multiple instances of KmerGenie on the same folder, specify the "-o" and "-t" parameters (output prefix, number of threads per instance).

: █ ③

W11-4:make clean

```
iu@bielinux[Downloads] pwd [ 8:23午後]
/home/iu/Downloads
iu@bielinux[Downloads] ls -ld kmer*
drwxr-xr-x 5 iu iu 4096 6月 18 2015 kmergenie-1.6982
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz
iu@bielinux[Downloads] cd kmergenie-1.6982 [ 8:24午後]
iu@bielinux[kmergenie-1.6982] pwd [ 8:24午後]
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls [ 8:24午後]
CHANGELOG LICENSE minia third_party
__init__.py main.cpp README wrapper.py
kmergenie makefile scripts
iu@bielinux[kmergenie-1.6982] less README [ 8:24午後]
iu@bielinux[kmergenie-1.6982] make clean [ 8:38午後]
rm -rf *.o minia/*.o
iu@bielinux[kmergenie-1.6982] ls [ 8:38午後]
CHANGELOG LICENSE minia third_party
__init__.py main.cpp README wrapper.py
kmergenie makefile scripts
iu@bielinux[kmergenie-1.6982] [ 8:39午後]
```

①

W11-5:make k=200

```
File Edit View Search Terminal Help Ja [ 8:23午後 ]
iu@bielinux[Downloads] pwd
/home/iu/Downloads
iu@bielinux[Downloads] ls -ld kmer*
drwxr-xr-x 5 iu iu 4096 6月 18 2015 kmergenie-1.6982
-rw-rw-r-- 1 iu iu 423235 6月 18 2015 kmergenie-1.6982.tar.gz
iu@bielinux[Downloads] cd kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] pwd
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG LICENSE minia third_party
__init__.py main.cpp README wrapper.py
kmergenie makefile scripts
iu@bielinux[kmergenie-1.6982] less README
iu@bielinux[kmergenie-1.6982] make clean
rm -rf *.o minia/*.o
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG LICENSE minia third_party
__init__.py main.cpp README wrapper.py
kmergenie makefile scripts
iu@bielinux[kmergenie-1.6982] make k=200 [ 8:39午後 ]
```

①

W11-5:make k=200

```
File Edit View Search Terminal Help
geint -DKMER_PRECISION=7
g++ -o specialk minia/Pool.o minia/Bank.o minia/Hash16.o minia/Bloom.o minia/Kmer.o minia/Utils.o minia/LinearCounter.o minia/BigInt.o minia/MultiConsumer.o minia/Hashing.o main.cpp -O4 -pthread
read -D largeint -DKMER_PRECISION=7 -lz -DSVN_REV=1.6982
scripts/test_install
Testing presence of specialk....
OK
Testing presence of Rscript....
R scripting front-end version 3.2.0 (2015-04-16)
OK
Testing basic Rscript functionality....
Rscript --no-init-file -e 'rnorm(1)'
[1] "rnorm(1)"
OK
Testing a simple KmerGenie example....
OK
Test successful. KmerGenie is ready, type `./kmergenie`.
iu@bielinux[kmergenie-1.6982] [ 8:44午後]
```

①

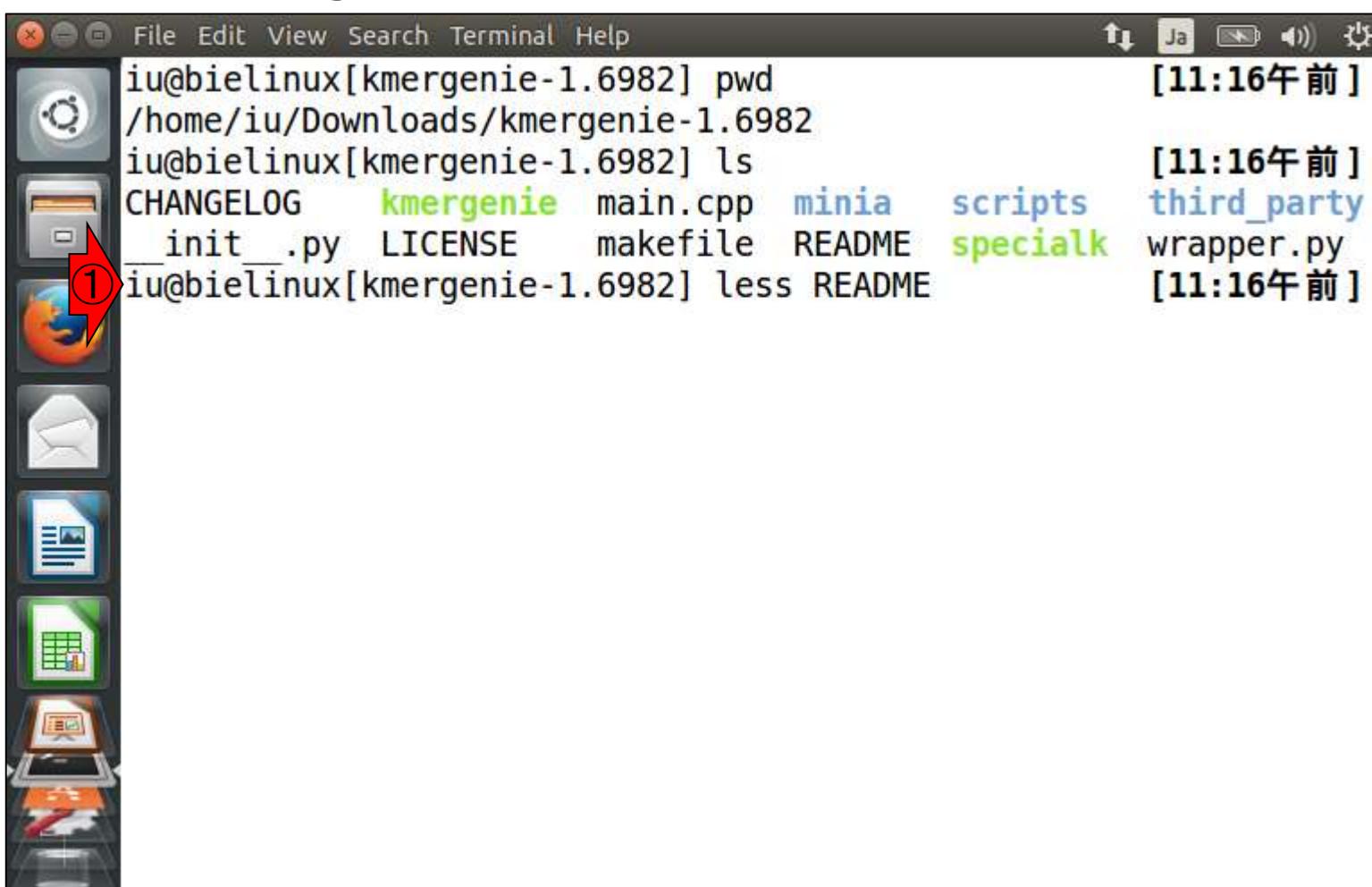
W11-5:make k=200

```
File Edit View Search Terminal Help
OK
Test successful. KmerGenie is ready, type `./kmer
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG  kmergenie  main.cpp  minia  scripts  third_party
 init__.py  LICENSE   makefile   README  specialk wrapper.py
iu@bielinux[kmergenie-1.6982] ls -l
total 200
-rw-r--r-- 1 iu iu 7511 6月 18 2015 CHANGELOG
-rw-r--r-- 1 iu iu 0 6月 18 2015 __init__.py
-rwxr-xr-x 1 iu iu 4401 6月 18 2015 kmergenie
-rw-r--r-- 1 iu iu 22745 6月 18 2015 LICENSE
-rw-r--r-- 1 iu iu 10753 6月 18 2015 main.cpp
-rw-r--r-- 1 iu iu 2432 6月 18 2015 makefile
drwxr-xr-x 2 iu iu 4096 1月 5 20:44 minia
-rw-r--r-- 1 iu iu 3220 6月 18 2015 README
drwxr-xr-x 2 iu iu 4096 6月 18 2015 scripts
-rwxrwxr-x 1 iu iu 122924 1月 5 20:44 specialk
drwxr-xr-x 2 iu iu 4096 6月 18 2015 third_party
-rw-r--r-- 1 iu iu 1945 6月 18 2015 wrapper.py
iu@bielinux[kmergenie-1.6982] [ 8:45午後]
```

2016年1月5日に作業している。①11個だったのが12個に増えている。specialkというものが増えたものようだ。②ls -l。③確かにspecialkという実行ファイルができた。他にも④miniaというディレクトリの内容もどこかが変更されたようだ



W11-6: README



```
File Edit View Search Terminal Help
iu@bielinux[kmergenie-1.6982] pwd [11:16午前]
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls [11:16午前]
CHANGELOG  kmergenie  main.cpp  minia  scripts  third_party
_init_.py  LICENSE   makefile   README  specialk  wrapper.py
iu@bielinux[kmergenie-1.6982] less README [11:16午前]
```

The screenshot shows a terminal window on a Linux desktop environment. The terminal output is as follows:

```
iu@bielinux[kmergenie-1.6982] pwd [11:16午前]
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls [11:16午前]
CHANGELOG  kmergenie  main.cpp  minia  scripts  third_party
_init_.py  LICENSE   makefile   README  specialk  wrapper.py
iu@bielinux[kmergenie-1.6982] less README [11:16午前]
```

A red arrow points to the word "README" in the command "less README". The terminal interface includes a menu bar, a toolbar with icons for file operations, and a vertical dock on the left containing icons for various applications like a web browser, file manager, and terminal.

W11-6: README

```
X - File Edit View Search Terminal Help  
KmerGenie  
-----  
installation:  
  
1) Prerequisite: Python >= 2.5 and RScript (included in R).  
2) Type `make` in the KmerGenie directory  
3) Optionally, `make install` will create a symbolic link of  
kmergenie to /usr/local/bin (yet, do not delete the original in  
stall folder)  
  
usage:  
  
./kmergenie reads_file  
  
reads_file is either a FASTA, FASTQ, FASTA.gz, FASTQ.gz file or  
a list of file names, one per line.  
  
README
```

①これが基本的な使い方。赤下線左端の./を見て、「kmergenieのパスを通さなければ、絶対パスで書かないといけないのか…」と認識する。そして、②赤枠内の Optionally以下の文章に目が留まる。実行プログラム(この場合kmergenie)のシンボリックリンクを /usr/local/binに置けばいいことは第4回でも説明した。第4回W9-5で紹介した「ln -s …」の記述法は結構面倒だが、「make install」と打てばいいらしいと解釈



①

②

W11-7:make install

```
File Edit View Search Terminal Help
iu@bielinux[kmergenie-1.6982] pwd
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG  kmergenie  main.cpp  minia  minia.h  scr  spec
            init_.py  LICENSE  makefile  README  spec.h
iu@bielinux[kmergenie-1.6982] less README
iu@bielinux[kmergenie-1.6982] make install
rm -f /usr/local/bin/kmergenie && ln -s `pwd`/kmergenie /usr/local/bin
ln: failed to create symbolic link '/usr/local/bin/kmergenie': Permission denied
make: *** [install] Error 1
iu@bielinux[kmergenie-1.6982] ls -ld /usr/local/bin [12:09午後]
drwxr-xr-x 2 root root 4096 12月 11 19:36 /usr/local/bin
iu@bielinux[kmergenie-1.6982] whoami [12:09午後]
iu
iu@bielinux[kmergenie-1.6982] █
```

(qと打ってlessから抜けて)①make install。エラーが出て失敗していることがわかる。この理由は、第4回W9-5でも説明しているが /usr/local/binディレクトリの所有者はrootさんで、rootさん以外のヒトは書き込み権限が与えられていない(つまりdrwxr-xr-xとなっており、wがない)。そこに③iuさんが書き込みを行おうとしたからPermission denied(権限がない)と文句を言われたというオチです



W11-7:make install

```
File Edit View Search Terminal Help
iu@bielinux[kmergenie-1.6982] pwd
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG  kmergenie  main.cpp  minia  scr
init_.py  LICENSE  makefile  README  spe
iu@bielinux[kmergenie-1.6982] kmergenie -h
zsh: command not found: kmergenie
iu@bielinux[kmergenie-1.6982] sudo make install      [12:40午後]
[sudo] password for iu:
rm -f /usr/local/bin/kmergenie && ln -s `pwd`/kmergenie /usr/local/bin
Kmergenie is installed, via a symlink to /usr/local/bin (do not
delete the contents of /home/iu/Downloads/kmergenie-1.6982)
iu@bielinux[kmergenie-1.6982]
```

①sudo make install。赤下線のところでパスワードを打ち込むよう促されたらpass1409。エラーが出ずに/usr/local/binにkmergenieの実行ファイルを置けたようだ(無事パスを通せたようだ)。②記述内容を眺めることでmake installの実体は、/usr/local/bin/kmergenieがあつたらそれをrmで消して、ln -sでパスを通す一連のコマンド実行のようだと学習する

W11-7:make install

```
File Edit View Search Terminal Help
iu@bielinux[kmergenie-1.6982] pwd
/home/iu/Downloads/kmergenie-1.6982
iu@bielinux[kmergenie-1.6982] ls
CHANGELOG  kmergenie  main.cpp  minia  scripts  [12:40午後]
           init_.py  LICENSE   makefile  README  specialk  third_party
iu@bielinux[kmergenie-1.6982] kmergenie -h
zsh: command not found: kmergenie
iu@bielinux[kmergenie-1.6982] sudo make install      [12:40午後]
[sudo] password for iu:
rm -f /usr/local/bin/kmergenie && ln -s `pwd`/kmergenie /usr/local/bin
Kmergenie is installed, via a symlink to /usr/local/bin (do not
delete the contents of /home/iu/Downloads/kmergenie-1.6982)
iu@bielinux[kmergenie-1.6982] kmergenie -h          [12:40午後]
KmerGenie

Usage:
  kmergenie <read_file> [options]

Options:
```

①パスを通す前は、kmergenieのヘルプを表示させようとしても②「そんなコマンドはない」と文句を言われるが、③パスを通した後は文句を言われません。./kmergenieとkmergenieの違いが分からないヒトは、第4回W9で復習

W11-7:kmergenie -h

```
File Edit View Search Terminal Help  
iu@bielinux[kmergenie-1.6982] kmergenie -h  
KmerGenie  
  
① Usage:  
    kmergenie <read_file> [options]  
  
Options:  
    --diploid      use the diploid model (default: haploid model)  
    --one-pass     skip the second pass to estimate k at 2 bp resolution (default: two passes)  
    -k <value>    largest k-mer size to consider (default: 121) ②  
    -l <value>    smallest k-mer size to consider (default: 15) ③  
    -s <value>    interval between consecutive kmer sizes (default: 10)  
    -e <value>    k-mer sampling value (default: auto-detected to use ~200 MB memory/thread)  
    -t <value>    number of threads (default: number of cores minus one)  
    -o <prefix>   prefix of the output files (default: histograms)  
iu@bielinux[kmergenie-1.6982]
```

①kmergenie -hの結果を眺める。②乳酸菌(haploid; 一倍体)の場合は気にしなくていいが、ヒトなどの二倍体ゲノムの場合は--diploidオプションをつけないといけないようだ。W10-6でVelvetの結果として得たのはk=31から191であった。この範囲でbest k-merを探索したい場合は、③「-l 31 -k 191」オプションを、④の位置につけて実行すればいいようだ

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6: FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W11-8: 実行

一通りの実行方法がわかったので、とりあえず①W5-4で作成したforward側のファイル(QC.1.trimmed.fastq.gz)のみを入力として31から191のk-merの探索範囲でkmmergenieを実行。約15分

```
File Edit View Search Terminal Help
iu@bielinux[kmmergenie-1.6982] cd ~/Documents/DRR024501/result
iu@bielinux[result] pwd [ 1:40午後]
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls [ 1:40午後]
fastqCount.txt hoge_151 QC.2.trimmed.fastq.gz
hoge_111 hoge_171 QC_qc_report.pdf
hoge_121 hoge_191 QC.stats.txt
hoge_131 QC.1.trimmed.fastq.gz QC.unpaired.trimmed.fastq
iu@bielinux[result] kmmergenie QC.1.trimmed.fastq.gz -l 31 -k 191
```

W11-8:途中経過1

リターンキーを押してから、確かに10秒くらいの状態。
①確かにオプションで指定した通りの範囲を探索してくれているようだ

```
File Edit View Search Terminal Help ↑ ↓ Ja 13:41 ⌂
iu@bielinux[kmergenie-1.6982] cd ~/Documents/DRR024501/result
iu@bielinux[result] pwd [ 1:40午後]
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls [ 1:40午後]
fastqCount.txt  hoge_151      QC.2.trimmed.fastq.gz
hoge_111        hoge_171      QC_qc_report.pdf
hoge_121        hoge_191      QC.stats.txt
hoge_131        QC.1.trimmed.fastq.gz  QC.unpaired.trimmed.fastq
iu@bielinux[result] kmergenie QC.1.trimmed.fastq.gz -l 31 -k 191
running histogram estimation
Linear estimation: ~48 M distinct 106-mers are in the reads
K-mer sampling: 1/14
| processing
[going to estimate histograms for values of k: 191 181 171 161 15
1 141 131 121 111 101 91 81 71 61 51 41 31
----- █
```



W11-8: 無事終了

終了後の状態。KmerGenieプログラムの主な目的は、Velvetなどのアセンブリプログラム実行時に最適だと思われるk-merの推定。forward側のみで実行した結果は、①k=87がお勧めだという結果だった。とりあえず②ls

```
-----Total time Wallclock 447.581
s
fitting model to histograms to estimate best k
estimation of the best k so far: 91
refining estimation around [85; 97], with a step of 2
running histogram estimation
Linear estimation: ~65 M distinct 59-mers are in the reads
K-mer sampling: 1/19
| processing
|
[going to estimate histograms for values of k: 97 95 93 91 89 87
85
-----
-----Total time Wallclock 189.77 s
fitting model to histograms to estimate best k
table of predicted num. of genomic k-mers: histograms.dat
recommended coverage cut-off for best k: 1
best k: 87
iu@bielinux[result] ls
```

[1:53午後]

①

②

W11-9：確認

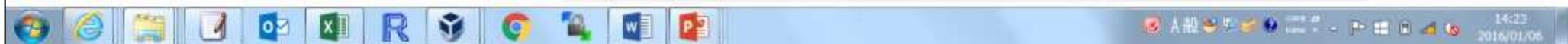
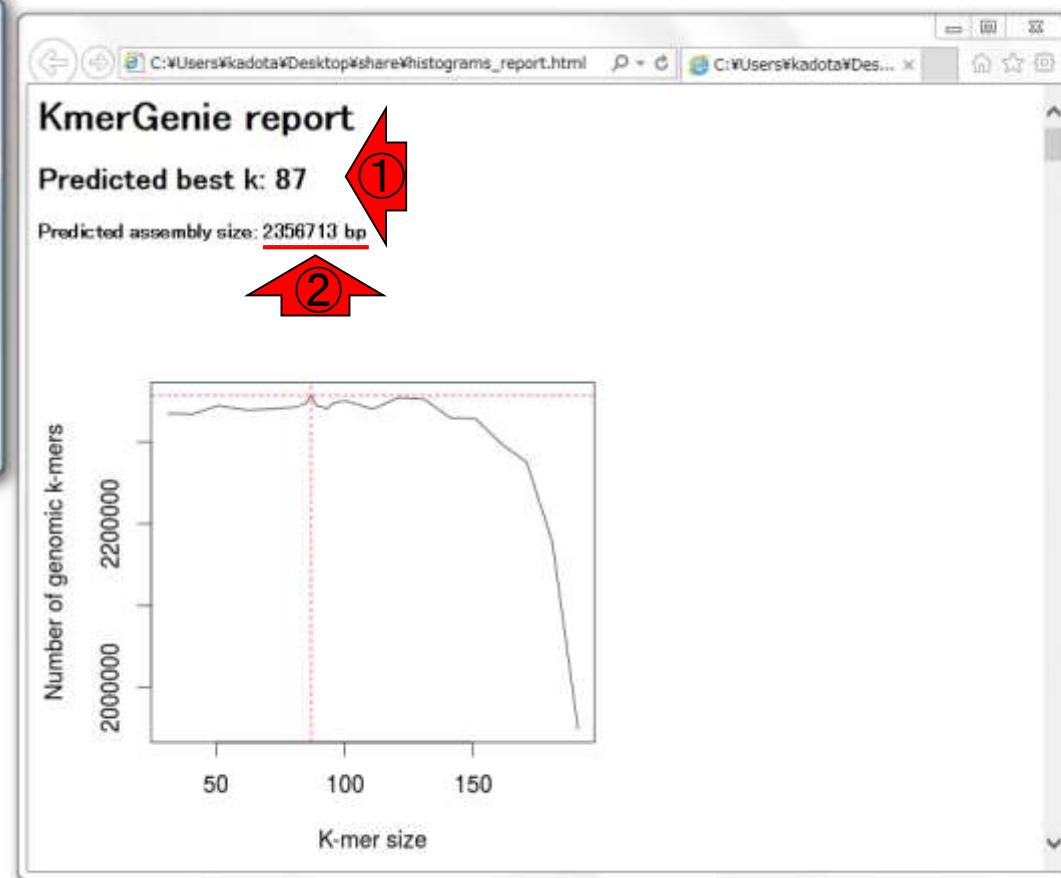
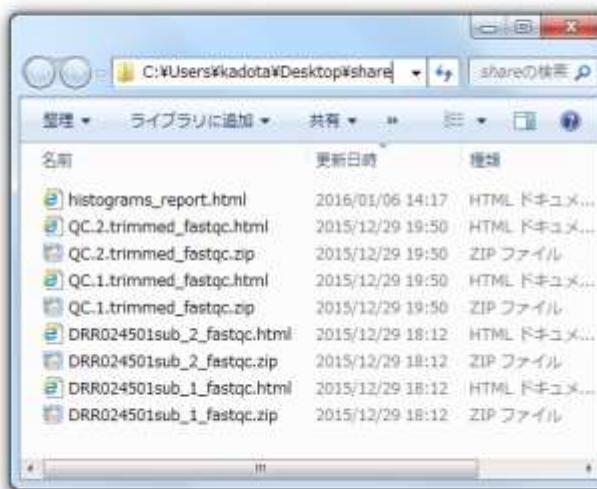
```
File Edit View Search Terminal Help
histograms-k151.histo
histograms-k151.histo.pdf
histograms-k161.histo
histograms-k161.histo.pdf
histograms-k171.histo
histograms-k171.histo.pdf
histograms-k181.histo
histograms-k181.histo.pdf
histograms-k191.histo
histograms-k191.histo.pdf
histograms-k31.histo
histograms-k31.histo.pdf
histograms-k41.histo
histograms-k41.histo.pdf
histograms-k51.histo
histograms-k51.histo.pdf
histograms-k61.histo
histograms-k61.histo.pdf
iu@bielinux[result] cp histograms_report.html ~/Desktop/mac_share
iu@bielinux[result]
```

① 1
② 2

画面が一気に流れる。KmerGenieは沢山のファイルをカレントディレクトリ上に吐き出すようだ。ファイル群histograms-k*.histo*は、個別のk値の結果を見たいときに必要となるのだろう。①このhtmlファイルさえ見れば基本的にOK。ゲストOS(Bio-Linux)上で「firefox histograms_report.html」とやってもよいが、画面が小さく見づらい。ここでは、②共有フォルダ経由でホストOSのshareフォルダにコピーして、ホストOS上でhtmlファイルを眺める

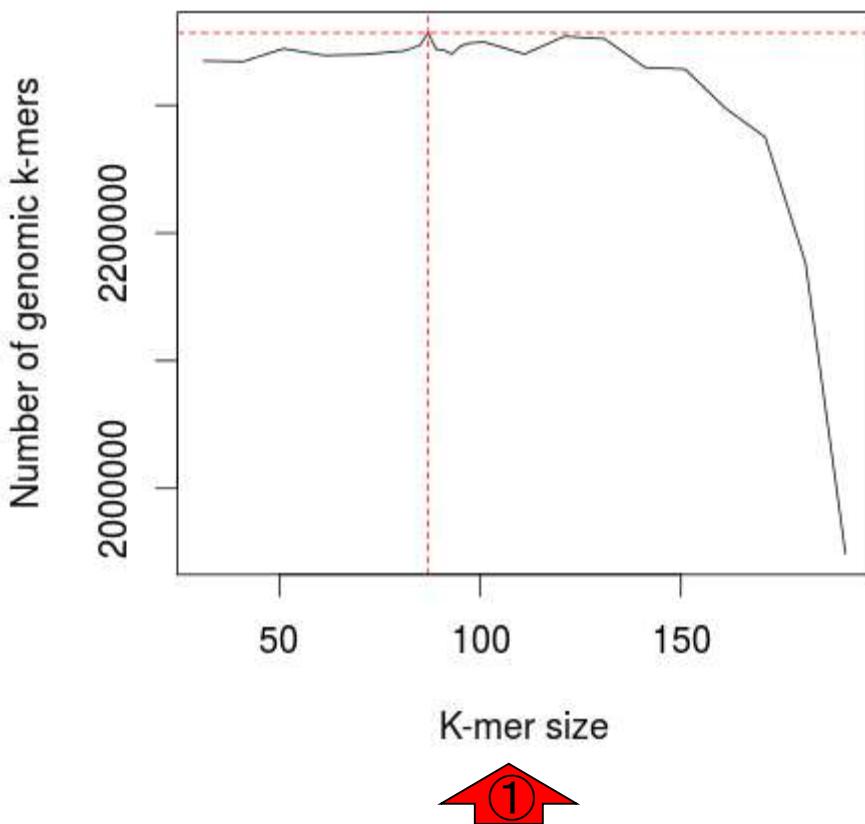
W11-9：確認

ホストOS上でhtmlファイルを眺めた画面。①アセンブリ時に用いるベストなk値は87だという結果。②が推定ゲノムサイズ。2,356,713 bp (約2.4MB)という結果



KmerGenie結果解説

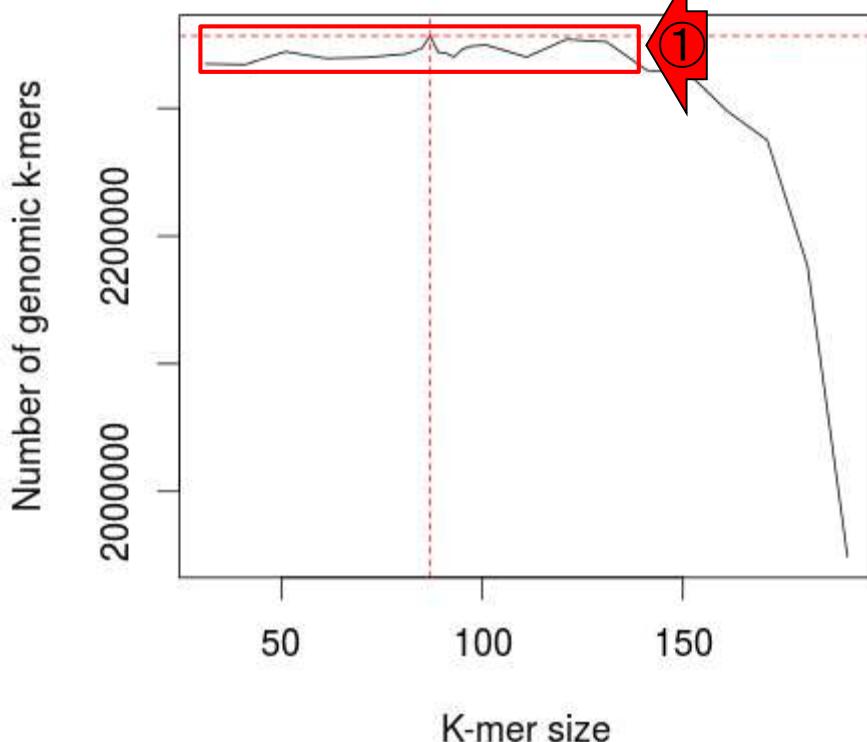
②



もう少し詳しく説明。①横軸が調べたk-merのk値。指定した探索範囲は、31から191なので妥当。縦軸の名称はNumber of *genomic* k-mers。*genomic*という形容詞がついていることからも想像できるが、これが第1部初日(2016.07.20)の最後のほうで示した「シークエンスエラー由来のものを除いたk-merの種類数」です

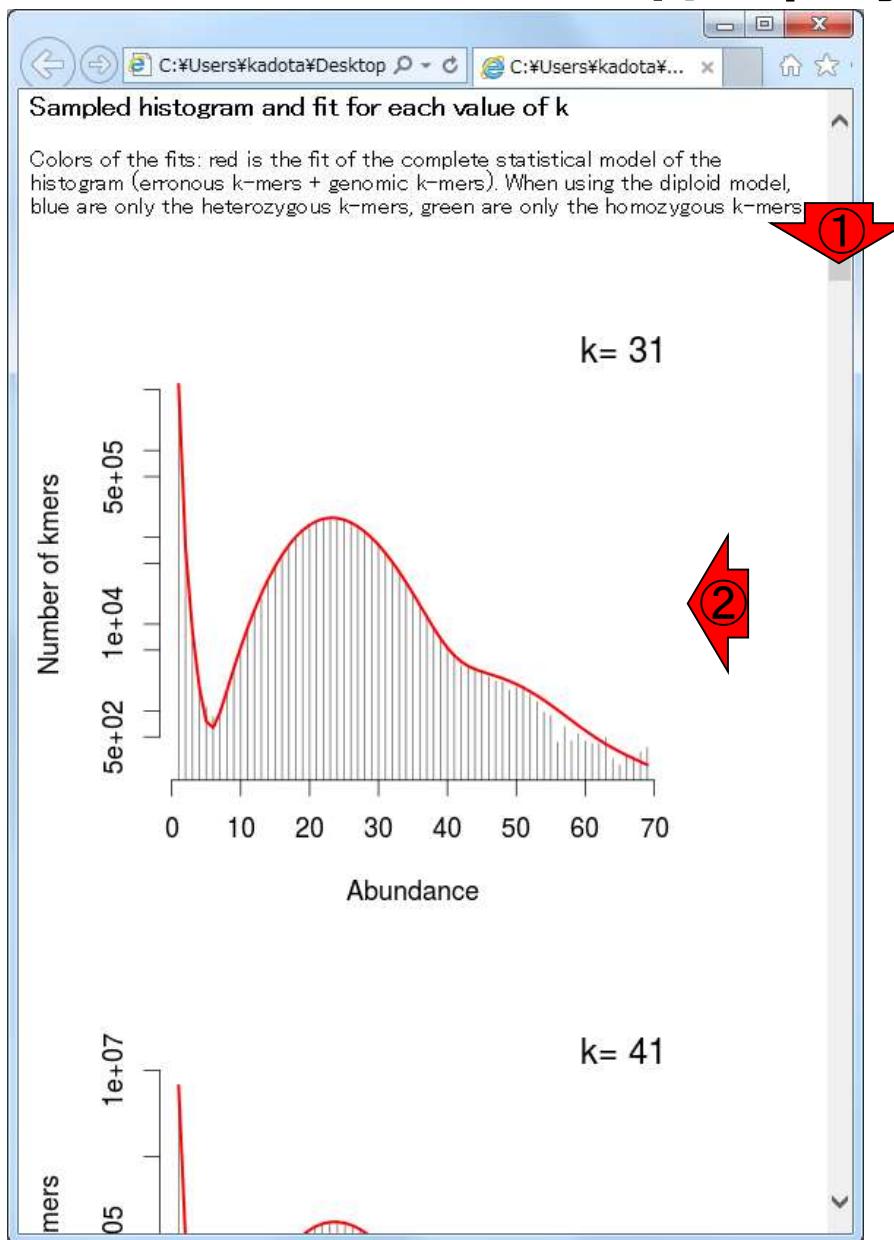


KmerGenie結果解説



①のあたりを眺めることで、 $k=30\text{--}130$ くらいの範囲でどのk-merを用いてもゲノムサイズが2.3–2.4MBという結果になるのだろうと解釈する。但し、これは「*genomic k-mers*の種類数」なのでシーケンスエラー由来k-merのフィルタリングが実装されていない(甘い)アセンブリプログラムだとそうはいかないだろう、とも想像する

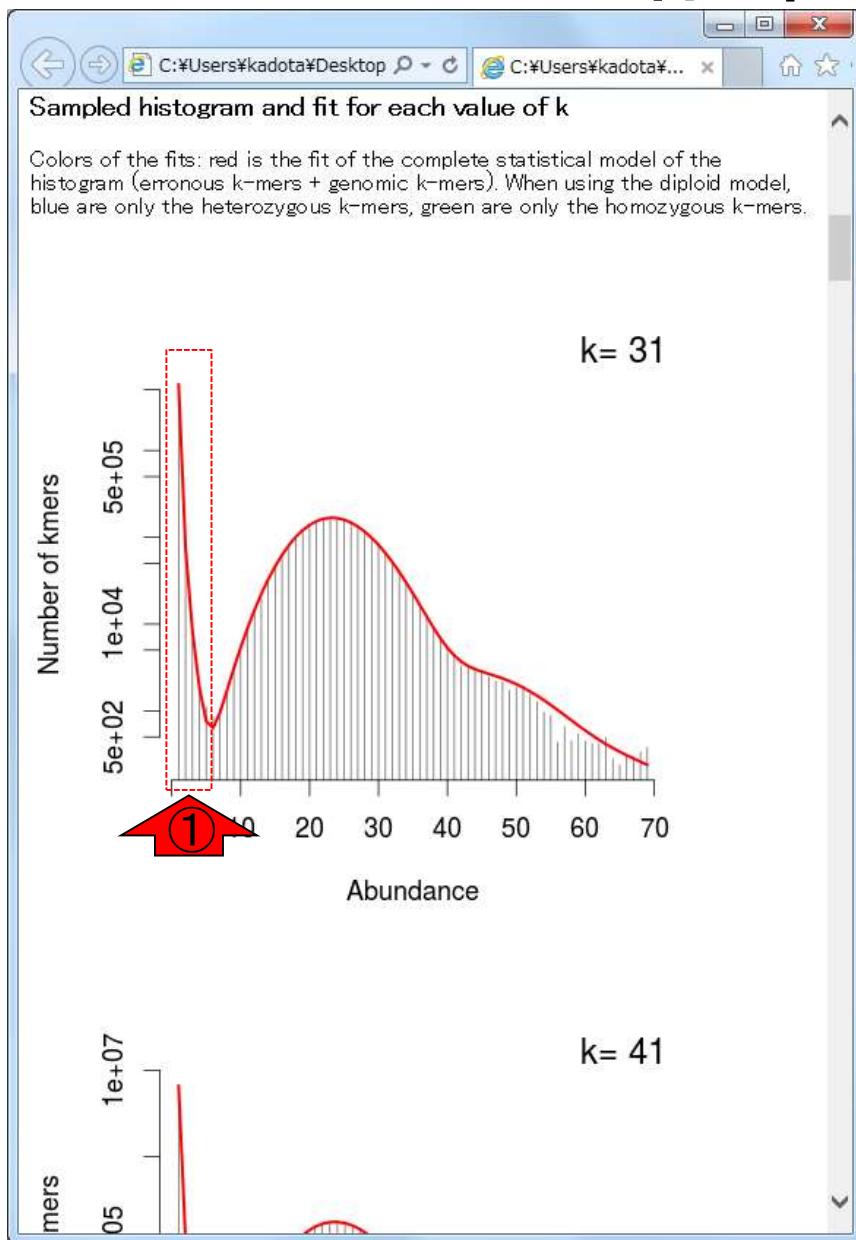
KmerGenie結果解説



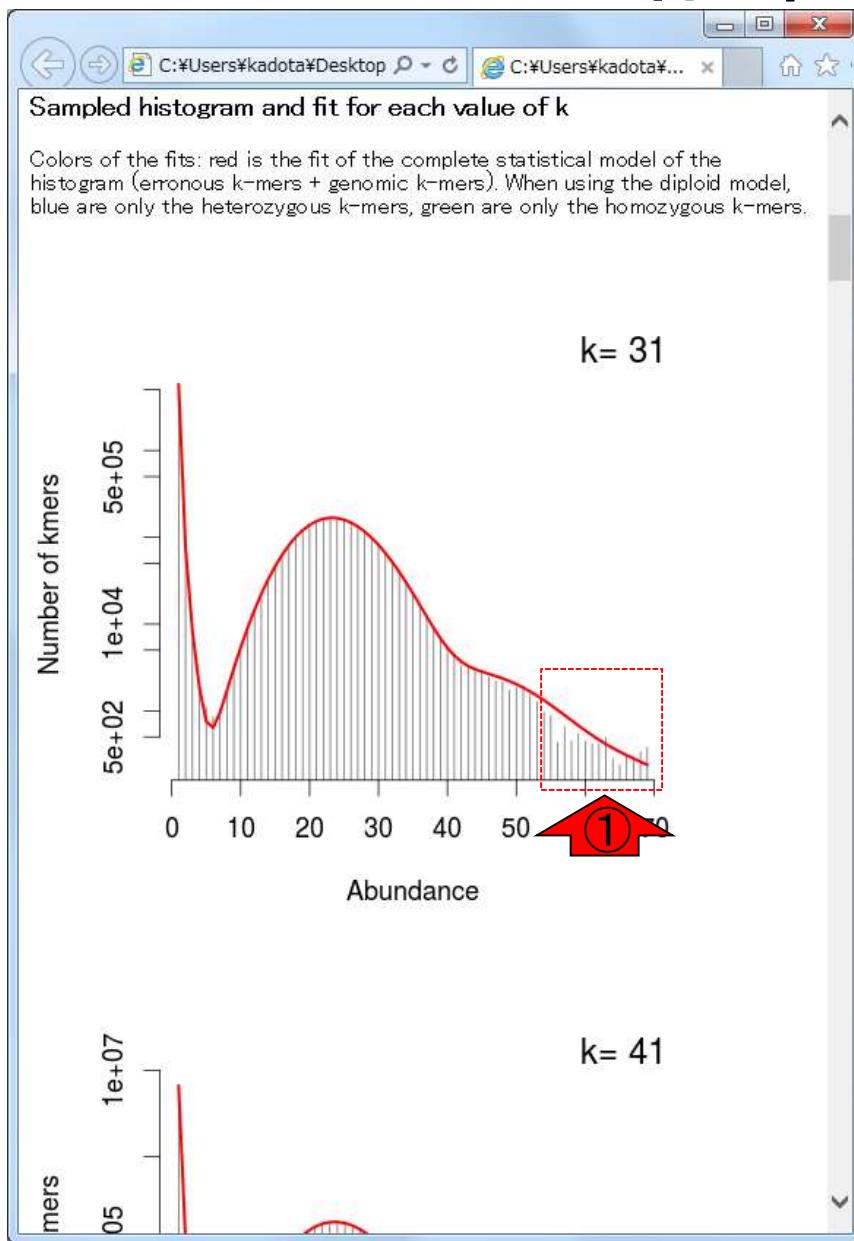
htmlファイル自体は結構縦長である。①のあたりから、オプションで指定した31から191のk値の個別の解析結果が見られる。例えば②がk=31のときのk-mer出現頻度分布

KmerGenie結果解説

KmerGenieは、①のシークエンスエラー由来k-merを除いた残りのk-merの種類数を*genomic*と判定しているのだろう



KmerGenie結果解説



ほぼ余談。実際には、①の付近のやたら沢山出現するk-merもフィルタリングしている(はず)。このあたりの多くは、リピート配列由来k-merの領域だからです

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W11-10:paired-end

The screenshot shows a Linux desktop environment. On the left, there is a vertical docked application menu with icons for various applications like a terminal, file manager, browser, and email. A red arrow labeled '①' points to the terminal icon in this menu. The main window is a terminal window titled 'iu@bielinux[result]'. It contains the following command and output:
iu@bielinux[result] pwd
/home/iu/Documents/DRR024501/result
iu@bielinux[result] less ~/Downloads/kmergenie-1.6982/README

paired-endの場合に、どう指定すればいいのか調べる。ここでは①入力ファイルがあるディレクトリ上でKmerGenieのREADMEファイルをlessで開く。打ち込み派のヒトは、タブ補完をうまく利用すべし。lessの利用法は、第3回W14-6

W11-10: paired-end

File Edit View Search Terminal Help

```
type ./kmergenie to see extra options
```

input:

Input reads should be exactly those the de novo assembler will use to create contigs, i.e. the list of all single and paired-end reads.

The order does not matter, KmerGenie treats the reads as an unordered set of k-mers. Orientation of the reads also does not matter.

With Velvet, if you have mate-pairs, Velvet uses them to create contigs, so do include them in KmerGenie.

Otherwise, if the mate-pairs are used only for scaffolding (i.e. `asm_flag=2` in SOAPdenovo), do not include them.

tips:

- * Take a look at the generated HTML report. It provides a s

①複数のファイルに分かれている場合はリストファイルを作成して入力として与えればよい。②入力ファイルの順番(order)やリードの向き(orientation)は気にしなくていいようだ。qでlessから抜ける

①

②

W11-10: paired-end

```
File Edit View Search Terminal Help  
iu@bielinux[result] pwd [10:26午前]  
/home/iu/Documents/DRR024501/result  
iu@bielinux[result] less ~/Downloads/kmergenie-1.6982/README  
iu@bielinux[result] ls [10:39午前]  
fastqCount.txt histograms-k71.histo  
histograms.dat histograms-k71.histo.pdf  
histograms.dat.pdf histograms-k81.histo  
histograms-k101.histo histograms-k81.histo.pdf  
histograms-k101.histo.pdf histograms-k85.histo  
histograms-k111.histo histograms-k85.histo.pdf  
histograms-k111.histo.pdf histograms-k87.histo  
histograms-k121.histo histograms-k87.histo.pdf  
histograms-k121.histo.pdf histograms-k89.histo  
histograms-k131.histo histograms-k89.histo.pdf  
histograms-k131.histo.pdf histograms-k91.histo  
histograms-k141.histo histograms-k91.histo.pdf  
histograms-k141.histo.pdf histograms-k93.histo  
histograms-k151.histo histograms-k93.histo.pdf  
histograms-k151.histo.pdf histograms-k95.histo  
histograms-k161.histo histograms-k95.histo.pdf
```

①lsするとhistograms*のファイルが沢山出て見づらいので削除する。実用上は削除候補ファイル群をlsやタブ補完でリストアップして、削除前に確認する(第4回のW1-2)

W11-10:paired-end

```
File Edit View Search Terminal Help
histograms-k181.histo.pdf hoge_121
histograms-k191.histo hoge_131
histograms-k191.histo.pdf hoge_151
histograms-k31.histo hoge_171
histograms-k31.histo.pdf hoge_191
histograms-k41.histo QC.1.trimmed.
histograms-k41.histo.pdf QC.2.trimmed.
histograms-k51.histo QC_qc_report.pdf
histograms-k51.histo.pdf QC.stats.txt
histograms-k61.histo QC.unpaired.trimmed.fastq
histograms-k61.histo.pdf
iu@bielinux[result] rm -f histograms*
iu@bielinux[result] ls
fastqCount.txt hoge_191
hoge_111 QC.1.trimmed.fastq.gz
hoge_121 QC.2.trimmed.fastq.gz
hoge_131 QC_qc_report.pdf
hoge_151 QC.stats.txt
hoge_171 QC.unpaired.trimmed.fastq
iu@bielinux[result]
```

[10:58午前] [10:59午前]

[10:59午前]

①rm -fで削除。②*はワイルドカードの1つ(第4回のW1-2)。ワイルドカードは使いこなせるとかなり便利。ちなみに当面の目的は③「paired-endのリストファイルを作成したい」。ここでは、ワイルドカードを駆使して美しく作成するTipsを紹介。基本戦略は「リストアップしたいもののみlsし、それをリダイレクトでファイルに書きだす」です。実際には多少余分なものがあっても書きだした上で、viなどのテキストエディタで余分なものを削除したりします

W11-10: ワイルドカード

```
File Edit View Search Terminal Help  
iu@bielinux[result] pwd  
/home/iu/Documents/DRR024501/result  
iu@bielinux[result] ls  
fastqCount.txt hoge_191  
hoge_111 QC.1.trimmed.fastq.gz  
hoge_121 QC.2.trimmed.fastq.gz  
hoge_131 QC_qc_report.pdf  
hoge_151 QC.stats.txt  
hoge_171 QC.unpaired.trimmed.fastq  
iu@bielinux[result] ls QC.*  
QC.1.trimmed.fastq.gz QC.stats.txt  
QC.2.trimmed.fastq.gz QC.unpaired.trimmed.fastq  
iu@bielinux[result] ls QC.*.tri*  
QC.1.trimmed.fastq.gz QC.unpaired.trimmed.fastq  
QC.2.trimmed.fastq.gz  
iu@bielinux[result] ls QC.[0-9].*  
QC.1.trimmed.fastq.gz QC.2.trimmed.fastq.gz  
iu@bielinux[result]
```

①「QC.*」だと、赤枠で示す目的のpaired-endファイル以外のものが表示される。②「QC.*.tri*」でも目的外のファイルが表示される。(この場合はQC.*.gzでもうまくいくが...)③任意の数字1字を表す[0-9]を利用して、目的のpaired-endファイルのみリストアップさせることができる。アセンブリプログラムPlatanus (ver. 1.2.4; Kajitani et al., 2014)のマニュアルを見るときに役立つ

[11:18午前]

[11:18午前]

[11:18午前]

[11:18午前]

Kajitani et al., *Genome Res.*, 24: 1384-1395, 2014

Aug 03 2016, NGSハンズオン講習会 Chikhi and Medvedev, *Bioinformatics*, 30: 31-37, 2014

W11-10: ワイルドカード

```
File Edit View Search Terminal Help  
iu@bielinux[result] ls  
fastqCount.txt  hoge_191  
hoge_111        QC.1.trimmed.fastq.gz  
hoge_121        QC.2.trimmed.fastq.gz  
hoge_131        QC_qc_report.pdf  
hoge_151        QC.stats.txt  
hoge_171        QC.unpaired.trimmed.fastq  
iu@bielinux[result] ls QC.* [11:18午前]  
QC.1.trimmed.fastq.gz  QC.stats.txt  
QC.2.trimmed.fastq.gz  QC.unpaired.trimmed.fastq  
iu@bielinux[result] ls QC.*.tri* [11:18午前]  
QC.1.trimmed.fastq.gz  QC.unpaired.trimmed.fastq  
QC.2.trimmed.fastq.gz  
iu@bielinux[result] ls QC.[0-9].* [11:18午前]  
QC.1.trimmed.fastq.gz  QC.2.trimmed.fastq.gz  
iu@bielinux[result] ls QC.[0-9].* > list.txt [11:18午前]  
iu@bielinux[result] more list.txt [11:56午前]  
QC.1.trimmed.fastq.gz  
QC.2.trimmed.fastq.gz  
iu@bielinux[result] [11:57午前]
```

③でリストアップできることを確認したら、④リダイレクト(>)を利用して任意のファイル名(ここではlist.txt)で保存すれば、リストファイルの完成。この種のテクニックは頻用する

W11-11 : paired-end実行

The terminal window shows the following session:

```
iu@bielinux[result] ls
fastqCount.txt  list.txt
hoge_111        QC.1.trimmed.fastq.gz
hoge_121        QC.2.trimmed.fastq.gz
hoge_131        QC_qc_report.pdf
hoge_151        QC.stats.txt
hoge_171        QC.unpaired.trimmed.fastq
hoge_191

iu@bielinux[result] kmergenie list.txt -l 31 -k 191
running histogram estimation
File list.txt starts with character "Q", hence is interpreted as a list of file names
Reading 2 read files
Linear estimation: ~103 M distinct 106-mers are in the reads
K-mer sampling: 1/31
| processing
|
[going to estimate histograms for values of k: 191 181 171 161
151 141 131 121 111 101 91 81 71 61 51 41 31
-----]
```

Red arrows labeled ① and ② point to the 'list.txt' file in the directory and the 'kmergenie' command respectively.

①リストファイルlist.txtがあることを確認して、②kmergenieコマンドを実行。画面は数秒後の状態。うまくリストファイルを読み込んでいるようだ。約20分

W11-11: 無事終了

```
File Edit View Terminal Help Ja
estimation of the best k so far: 141
refining estimation around [135; 147], with a step of 2
running histogram estimation
File list.txt starts with character "Q", hence is interpreted as a list of file names
Reading 2 read files
Linear estimation: ~119 M distinct 84-mers are in the reads
K-mer sampling: 1/36
| processing
|
[going to estimate histograms for values of k: 147 145 143 141
139 137 135
-----
----- Total time Wallclock 286.84
s
fitting model to histograms to estimate best k
table of predicted num. of genomic k-mers: histograms.dat
recommended coverage cut-off for best k: 2
best k: 141
iu@bielinux[result] ls
```

終了後の状態。forward側のみ(single-end)で実行した結果($k=87$)と違って、① paired-endのときは $k=141$ がお勧めとなっていることがわかる。とりあえず②ls

①

②

[12:30午後]

W11-11：確認

```
File Edit View Terminal Help
histograms-k137.histo      histogram
histograms-k137.histo.pdf    histograms_k137.histo
histograms-k139.histo      histograms-k81.histo.pdf
histograms-k139.histo.pdf    histograms-k91.histo
histograms-k141.histo      histograms-k91.histo.pdf
histograms-k141.histo.pdf   histograms_report.html
histograms-k143.histo      hoge_111
histograms-k143.histo.pdf   hoge_121
histograms-k145.histo      hoge_131
histograms-k145.histo.pdf   hoge_151
histograms-k147.histo      hoge_171
histograms-k147.histo.pdf   hoge_191
histograms-k151.histo      list.txt
histograms-k151.histo.pdf   QC.1.trimmed.fastq.gz
histograms-k161.histo      QC.2.trimmed.fastq.gz
histograms-k161.histo.pdf   QC_qc_report.pdf
histograms-k171.histo      QC.stats.txt
histograms-k171.histo.pdf   QC.unpaired.trimmed.fastq
iu@bielinux[result] cp histograms_report.html ~/Desktop/mac_share/histograms_report_paired.html
```

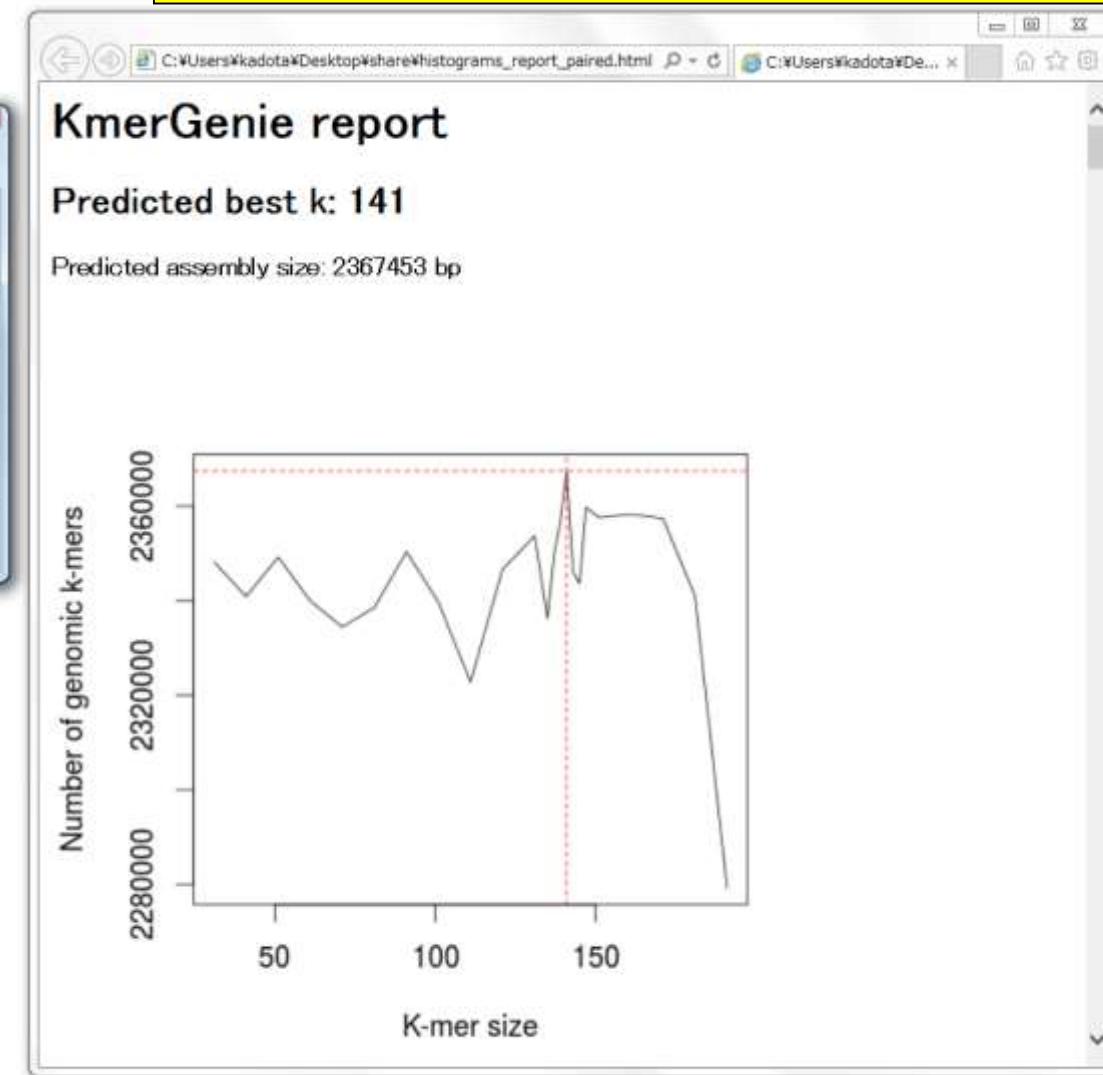
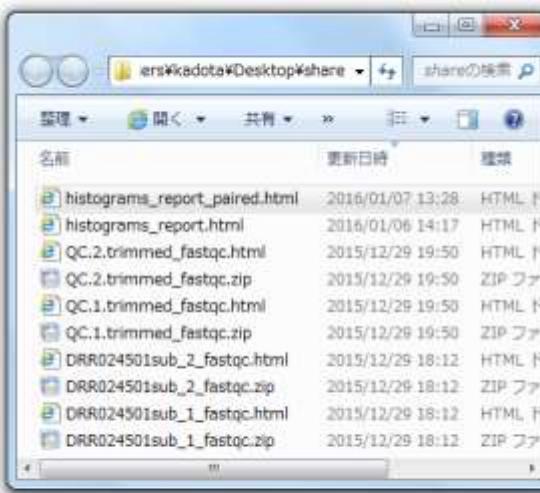
W11-9と同様に画面が一気に流れる。ここでは、
(single-endの結果ファイルがhistograms_report.htmlとして既に存在するので)②共有フォルダ経由でホストOSのshareフォルダにhistograms_report_paired.htmlという名前でコピーして、ホストOS上でhtmlファイルを眺める

①

②

W11-11：確認

single-end ($k=87$)とpaired-end ($k=141$)で推奨 k 値は大きく異なるものの、ゲノムサイズの推定値はほとんど変わらないことがわかる(single-endは2356713 bp、paired-endは2367453 bp)



Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6: FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



第6回原稿PDFのp45

億バイト)であったことを思い出せば納得できるであろう。尚、このデータの正解は、配列数が3 (1 chromosome + 2 plasmids)、2,400,586 bp (約 2.4MB) である⁴⁾。k 値の選択の重要性がよくわかる例といえよう。

通常、Velvetを実行する場合は複数の異なるk値を用いてアセンブルを行い、それらの結果を眺める[W10]。ここでは、計10個のk値 (k=31, 61, 91, 111, 121, 131, 151, 171, 181, 191) で実行した結果を眺め、主に配列数の観点から、k=171周辺の結果が一番よさそうだと解釈する。もちろんこのデータの場合は、「真のゲノムサイズは約2.4MB」だという答えがわかった状態でアセンブリ結果の評価を行っていることになるが、実際には近縁種との比較により妥当と考えられるゲノムサイズを検討する。ここではそのような情報が得られなかつと仮定して「ゲノムサイズ推定」を行い、アセンブリ結果の評価を行う。

ゲノムサイズ推定

ゲノムサイズの推定は、フローサイトメトリー (flow cytometry) という手法を用いて実験的に求めるやり方

かる。推奨のk値が大きく異なるのは、paired-end ($k=141$) では single-end ($k=87$) に比べデータ量が単純に2倍になっているからである。実際には paired-end のデータを用いてアセンブリを行うので、paired-endでの推奨k値 (=141) の前後である $k=131\text{--}191$ あたりを手厚く探索するとよい結果が得られそうだと判断する。

配列長によるフィルタリング

比較的マイナーな事柄ではあるが、通常下記に示す3つの理由から、アセンブリ結果から短い配列を除外する：

1. MiSeqを含むショートリードのde novoアセンブリでは、挿入配列 (insertion sequence) やリボソームRNA遺伝子領域 (rDNA) のような、ゲノム中に複数コピーが散在する反復領域 (dispersed repeat) の再現は難しい。配列 (コンティグ) がこれらの反復領域部分で分断されてしまうからである。アセンブリ結果に含まれる短いコンティグは、これらの反復領域の一部である場合が多く、その後の解析には大きな影響を及

W12-1:ダウンロード

http://www.iu.a.u-tokyo.ac.jp/~kadota/book/fastaLengthFilter.py

```
#!/usr/bin/env python
# coding:utf-8

import math
import sys

class Fasta():

    @staticmethod
    def read(inputfile):
        a = open(inputfile).read()
        entries = a.split(">")[1:]
        for entry in entries:
            lines = entry.split("\n")
            fasta = Fasta()
            fasta.header = ">" + lines[0]
            fasta.seq = "".join(lines[1:])
            fasta.length = len(fasta.seq)
            yield fasta

    def main(fileName, threshold=0):
        seqList = [seq for seq in Fasta.read(fileName) if seq.length >= threshold]
        seqList.sort(key=lambda seq: seq.length, reverse=True)
        digit = int(math.log10(len(seqList))) + 1
        for i, seq in enumerate(seqList):
            print ">sequence" + str(i + 1).zfill(digit)
            print seq.seq

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print
        print "#tRemove sequences shorter than threshold."
        print "#tUsage : fastaLengthFilter.py <filename> <threshold>"
        print
        exit()

    fileName, threshold = sys.argv[1:3]
    main(fileName, int(threshold))
```

multi-FASTA形式ファイルを入力として、指定した配列長未満の配列を除くPythonプログラム (fastaLengthFilter.py; 第6回筆頭著者作) をダウンロード。簡単な自作プログラムは~/binに置き、そこにパスを通して使っている(ヒトもいる)。そうすることで毎回/usr/local/binなどにシンボリックリンクをはる手間が省ける

W12-1: ダウンロード

The screenshot shows a terminal window with a dark theme. On the left, there is a vertical dock with icons for various applications: a terminal, file manager, browser, email, calendar, and others. The terminal window has a red border and contains the following text:

```
iu@bielinux[result] cd  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls  
Desktop Downloads Music Public Videos  
Documents igv Pictures Templates  
iu@bielinux[iu] mkdir bin [12:23午後]  
iu@bielinux[iu] ls [12:23午後]  
bin Documents igv Pictures Templates  
Desktop Downloads Music Public Videos  
iu@bielinux[iu] cd bin [12:23午後]  
iu@bielinux[bin] pwd [12:23午後]  
/home/iu/bin  
iu@bielinux[bin] wget -cq http://www.iu.a.u-tokyo.ac.jp/~kadota/book/fastaLengthFilter.py  
iu@bielinux[bin] ls -l [12:23午後]  
total 4  
-rw-rw-r-- 1 iu iu 1109 1月 6 18:57 fastaLengthFilter.py  
iu@bielinux[bin]
```

Red numbered arrows point to specific parts of the terminal output:

- ① Points to the terminal icon in the dock.
- ② Points to the browser icon in the dock.
- ③ Points to the calendar icon in the dock.
- ④ Points to the file manager icon in the dock.
- ⑤ Points to the file "fastaLengthFilter.py" in the terminal output.
- ⑥ Points to the file "fastaLengthFilter.py" in the terminal output.

A yellow box highlights the text "在するURLを指定すればそうなるが)ダウ色になっていないの1)が自分(この場合iu)".

①ホームディレクトリに移動し、binディレクトリがないことを確認して、②mkdirで作成。③binディレクトリに移動し、④fastaLengthFilter.pyが存在するURLを指定してwget。⑤(実行権限があればそうなるが)ダウンロードしたファイルが緑色になっていないので、⑥実行権限(第4回W3-1)が自分(この場合iu)にもないことを認識する

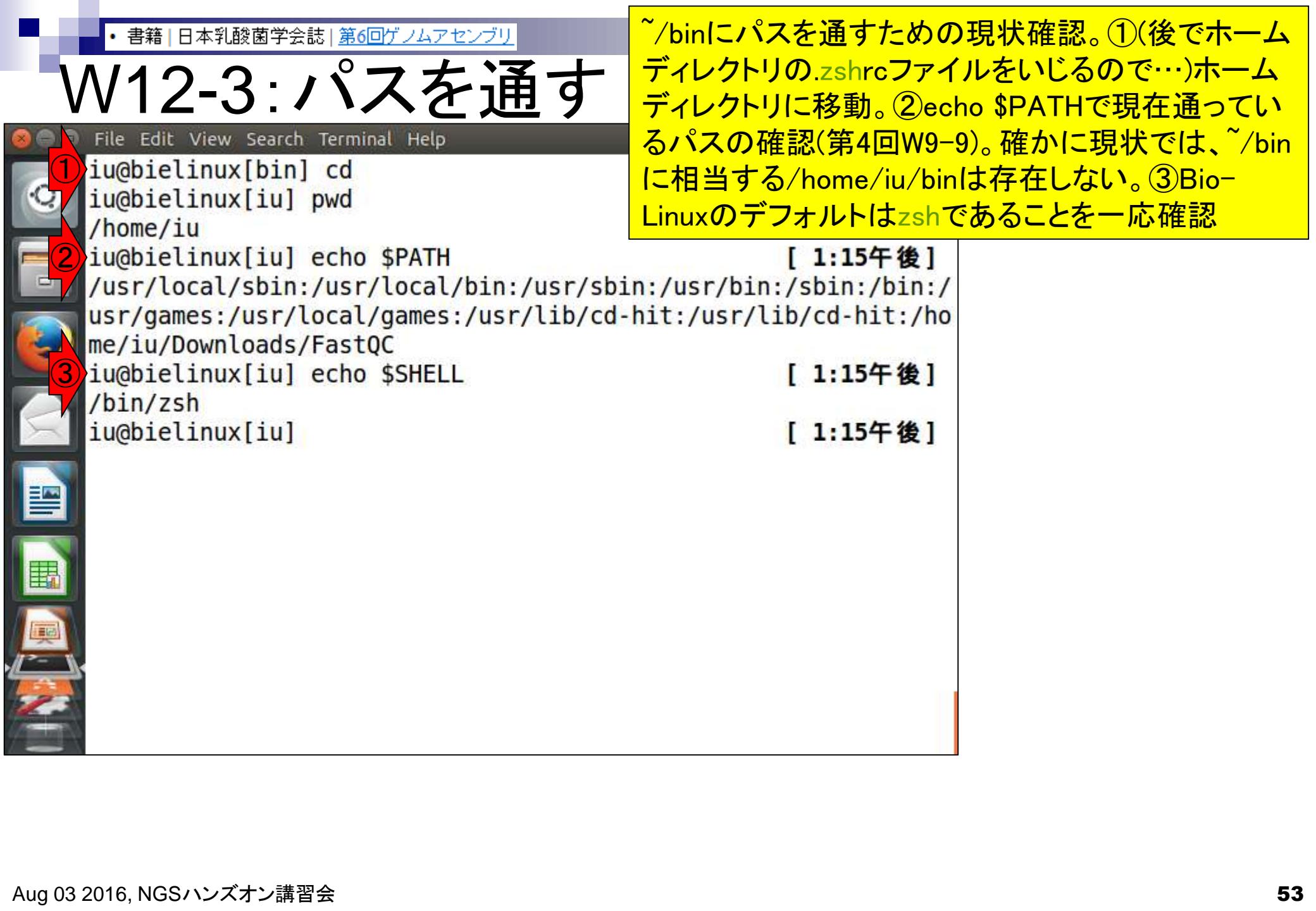
W12-2:permission

```
File Edit View Search Terminal Help  
iu@bielinux[bin] pwd  
/home/iu/bin  
iu@bielinux[bin] ls -l  
total 4  
-rw-rw-r-- 1 iu iu 1109 1月 6 18:57 fastaLengthFilter.py  
iu@bielinux[bin] chmod 755 fastaLengthFilter.py [12:44午後]  
iu@bielinux[bin] ls -l  
total 4  
-rwxr-xr-x 1 iu iu 1109 1月 6 18:57 fastaLengthFilter.py  
iu@bielinux[bin] █ [12:44午後]
```



①chmodで実行権限を変更。オプションで「755」とすると「rwxr-xr-x」になる。最初の3文字分は自分の権限に関する部分。「読み込み(r)、書き込み(w)、実行(x)」の全てを許可するという意味でrwxとなっている。ここでは、自分以外には書き込み権限を与えない設定にしている。私はいつも755

W12-3: パスを通す



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title bar says "File Edit View Search Terminal Help". The terminal content is as follows:

```
iu@bielinux[bin] cd  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:  
usr/games:/usr/local/games:/usr/lib/cd-hit:/usr/lib/cd-hit:/ho  
me/iu/Downloads/FastQC  
iu@bielinux[iu] echo $SHELL  
/bin/zsh  
iu@bielinux[iu]
```

Three red arrows numbered 1, 2, and 3 point to the icons in the vertical dock on the left side of the screen.

~/binにパスを通すための現状確認。①(後でホームディレクトリの.zshrcファイルをいじるので...)ホームディレクトリに移動。②echo \$PATHで現在通っているパスの確認(第4回W9-9)。確かに現状では、~/binに相当する/home/iu/binは存在しない。③Bio-Linuxのデフォルトはzshであることを一応確認

[1:15 午後]

[1:15 午後]

[1:15 午後]

W12-3: パスを通す

```
File Edit View Search Terminal Help  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls -l .zshrc  
-rw----- 1 iu iu 392 12月 22 15:19 .zshrc  
iu@bielinux[iu] tail -n 5 .zshrc  
# fi  
  
export PATH=$PATH:/home/iu/Downloads/FastQC  

```

①.zshrcファイルの存在確認と②最後の5行分を表示。③テキストエディタgedit(viやemacsでもよい)を用いて、赤枠部分に:`./home/iu/bin`を追加して保存する。geditの使い方は第4回W10-3にもあります。

~/Desktop/backup
に.zshrc_orgがあります。消しちゃったヒトはそれで復旧してください

W12-3: パスを通す

赤枠部分に:/home/iu/binを追加して保存し、geditを終了した後の状態。④もう一度.zshrcファイルの最後の5行分を表示

The screenshot shows a terminal window with a command-line interface. On the left, there is a vertical docked application bar with icons for various applications. Red numbered arrows (1, 2, 3, 4) point to the icons for the terminal, file manager, email client, and browser respectively. The terminal window contains the following text:

```
iu@bielinux[iu] pwd [ 1:17 午後]
/home/iu
iu@bielinux[iu] ls -l .zshrc [ 1:18 午後]
-rw----- 1 iu iu 392 12月 22 15:19 .zshrc
iu@bielinux[iu] tail -n 5 .zshrc [ 1:18 午後]
# fi

export PATH=$PATH:/home/iu/Downloads/FastQC [red box]
export CLASSPATH=/home/iu/Downloads/Rockhopper.jar
iu@bielinux[iu] gedit .zshrc [ 1:18 午後]

** (gedit:32481): WARNING **: Error querying file info: Error
when getting information for file '/home/iu/.goutputstream-3XD
NAY': No such file or directory
iu@bielinux[iu] tail -n 5 .zshrc [ 1:23 午後]
```

The text in the red box is the line `export PATH=$PATH:/home/iu/Downloads/FastQC`. The terminal window has a dark theme with white text on a black background. The docked bar has a light gray background.

W12-3: パスを通す

The screenshot shows a terminal window with a dark theme. On the left, there is a vertical dock with icons for a terminal, file manager, browser, email, and file viewer. Red numbered arrows (2, 3, 4) point to the icons for terminal, file manager, and file viewer respectively. The terminal window displays the following session:

```
iu@bielinux[iu] ls -l .zshrc [ 1:18 午後]
-rw----- 1 iu iu 392 12月 22 15:19 .zshrc
iu@bielinux[iu] tail -n 5 .zshrc [ 1:18 午後]
# fi

export PATH=$PATH:/home/iu/Downloads/FastQC [red box]

export CLASSPATH=/home/iu/Downloads/Rockhopper.jar
iu@bielinux[iu] gedit .zshrc [ 1:18 午後]

** (gedit:32481): WARNING **: Error querying file info: Error
when getting information for file '/home/iu/.goutputstream-3XD
NAY': No such file or directory
iu@bielinux[iu] tail -n 5 .zshrc [ 1:23 午後]
# fi

export PATH=$PATH:/home/iu/Downloads/FastQC:/home/iu/bin [red box]

export CLASSPATH=/home/iu/Downloads/Rockhopper.jar
iu@bielinux[iu] [ 1:27 午後]
```

The command `gedit .zshrc` is shown running, indicated by a red arrow pointing to the file viewer icon. The line `# fi` is shown being typed at the end of the file. The final command `iu@bielinux[iu]` is shown with a black bar over it, indicating the terminal is still active.

W12-4：確認

```
File Edit View Search Terminal Help  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls -l .zshrc  
-rw----- 1 iu iu 405 1月 8 13:23 .zshrc  
iu@bielinux[iu] echo $PATH [ 1:37 午後 ]  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/  
usr/games:/usr/local/games:/usr/lib/cd-hit:/usr/lib/cd-hit:/ho  
me/iu/Downloads/FastQC  

```

①「gedit .zshrc」を実行したこのターミナルは、まだターミナル起動時の環境設定のまま。②source コマンドで編集後の環境設定ファイル(.zshrc)を再読み込みしておく。その後もう一度echo \$PATHで確認すると、③確かに自分が追加した赤枠のものが見えている。基本的にこれでもうOK

W12-4：確認

参考

File Edit View Search Terminal Help

```
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls -l .zshrc  
-rw----- 1 iu iu 405 1月 8 13:23 .zshrc  
iu@bielinux[iu] echo $PATH [ 1:37 午後 ]  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:  
usr/games:/usr/local/games:/usr/lib/cd-hit:/usr/lib/cd-hit:/ho  
me/iu/Downloads/FastQC  
iu@bielinux[iu] source .zshrc [ 1:37 午後 ]  
iu@bielinux[iu] echo $PATH [ 1:38 午後 ]  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:  
usr/games:/usr/local/games:/usr/lib/cd-hit:/usr/lib/cd-hit:/ho  
me/iu/Downloads/FastQC:/home/iu/Downloads/FastQC:/home/iu/bin  

```

以下はおまけです。よく見ると、①の赤下線部分が重複しており、②左側にあるものと全く同じ。やらかしてしまったのではないかと思うが、重複しているだけで実際には問題ない。だって、③ここにも重複があるが、問題ないから。重複を防ぐには、.zshrc編集時に余分に追加されそうなところを削除しておけばよい。つまり…

②

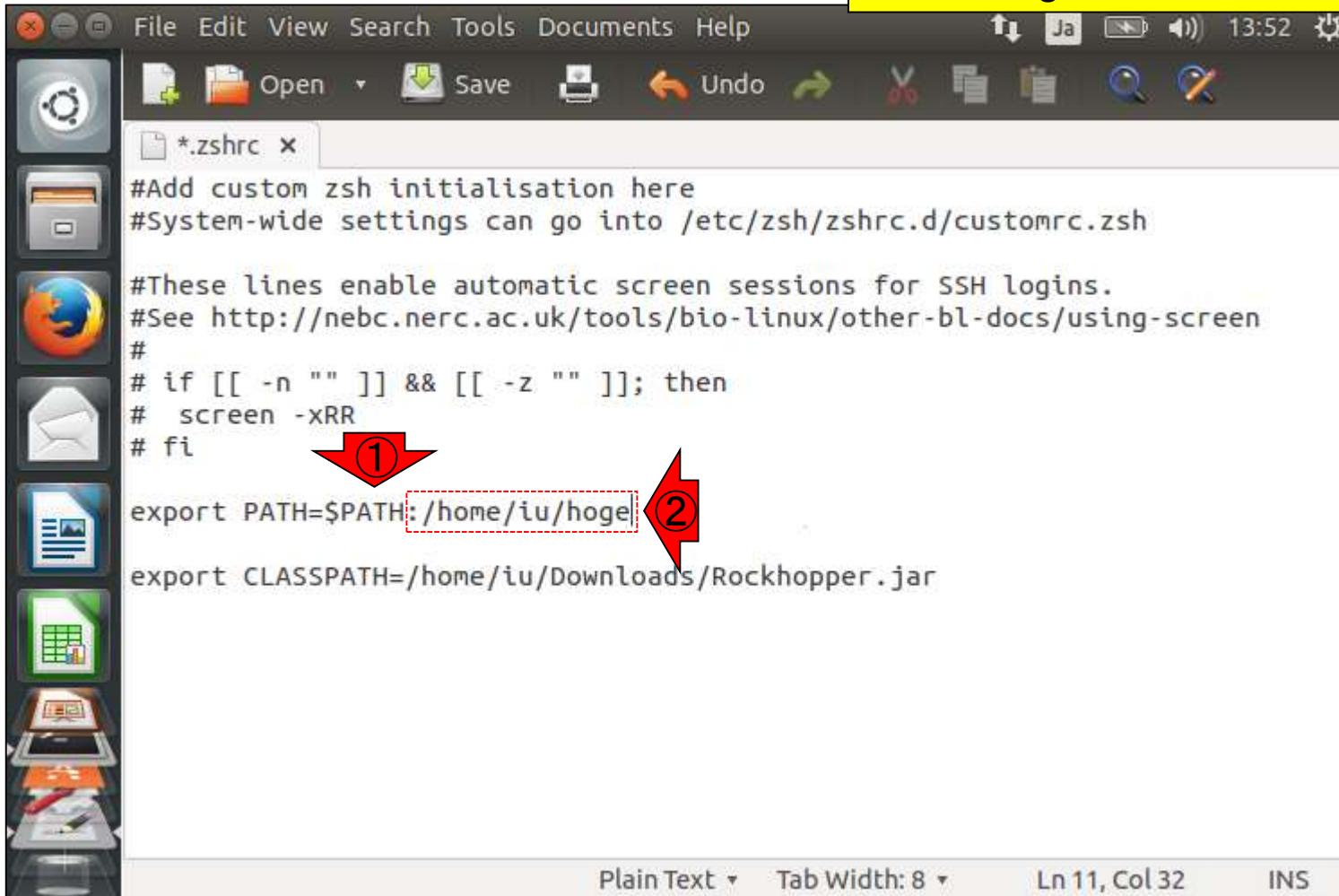
①

③

W12-4：確認

参考

例えば~/hogeディレクトリ以下のファイルにパスを通して
たい場合は、①の「\$PATH」の部分のみ残して、②赤
枠の~/hogeディレクトリの絶対パスを追加すればよい



```
File Edit View Search Tools Documents Help
Open Save Undo Redo Cut Copy Paste Find Replace
*.zshrc x
#Add custom zsh initialisation here
#System-wide settings can go into /etc/zsh/zshrc.d/customrc.zsh

#These lines enable automatic screen sessions for SSH logins.
#See http://nebc.nerc.ac.uk/tools/bio-linux/other-bl-docs/using-screen
#
# if [[ -n "" ]] && [[ -z "" ]]; then
#   screen -xRR
# fi
export PATH=$PATH:/home/iu/hoge
export CLASSPATH=/home/iu/Downloads/Rockhopper.jar

Plain Text Tab Width: 8 Ln 11, Col 32 INS
```

W12-5: パスと改行コード

```
File Edit View Search Terminal Help  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls  
bin Documents igv Pictures Templates  
Desktop Downloads Music Public Videos  
iu@bielinux[iu] where fastaLengthFilter.py  
/home/iu/bin/fastaLengthFilter.py  
/home/iu/bin/fastaLengthFilter.py  
iu@bielinux[iu] file ~/bin/fastaLengthFilter.py [ 6:12 午後 ]  
/home/iu/bin/fastaLengthFilter.py: Python script, ASCII text e  
xecutable  
iu@bielinux[iu]
```

① ②

A vertical dock on the left side of the screen contains several icons: a terminal window, a file browser, a file manager, a web browser (Firefox), an email client, a file viewer, a spreadsheet application, a presentation slide, and a terminal icon.

①whereコマンドでパスが通っているこ
とを念のため確認。②fileコマンドで
Linuxの改行コードになっているかどうか
確認。「ASCII text executable」で終わ
っている場合は問題ないが、「ASCII
text executable, with CRLF line
terminators」となっている場合は、第4
回W13-6を参考にして、nkfコマンドを用
いてLinuxの改行コードに変換しておく

[6:12 午後]

[6:12 午後]

W12-6:ヘルプを表示

```
File Edit View Search Terminal Help  
iu@bielinux[iu] pwd  
/home/iu  
iu@bielinux[iu] ls  
bin Documents igv Pictures Template  
Desktop Downloads Music Public Videos  
iu@bielinux[iu] fastaLengthFilter.py -h  
  
② Remove sequences shorter than threshold.  
Usage : fastaLengthFilter.py <filename> <threshold>  
  
iu@bielinux[iu] fastaLengthFilter.py  
  
Remove sequences shorter than threshold.  
Usage : fastaLengthFilter.py <filename> <threshold>  
  
iu@bielinux[iu]
```

①-hをつけてfastaLengthFilter.pyを実行。
②このプログラムが何をするものなのについての簡単なdescription、および③基本的な利用法(Usage)が表示される。④実は①は偶然うまくいっただけ。このプログラムに関しては、-hオプションを受け付けてはおらず、入力ファイル(第1引数)とthreshold(第2引数)の2つのオプションを同時に入力しなかった場合に②と③が表示される。つまり④が正解

[6:23 午後]

[6:23 午後]

W12-7: 実行準備

fastalengthFilter.pyを実行すべく、①W10-5で作成したVelvetアセンブリ結果ファイル(contigs.fa)に近いディレクトリに移動し、②ls

```
File Edit View Search Terminal Help  
iu@bielinux[iu] pwd [ 6:39 午後 ]  
/home/iu  
iu@bielinux[iu] ls [ 6:39 午後 ]  
bin Documents igv Pictures Templates  
Desktop Downloads Music Public Videos  
iu@bielinux[iu] fastalengthFilter.py -h [ 6:39 午後 ]  
  
Remove sequences shorter than threshold.  
Usage : fastalengthFilter.py <filename> <threshold>  
  
iu@bielinux[iu] fastalengthFilter.py [ 6:39 午後 ]  
  
Remove sequences shorter than threshold.  
Usage : fastalengthFilter.py <filename> <threshold>  
  
① iu@bielinux[iu] cd ~/Documents/DRR024501/result [ 6:39 午後 ]  
iu@bielinux[result] pwd [ 6:39 午後 ]  
/home/iu/Documents/DRR024501/result  
iu@bielinux[result] ls [ 6:39 午後 ]
```

W12-7: 実行準備

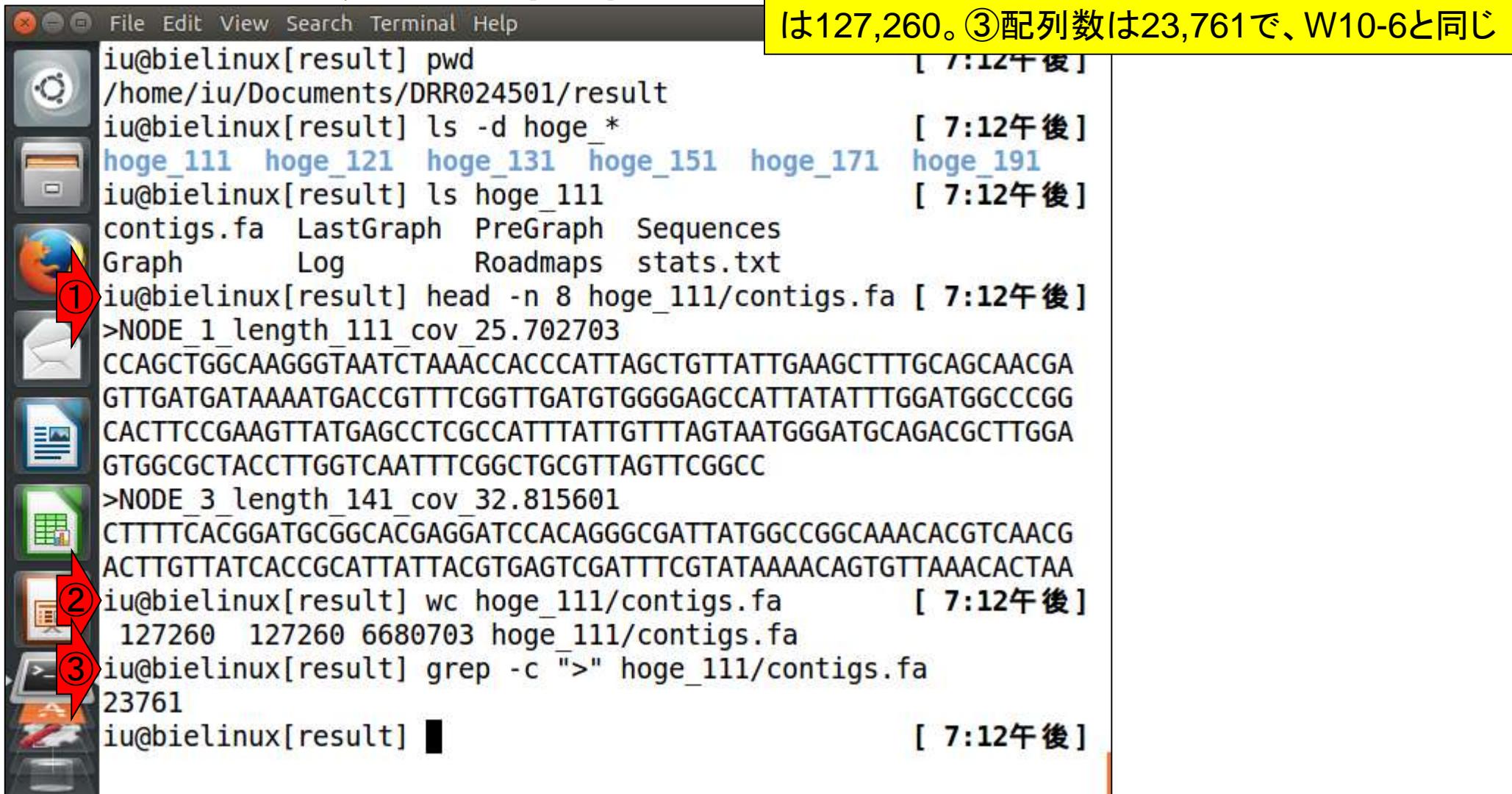
```
File Edit View Search Terminal Help
histograms-k143.histo      hoge_111
histograms-k143.histo.pdf   hoge_121
histograms-k145.histo      hoge_131
histograms-k145.histo.pdf   hoge_151
histograms-k147.histo      hoge_171
histograms-k147.histo.pdf   hoge_191
histograms-k151.histo      list.txt
histograms-k151.histo.pdf   QC.1.trimmed.fastq.gz
histograms-k161.histo      QC.2.trimmed.fastq.gz
histograms-k161.histo.pdf   QC_qc_report.pdf
histograms-k171.histo      QC.stats.txt
histograms-k171.histo.pdf   QC.unpaired.trimmed.fastq
iu@bielinux[result] rm -f histograms*          [ 6:44 午後 ]
iu@bielinux[result] ls                          [ 6:44 午後 ]
fastqCount.txt  list.txt
hoge_111        QC.1.trimmed.fastq.gz
hoge_121        QC.2.trimmed.fastq.gz
hoge_131        QC_qc_report.pdf
hoge_151        QC.stats.txt
hoge_171        QC.unpaired.trimmed.fastq
hoge_191
iu@bielinux[result] [ 6:44 午後 ]
```

W11-11で作成したKmerGenie実行結果ファイル群(histograms*)が一気に表示されて見づらいので①削除。とりあえず②k=111で実行したVelvetアセンブリ結果フォルダhoge_111中のcontigs.faを例題としてfiltreringを行う

W12-7: 実行準備

```
iu@bielinux[result] pwd [ 7:12 午後]
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls -d hoge_*
hoge_111  hoge_121  hoge_131  hoge_151  hoge_171  hoge_191 [ 7:12 午後]
iu@bielinux[result] ls hoge_111 [ 7:12 午後]
contigs.fa  LastGraph  PreGraph  Sequences
Graph      Log        Roadmaps  stats.txt
iu@bielinux[result] head -n 8 hoge_111/contigs.fa [ 7:12 午後]
>NODE_1_length_111_cov_25.702703
CCAGCTGGCAAGGGTAATCTAAACCACCCATTAGCTGTTATTGAAGCTTGAGAACGA
GTTGATGATAAAAATGACCCTTCGGTTGATGTGGGGAGCCATTATTTGGATGGCCCGG
CACTTCCGAAGTTATGAGCCTCGCCATTATTGTTAGTAATGGGATGCAGACGCTTGGA
GTGGCGCTACCTTGGTCAATTTCGGCTGCGTTAGTCGGCC
>NODE_3_length_141_cov_32.815601
CTTTTCACGGATGCGGCACCGAGGATCCACAGGGCGATTATGCCGGCAAACACGTCAACG
ACTTGTTATCACCGCATTATTACGTGAGTCGATTCGTATAAAACAGTGTAAACACTAA
iu@bielinux[result] wc hoge_111/contigs.fa [ 7:12 午後]
127260 127260 6680703 hoge_111/contigs.fa
iu@bielinux[result] grep -c ">" hoge_111/contigs.fa
23761
iu@bielinux[result] [ 7:12 午後]
```

① hoge_111/contigs.fa の最初の8行分を表示。
description行は「NODE_...」みたいな感じになっていることがわかる。② wc実行結果を表示。行数は127,260。③配列数は23,761で、W10-6と同じ



W12-8: 実行

①hoge_111/contigs.faを入力として
300 bp以上の配列のみ残した結果を
contigs_300.faというファイル名で出力

```
File Edit View Search Terminal Help [ 8:30午後 ]
iu@bielinux[result] pwd
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls -d hoge_*
hoge_111 hoge_121 hoge_131 hoge_151 hoge_171 hoge_191 [ 8:30午後 ]
iu@bielinux[result] ls hoge_111 [ 8:30午後 ]
contigs.fa LastGraph PreGraph Sequences
Graph Log Roadmaps stats.txt
iu@bielinux[result] fastaLengthFilter.py hoge_111/contigs.fa 3
00 > hoge_111/contigs_300.fa
iu@bielinux[result] ls hoge_111 [ 8:30午後 ]
contigs_300.fa Graph Log Roadmaps stats.txt
contigs.fa LastGraph PreGraph Sequences
iu@bielinux[result] [ 8:30午後 ]
```

W12-8: 実行

```
iu@bielinux[result] pwd [ 8:30午後]
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls -d hoge_*
hoge_111 hoge_121 hoge_131 hoge_151 hoge_171 hoge_191 [ 8:30午後]
iu@bielinux[result] ls hoge_111 [ 8:30午後]
contigs.fa LastGraph PreGraph Sequences
Graph Log Roadmaps stats.txt
iu@bielinux[result] fastaLengthFilter.py hoge_111/contigs.fa 3
00 > hoge_111/contigs_300.fa
iu@bielinux[result] ls hoge_111 [ 8:30午後]
contigs_300.fa Graph Log Roadmaps stats.txt
contigs.fa LastGraph PreGraph Sequences
iu@bielinux[result] wc hoge_111/contigs_300.fa [ 8:30午後]
 3874 3874 840668 hoge_111/contigs_300.fa
iu@bielinux[result] grep -c ">" hoge_111/contigs_300.fa
1937
iu@bielinux[result] grep -v ">" hoge_111/contigs_300.fa | wc
 1937 1937 813550
iu@bielinux[result] [ 8:33午後]
iu@bielinux[result] [ 8:33午後]
```

①行数が127,260行から3,874行に激減している!
②配列数も23,761個から1,937個に激減。
③総塩基数も5,718,204 bpから(813,550 – 1,937
=) 811,613 bpに激減していることがわかる



④の手順でも同じ結果になることを確認しており、プログラムのバグではない

W12-8: 実行

```
File Edit View Search Terminal Help [ 8:30 午後 ]  
iu@bielinux[result] pwd  
/home/iu/Documents/DRR024501/result  
iu@bielinux[result] ls -d hoge_* [ 8:30 午後 ]  
hoge_111 hoge_121 hoge_131 hoge_151 hoge_171 hoge_191  
iu@bielinux[result] ls hoge_111 [ 8:30 午後 ]  
contigs.fa LastGraph PreGraph Sequences  
Graph Log  
iu@bielinux[re...  
00 > hoge_111/ ~NGS、RNA-seq、ゲノム、トランскриプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～  
(last modified 2015/12/22, since 2011)  
contigs_300.fa  
contigs.fa  
iu@bielinux[re...  
3874 3874 8 [ 8:33 午後 ]  
iu@bielinux[re...  
1937 [ 8:33 午後 ]  
iu@bielinux[result]  
1937 1937 81 [ 8:33 午後 ]  
iu@bielinux[result]  
iu@bielinux[result]
```

(Rで)塩基配列解析
～NGS、RNA-seq、ゲノム、トランскриプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～
(last modified 2015/12/22, since 2011)

What's new?

- ① • この Rと関連する多群
- ② • 前処理 | フィルタリング | [ACGT以外のcharacter "-"をNに変換](#) (last modified 2013/06/18)
- ③ • 前処理 | フィルタリング | [ACGT以外の文字数が閾値以下の配列を抽出](#) (last modified 2015/09/12)
- ④ • 前処理 | フィルタリング | [重複のない配列セットを作成](#) (last modified 2013/06/18)
- 前処理 | フィルタリング | [指定した長さ以上の配列を抽出](#) (last modified 2014/02/07)
- 前処理 | フィルタリング | [任意のリード\(サブセット\)を抽出](#) (last modified 2014/08/21)
- 前処理 | フィルタリング | [指定した長さの範囲の配列を抽出](#) (last modified 2015/02/26)
- 前処理 | フィルタリング | [任意のIDを含む配列を抽出](#) (last modified 2013/06/18)

W12-8: 実行

```
File Edit View Search Terminal Help
iu@bielinux[result] pwd
/home/iu/Documents/DRR024501/result
iu@bielinux[result] wc hoge_111/contigs_300.fa      [ 8:56 午後 ]
 3874 3874 840668 hoge_111/contigs_300.fa
iu@bielinux[result] grep -c ">" hoge_111/contigs_300.fa
1937
iu@bielinux[result] grep -v ">" hoge_111/contigs_300.fa | wc
 1937 1937 813550
iu@bielinux[result] tail -n 4 hoge_111/contigs_300.fa
>sequence1936
ATAATACGTTAACTTGCTTCAGCACTTCTTCACCGCTGAATAAACATTACCTAAGAACCT
CGCAAGAAAGTATTATCACTTCTTACTAGCAAACGTACGTAACATTCCACAATACTG
ATGTCTTCAGTAAATTCTGGTGTAGATCCTGGGCATGTAAGCGCGGGTAGTAGTAACGCC
CCAGTTAACAGTACCGGTATCTGGTTTACATTACCAGCAATACTGCATTAAAAGACTAG
TAGTTGCATCGAACGAGAACGAGGGCCGTTTATCGCCTGGATGGAGAAT
>sequence1937
CTAACCGATAAAACTAAAGTTAAAAATATACAACCTTGCTAGTATAACAGTATTATCAAT
AAATCGCAACTGTAACGCTTAAATAGTCTATTAAACTTCCAGCACTGTTCTAAATGAA
TTCCCATGACTTGCTTCAATAATACAATGTCATCATTGTAAGGCTATCTGCAGTTGTGCG
GTCTTTGTTCTTTGTTCAATATCAAATAATGTAATGCATCTGCCGAATATTATTTG
TAGCTCATCTCGTAAAGCCTTCATATGGGGCCAATTAAAAACACGGATTGA
iu@bielinux[result] [ 8:56 午後 ]
```

①行数が3,874行、②配列数が1,937個に激減。③description行以外の行数も1,937行、という結果に違和感を覚えるかもしれない。これはfastaLengthFilter.pyの出力ポリシーは、塩基配列部分が1配列1行だからである。実際、④最後の4行を出力させると、2配列分表示される。

W12-9: 全て実行

File Edit View Search Terminal Help

```
iu@bielinux[result] pwd  
/home/iu/Documents/DRR024501/result  
iu@bielinux[result] ls -d hoge_*  
hoge_111  hoge_121  hoge_131  hoge_151  hoge_171  hoge_191  
iu@bielinux[result] fastaLengthFilter.py hoge_121/contigs.fa 3  
00 > hoge_121/contigs_300.fa  
iu@bielinux[result] fastaLengthFilter.py hoge_131/contigs.fa 3  
00 > hoge_131/contigs_300.fa  
iu@bielinux[result] fastaLengthFilter.py hoge_151/contigs.fa 3  
00 > hoge_151/contigs_300.fa  
iu@bielinux[result] fastaLengthFilter.py hoge_171/contigs.fa 3  
00 > hoge_171/contigs_300.fa  
iu@bielinux[result] fastaLengthFilter.py hoge_191/contigs.fa 3  
00 > hoge_191/contigs_300.fa  
iu@bielinux[result]  
iu@bielinux[result]
```

同じ作業を、他のk値のアセンブリ結果に対しても実行。ないヒトはやったつもりでエーアーハンズオン。入力ファイルがあるディレクトリをカレントディレクトリにしなかった理由は、複数のディレクトリ上にあるファイルを一気に処理できるようにしたかったからです



[9:09 午後]
[9:09 午後]

フィルタリング結果ファイルに対して
、①行数や②ファイルサイズを表示
。ないヒトはエアーハンズオン

W12-9：結果を確認

```
File Edit View Search Terminal Help [ 9:11 午後 ]
iu@bielinux[result] pwd
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls -d hoge_*
hoge_111 hoge_121 hoge_131 hoge_151 hoge_171 hoge_191 [ 9:11 午後 ]
iu@bielinux[result] grep -v ">" hoge_121/contigs_300.fa | wc
2942 2942 1566326
iu@bielinux[result] grep -v ">" hoge_131/contigs_300.fa | wc
2449 2449 2153849
iu@bielinux[result] grep -v ">" hoge_151/contigs_300.fa | wc
1306 1306 2600683
iu@bielinux[result] grep -v ">" hoge_171/contigs_300.fa | wc
168 168 2381691
iu@bielinux[result] grep -v ">" hoge_191/contigs_300.fa | wc
336 336 2405767
iu@bielinux[result] [ 9:11 午後 ]
```

①

②

W12-9: 結果を確認

- 全アセンブル結果に対して実行[W12-9]

300 bp以上の配列のみ残した結果をcontigs_300.faというファイル名で出力。

```
pwd  
ls -d hoge_*
```

```
fastaLengthFilter.py hoge_121/contigs.fa 300 > hoge_121/contigs_300.fa  
fastaLengthFilter.py hoge_131/contigs.fa 300 > hoge_131/contigs_300.fa  
fastaLengthFilter.py hoge_151/contigs.fa 300 > hoge_151/contigs_300.fa  
fastaLengthFilter.py hoge_171/contigs.fa 300 > hoge_171/contigs_300.fa  
fastaLengthFilter.py hoge_191/contigs.fa 300 > hoge_191/contigs_300.fa
```

```
pwd  
ls -d hoge_*
```

```
grep -v ">" hoge_121/contigs_300.fa | wc  
grep -v ">" hoge_131/contigs_300.fa | wc  
grep -v ">" hoge_151/contigs_300.fa | wc  
grep -v ">" hoge_171/contigs_300.fa | wc  
grep -v ">" hoge_191/contigs_300.fa | wc
```

実際に行っているのは、赤枠内のコピペ。系統的に記述できている(違いはディレクトリ名部分のみ)ことがわかる。赤枠部分のみからなるファイルは、一般的なシェルスクリプトといえる。私はこういうものを作つて一気に系統的な解析を行います

W12-10:これまでのまとめ

(a) フィルタリング前

k-mer	コンティグ数	総塩基数	ウェブ資料
31	29502	4077679	W10-4
61	15445	3886574	W10-4
91	8583	3412266	W10-4
111	23761	5718204	W10-5
121	15776	4690144	W10-5
131	8398	3710829	W10-5
151	1306	2599377	W10-5
171	168	2381523	W10-5
181	198	2386048	W10-1
191	336	2405431	W10-5

(b) フィルタリング後

コンティグ数	総塩基数	ウェブ
2942	1563384	W12-9
2449	2151400	W12-9
1306	2599377	W12-9
168	2381523	W12-9
336	2405431	W12-9

配列長(< 300 bp)によるフィルタリング前は、最大で約5.7MB (k=111でのVelvetアセンブリ結果)というゲノムサイズに達していたが、フィルタリング後は最大でも約2.6MB (k=151の結果)となっていることがわかる

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



DDBJ Pipeline

(a) フィルタリング前

k-mer	コンティグ数	総塩基数	ウェブ資料
31	29502	4077679	W10-4
61	15445	3886574	W10-4
91	8583	3412266	W10-4
111	23761	5718204	W10-5
121	15776	4690144	W10-5
131	8398	3710829	W10-5
151	1306	2599377	W10-5
171	168	2381523	W10-5
181	198	2386048	W10-1
191	336	2405431	W10-5

(b) フィルタリング後

コンティグ数	総塩基数	ウェブ資料
--------	------	-------

2942	1563384	W12-9
2449	2151400	W12-9
1306	2599377	W12-9
168	2381523	W12-9
336	2405431	W12-9

DDBJ PipelineでVelvetが利用可能である。①実際にk=131で`de novo`アセンブリを行う一連の手順を示し、Bio-Linux上の数値(W10-5やW12-9)と同じ結果が得られて安心するところまでがW13からW18の内容。ダイジェストで紹介

①

第6回原稿PDFのp47

DDBJ Pipeline (概要からアカウント作成まで)

これまで行ってきた解析結果は、全データの約1/10のリード数からなるサブセットについてのものである。乳酸菌を含むバクテリア程度であれば、オリジナルデータを用いた *de novo* アセンブリも実行可能かもしれない。どの程度のデータ量まで手元のPCで可能か？どの程度メモリを積んだノートPCであればこのデータ量の解析が可能か？といったデータ量やスペックに関するグレーゾーンの議論はここでは行わない。計200万リードからなる paired-end RNA-seq データの *de novo* transcriptome assembly が2GBメモリでできた（第5回の W5-2）こと、本稿で示した合計約60万リードからなる paired-end データの Velvet アセンブリがマニュアル通りのやり方でできたことなど、実体験の積み重ねのほうが有意義であろう。

データ量が大きな配列解析を行う手段の1つは、国立遺伝学研究所（以下、遺伝研）が運用するスーパーコンピュータシステム（以下、スパコン）¹⁸⁾の利用である。本連載で何度か紹介した DDBJ Read Annotation Pipeline（以下、DDBJ Pipeline）³⁾は、遺伝研・大量遺伝情報研究室において開発・運用されている NGS 解析に特化した遺伝研スパコンを遠隔利用できるクラウドウェブサービスの1つで

①

かには、DDBJ Pipeline インポートまたはアップロードで遺伝研スパコンに設置する作業と読み替えればよい。この作業には、3通りのやり方が存在する：

- ① DRA/ERA/SRA から始まる ID の指定（DRAからのインポート）
- ② FTP 経由でアップロード
- ③ HTTP 経由でアップロード

公共 DB で公開されている NGS データを解析したい場合は、①で DRA ID を与えればよい。但し、本稿で取り扱っている DRR024501 という ID は、DRR～であり DRA～ではない。つまり DRR024501 を与えてもエラーとなるため、DRR024501 を頼りに公共 DB を眺めて指定可能な DRA ID（この場合は DRA002643）を探す必要がある [W2; W13-3]。手元のファイルを解析したい場合は、② FTP 経由か③ HTTP 経由でアップロードする。FTP 経由のアップロードは、FTP クライアントソフトウェア（以下、FTP ソフト）を利用する [W13-6]。WinSCP（Windows 用）や Cyberduck（Macintosh 用）がホスト OS 上にインストールされていれば、マニュアルの指示通りに設定情報



②

W13-1 : DDBJ pipeline

スライドを見るだけ



DDBJ Pipeline(概要からアカウント作成まで)

- 国立遺伝学研究所のスーパーコンピュータシステム
 - 引用文献(たぶんこれが最新論文): [Mashima et al., Nucleic Acids Res., 2016](#)
 - 引用文献(web上で指定された公式論文): [Ogasawara et al., Nucleic Acids Res., 2013](#)
 - [DDBJ Read Annotation Pipeline](#) (DDBJ Pipeline): [Nagasaki et al., DNA Res., 2013](#)
 - 遺伝研・大量遺伝情報研究室
- Galaxy: [Goecks et al., Genome Biol., 2010](#)
- TRAPLINE: [Wolfien et al., BMC Bioinformatics, 2016](#)
- Orione: [Cuccuru et al., Bioinformatics, 2014](#)
- 遺伝研スパコンの[登録ユーザ情報](#)

①

DDBJ Pipeline(クエリファイルの登録)

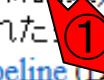
- 手元のファイル[W13-4]
W5-4で作成した解析したい手元のファイルを共有フォルダ (~/Desktop/mac_share) 経由でホストOSにコピー。コピーするファイルは、[QC.1.trimmed.fastq.gz](#)(57,061,392 bytes; 約55MB)と[QC.2.trimmed.fastq.gz](#)(62,989,289 bytes; 約61MB)です。ここでは[QC.\[0-9\].*.gz](#)で2つのファイルを指定しています。

```
cd ~/Documents/DRR024501/result  
  
pwd  
ls  
cp QC.[0-9].*.gz ~/Desktop/mac_share
```

W13-1 : DDBJ pipeline

DDBJ Pipeline(概要からアカウント作成まで)

- ・国立遺伝学研究所のスーパーコンピュータシステム
 - 引用文献(たぶんこれが最新論文): [Mashima et al., Nucleic Acids Res., 2016](#)
 - 引用文献(web上で指定された)論文): [Ogasawara et al., Nucleic Acids Res., 2013](#)
 - [DDBJ Read Annotation Pipeline](#) (DDBJ Pipeline): [Nagasaki et al., DNA Res., 2013](#)
 - 遺伝研・大量遺伝情報研究室
- ・ [Galaxy](#): [Goecks et al., Genome Biol., 2010](#)
- ・ [TRAPLINE](#): [Wolfien et al., BMC Bioinformatics, 2016](#)
- ・ [Orione](#): [Cuccuru et al., Bioinformatics, 2014](#)
- ・ 遺伝研スパコンの[登録ユーザ情報](#)



①DDBJ Pipeline利用時には、アカウントを作成しておかねばなりません。その際に入力する「氏名・所属・利用目的」は、②遺伝研スパコンのウェブサイト上で公開されるのでご注意ください

スライドを見るだけ

DDBJ Pipeline(クエリファイルの登録)

- ・手元のファイル[W13-4]
W5-4で作成した解析したい手元のファイルを共有フォルダ (~/Desktop/mac_share) 経由でホストOSにコピー。コピーするファイルは、[QC.1.trimmed.fastq.gz](#)(57,061,392 bytes; 約55MB)と[QC.2.trimmed.fastq.gz](#)(62,989,289 bytes; 約61MB)です。ここでは[QC.\[0-9\].*.gz](#)で2つのファイルを指定しています。

```
cd ~/Documents/DRR024501/result  
  
pwd  
ls  
cp QC.[0-9].*.gz ~/Desktop/mac_share
```

W13-1 : DDBJ pipeline



つまり、①DDBJ Pipelineの新規アカウント作成時の登録情報(氏名・所属・利用目的)は、遺伝研スパコンの②登録ユーザ情報の③のところで公開される。企業の方は、受託解析に使うことはできません。研究開発用途としてお使いください

スライドを見るだけ

This screenshot shows the "登録ユーザ情報 (2012年3月1日～現在)" (Registration User Information) page from the NIG Supercomputer Facilities. The page includes the NIG logo, the text "大学共同利用法人 情報・システム研究機構 国立遺伝学研究所 スーパーコンピュータシステム SuperComputer Facilities of National Institute of Genetics", and a search bar. A red arrow labeled ② points to the "ユーザ登録数一覧" (List of Registered Users) link. Another red arrow labeled ③ points to the "DDBJ pipelineユーザ" (DDBJ pipeline user) row in the table below. The table provides a breakdown of registered users by category:

ユーザ分類	外部ユーザ数	遺伝研所属ユーザ数	総ユーザ数
一般研究ユーザ	477	45	522
一般研究ユーザ - 大規模	105	35	140
Webサービスユーザ	658	12	670
DDBJ pipelineユーザ	919	43	962
業務ユーザ	0	83	83
スーパーコンピュータ管理者	0	15	15
全ユーザ合計数	2159	233	2392

W13-4: 手元のファイル

(a) フィルタリング前

(b) フィルタリング後

k-mer	コンティグ数	総塩基数	ウェブ資料
31	29502	4077679	W10-4
61	15445	3886574	W10-4
91	8583	3412266	W10-4
111	23761	5718204	W10-5
121	15776	4690144	W10-5
131	8398	3710829	W10-5
151	1306	2599377	W10-5
171	168	2381523	W10-5
181	198	2386048	W10-1
191	336	2	

コンティグ数	総塩基数
2942	1563384
2449	2151400
1306	2599377
168	2381523

おさらい。①Bio-Linuxで行ったVelvetの入力ファイルは、②のFaQCs実行結果ファイル。これをDDBJ Pipeline上にFTP経由でアップロードしてVelvetを実行しようとしている

スライドを見るだけ

①

```

iu@bielinux[result] pwd
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls
fastqCount.txt  hoge_171          QC_qc_report.pdf
hoge_111        hoge_191          QC.stats.txt
hoge_121        list.txt          QC.unpaired.trimmed.fastq
hoge_131        QC.1.trimmed.fastq.gz
hoge_151        QC.2.trimmed.fastq.gz
iu@bielinux[result]

```

[2:39 午後]

[2:39 午後]

[2:39 午後]

W13-4: ファイル名に注意

バイオインフォマティクス分野の常識・非常識

①

コマンドライン環境初心者がよく犯すミスは、適切な場所への半角スペースの入れ忘れである。例えば、「ls -a」と打っているつもりで「ls-a」と打つと、ls と -a の間に半角スペースが適切に挿入されていないため、ls-a というコマンドとして認識される。当然のことながら「そのようなコマンドはない (command not found)」といわれる [ウェブ資料 13]。この程度であれば明確にエラーと認識できるので対処のしようはある。コマンドライン環境では、半角スペースは明確な意味を持つ場合が多い。それゆえ、NGS データ解析に限らず、解析したいファイルを入力として与える際にファイル名の中にスペースを入れるのは、コマンドライン環境中心のバイオインフォマティクスの世界では非常識である。他に「全角文字」や「[, #, \$などの英数字以外の文字」も忌避される。一般に多数のファイルが 1 つのディレクトリ内に存在する場合、意味を持たせたファイル名にすることが多い。例えば、group1_rep1.fa, group1_rep2.fa, group2_rep1.fa, group2_rep2.fa のような具合である。この場合、上記のようなブラックリストを眺めるのではなく、使っても大丈夫という経験に基づくホワイトリストを利用するほうが手っ取り早い。例えば、著者らは主に「xxx_yy_zzzz_001.fa」のような英数字とアンダースコア (_) の組合せを利用する。

連載第2回原稿の①「バイオインフォマティクス分野の常識・非常識」を再掲。非常識なファイル名で実行を試みて「うまくいかないんですけど」的な質問は ×

スライドを見るだけ

ファイルの拡張子にも気をつけたほうがよい。 上記 FASTA ファイルの拡張子は .fa であったが、.fasta でもよい。拡張子が .fa か .fasta であれば、常識的な範囲で他はなんでもよいという意味合いで *.fa や *.fasta という表現もなされる。塩基ごとのクオリティ情報を含む FASTQ 形式ファイル (以下、FASTQ ファイル) の拡張子は、*.fq または *.fastq が一般的である。もちろんそれ以外の拡張子でも受け入れてくれる NGS 解析用プログラムは存在するかもしれない。例えば、オプションで FASTQ 形式だということを宣言さえしておけば、実際の拡張子は .txt でも .doc でも .pdf でも受け入れてくれるかもしれない。しかし著者らは、*.doc や *.pdf での動作確認 (ブラックリスト作成) には関与しない。無難な *.fq や *.fastq を素直に利用する。

W15-2: Select Tools

デフォルトはマッピングになっているので、①de novo Assemblyにチェックを入れて、②ページ下部に移動

The screenshot shows the 'Selecting Tools for Basic Analysis of DDBJ ANNOTATION PIPELINE'. The sidebar on the left lists various analysis steps and job statuses. The main panel displays two tables of tools for genome mapping and de novo assembly.

Reference Genome Mapping:

Tool	Help	Version	Input data		Evaluation			Analysis		Output format			Comment
			Base space	Color space	Paired end	Depth	Coverage	Error rate	SNP	Indel	.gff	.bed	
BLAT		34	✓					✓					Single-end analysis only
bwa		0.6.1	✓		✓	✓	✓	✓			✓		
Bowtie		0.12.7	✓	✓	✓	✓	✓	✓	✓	✓			
TopHat		1.0.11	✓		✓	✓	✓	✓	✓			✓	
Bowtie2		2.0.0	✓	✓	✓	✓	✓	✓	✓	✓	✓		For reads longer than about 50 bp, Bowtie2 is generally faster, more sensitive, and uses less memory than Bowtie1.
TopHat2		2.0.9	✓		✓	✓	✓	✓	✓			✓	

de novo Assembly:

Tool	Help	Version	Base space	Color space	Paired-end	MSS (WGS)	Comment
SOAPdenovo		1.05	✓		✓		
ABySS		1.3.2	✓		✓		Maximum K-mer value is 64.
Velvet		1.2.10	✓		✓	✓	We severely recommend when performing Velvet, total length of those reads is up to 22G bp. Maximum K-mer value is 64.

W15-2: Select Tools

Preprocessing Start

step-1

- Preprocessing
- Mapping / de novo Assembly

step-2

- Workflow
 - Genome (SNP/Short Indel)
 - RNA-seq (Tag count)
 - ChIP-seq

JOB STATUS

step1.

- Preprocessing

step1.

- Mapping

step1.

- de novo Assembly

step2-All status

HELP

- HELP
- TUTORIAL
- Contact Us.
 - DDJB Read Annotation Pipeline.
 - Development Team.

① [Velvet](#)

② [NEXT](#)

Selecting Tools for Basic...

	Tool	Help	Version	Base space	Color space	Paired-end	MSS (WGS)	Comment
<input type="checkbox"/>	SOAPdenovo		1.05	✓		✓		
<input type="checkbox"/>	ABYSS		1.3.2	✓		✓		Maximum K-mer value is 64.
<input checked="" type="checkbox"/>	Velvet		1.2.10	✓		✓	✓	We severe recommend when performing Velvet, total length of those reads is up to 22G bp. Maximum K-mer value is 64.
<input type="checkbox"/>	Trinity		r2013-02-25	✓		✓		RNA-Seq De novo Assembly
<input type="checkbox"/>	Platanus		1.2.2	✓		✓		
<input type="checkbox"/>	HGAP		Protocol3 (v 2.2.0)					HGAP Pipeline for PacBio Sequence based on SMRT Analysis v2.2.0. For bax.h5 file only. (Beta version)

Mapping Contigs by de novo Assemble to Reference Sequences.
The contigs will be aligned to reference genome.

Tool	Comment
<input checked="" type="radio"/> BLAT	Single-end analysis only

W15-3: Set QuerySet

Generating Query Sets from Query Read Files

Paired-end analysis
Layout of paired sequence. 5'-3' 3'-5'

5'	Linker(1)	Target	3'	Linker(2)	Linker(3)	3'	Target	5'	Linker(4)
	Run ACCESSION	Read length	Quality Score						
①	<input checked="" type="checkbox"/> QC.1.trimmed.fastq.gz	bp		②				③	

QUERY SET

RESET BACK NEXT

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly
- step2-All status

HELP

HELP TUTORIAL

Contact Us.

①解析したいデータにチェックを入れて、②Set as Pair-End。③NEXT。本来この画面は、複数のクエリファイルを使用する場合に、それらを連結して1つのクエリセットとしてジョブを実行するか、別々のクエリセットとして並列して実行するかを指定する画面である。今回はクエリファイル1つを単独で使用するので、この画面にはあまり意味がない

灰色は読まない

W15-4 : Set Ass.Options

Setting for De Novo Assembly

velvet

Set optional parameters of the paired-end analysis

Step1) Convert sequences

Shuffle the sequence. perl shuffleSequence query_1.fastq query_2.fastq shuffle_query_pe.fastq

Running velvet. Velvet output_directory/ **① 23 -fastq** -shortPaired shuffle_query_pe.fastq

Step2) Assembly

Velvetg output_directory/ **② -ins_length 300 -exp_cov auto**

Step3) Set parameters of the CONFIG mapping tool

Step4) Create assembled sequences in FASTA file from pileupped reads to [submit WGS division of DDBJ](#).

Set filtered length for contigs
 perl lengthfilter.pl pileupFile **④ 100** out_WGS.txt

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly**
- step2-All status

HELP

HELP

TUTORIAL

Contact Us.
DOBJ Read Annotation Pipeline.

W7-3のBio-Linux上での実行経験から、DDBJ Pipelineでは、①k=23がデフォルトなのだろう。W7-8のvelvetg実行時は特にオプションを指定しなかったが、②で示されているようなオプションもあるのだと学ぶ。③長さによる配列のフィルタリング(④デフォルトは100 bp)もやってくれるようだ

W15-4 : Set Ass.Options

W12-10(スライド72)を眺め、とりあえず
 ①k=131でBio-Linux上で指定したオプションと同じにして(②~④)実行。⑤NEXT

Setting for De Novo Assembly

velvet

Set optional parameters of the paired-end analysis

Step1) Convert sequences

Shuffle the sequence.
perl shuffleSequence query_1.fastq query_2.fastq shuffle_query_pe.fastq

Running velvet.
Velvet output_directory/ **131 -fastq** -shortPaired shuffle_query_pe.fastq

Step2) Assembly

Velvetg output_directory/

Step3) Set parameters of the CONFIG mapping tool

Step4) Create assembled sequences in FASTA file from pileupped reads to [submit WGS division of DDBJ](#).

Set filtered length for contigs
 perl lengthfilter.pl pileupFile 300 out_WGS.txt

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. **de novo Assembly**
- step2-All status

HELP

TUTORIAL

Contact Us.
DOBJ Read Annotation Pipeline.

BACK NEXT

W15-4 : Set Ass.Options

The screenshot shows the DDBJ Pipeline interface for setting assembly options. The main title is "Setting for De Novo Assembly". The navigation bar at the top includes steps: Select Query Files, Select Tools, Set QuerySet, Set Ass. Options (highlighted in orange), Confirmation, and Running Status. On the left, there's a sidebar with sections for ACCOUNT (login ID [agribio], Logout, Change password), ANALYSIS (Data setup, DRA Start, FTP upload, HTTP upload, DRA Import, Preprocessing Start, step-1, Preprocessing, Mapping / de novo Assembly, step-2, Workflow (Genome (SNP/Short Indel), RNA-seq (Tag count), ChIP-seq)), JOB STATUS (step1. Preprocessing, step1. Mapping, step1. de novo Assembly, step2-All status), and HELP (HELP, TUTORIAL, Contact Us.). The main content area is titled "Setting for De Novo Assembly" and contains four steps: Step1) Convert sequences, Step2) Assembly, Step3) Set parameters of the CONFIG mapping tool, and Step4) Create assembled sequences in FASTA file from pileupped reads to submit WGS division of DDBJ. Step1 is highlighted with a red border. Step2 has a red arrow pointing to it from the top right. Step4 has a red arrow pointing to it from the bottom right.

W15-5: Confirmation

①アセンブリが終了したら、アカウント作成時に指定したアドレス宛にメールが送られる。②RUN。③OK

The screenshot shows the 'Run Confirmation' step of the pipeline. A red box highlights the 'Destination of mail' field where the email address 'kadota@bi.a.u-tokyo.ac.jp' is entered. Red arrows point from the numbered steps to specific UI elements: arrow 1 points to the 'Destination of mail' input field; arrow 2 points to the 'RUN' button; and arrow 3 points to the 'OK' button in a confirmation dialog.

Run Confirmation

Destination of mail
When the request is completed, the system sends an email to this address.
kadota@bi.a.u-tokyo.ac.jp * Required
Result files will be deleted 60 days after submission.

BACK RUN

Assembly [velvet]

Query sets
Query set1

PairedOrientation	RunAccession	RunAlias	RowLength	QualityScore1	QualityScore2
paired	20392	L.hokkaidonensis_MiSeq_denovo			

Assembly commands
velvet

Set optional parameters of the paired-end analysis

Step1) Convert sequences
Shuffle the sequence.
perl shuffleSequences_fastq.pl query_1.fastq query_2.fastq shuffle_query_pe.fastq
Running velvet.
Velvetg output_directory/ 131 -fastq -shortPaired shuffle_query_pe.fastq

Step2) Assembly
Velvetg output_directory/

Step3) Set parameters of the CONFIG mapping tool

Step4) Create assembled sequences in FASTA file from pileupped reads to [submit WGS division of DDBJ](#).

Web ページからのメッセージ
Do you really want to execute pipeline programs?
OK キャンセル

W16-1 : 計算終了

①DDBJ Pipelineから計算終了メールが届いたら、②DDBJ Pipelineに再度ログインして計算結果を眺める



2016/01/15 (金) 20:37

pipeline_team@g.nig.ac.jp

Job finished : DDBJ Read Annotation Pipeline



宛先 kadota@biau-tokyo.ac.jp

C C pipeline_report@g.nig.ac.jp

このメッセージから余分な改行を削除しました。

Dear agribio,

Your request to DDBJ pipeline service has finished.

Please visit the web site to obtain analytical results.

Request ID: 21119

URL: <https://p.ddbj.nig.ac.jp/>



If you have troubles in this service, please write to pipeline_dev@ddbj.nig.ac.jp

Thank you for trying our analytical service.

Regards,

DDBJ

W18-1: フィルタリング前の結果

The screenshot shows the DDBJ Pipeline Detail View interface. On the left, a sidebar lists various upload and import options. The main area displays assembly statistics and a command history table.

Assembly statistics:

		Contig # : 8,398
		Total contig size : 3,710,829
		Maximum contig size : 6,572
		Minimum contig size : 261
		N50 contig size : 506

Time:

Wait time	Start time	End time
0: 0:17	2016-01-15 18:52:36	2016-01-15 20:36:28

Command History:

Command	Start time	End time	Log1	Log2	Result	MD5
perl shuffleSequences_fastq.pl QC.1.trimmed.fastq QC.2.trimmed.fastq QC_pe.fastq	2016-01-15 18:52:36	2016-01-15 18:52:46				
velveth outputDir/ 131 -fastq -shortPaired /home/w3pipeline/refdata/tmp/agribio/21119/20926/unspecified/QC_pe.fastq	2016-01-15 18:52:46	2016-01-15 18:53:28	View		Download(66.5 MB)	MD5
velvetg outputDir/	2016-01-15 18:54:22	2016-01-15 20:35:14	View		Download(66.5 MB)	

Top of page

ページ下部に移動した後の状態。①「Download(66.5MB)」をクリックすると、velvet.zipというzip圧縮ファイルをダウンロードできる。共有フォルダに保存してBio-Linux上で眺める。とオリジナル(第6回ウェブ資料)はなっているが、講習会では、~/Desktop/backup上にあるvelvet.zipを取り扱うので、以降の内容は若干異なる

W18-2: Bio-Linuxで解凍

```
File Edit View Search Terminal Help  
iu@bielinux[backup] pwd  
/home/iu/Desktop/backup  
iu@bielinux[backup] ls -l velvet.*  
-rw-rw-r-- 1 iu iu 66517438 1月 18 14:01 velvet.zip  
iu@bielinux[backup] unzip velvet.zip  
Archive: velvet.zip  
  creating: velvet/  
  inflating: velvet/Log  
  inflating: velvet/Sequences  
  inflating: velvet/Roadmaps  
  inflating: velvet/PreGraph  
  inflating: velvet/Graph  
  inflating: velvet/contigs.fa  
  inflating: velvet/stats.txt  
  inflating: velvet/LastGraph  
iu@bielinux[backup]
```

①~/Desktop/backupにある、②velvet.zipの存在確認。③unzipで解凍。④velvetフォルダ中のcontigs.faがVelvetの生の出力ファイル。このあたりは手打ちでやってください

[4:38 午後]

[4:38 午後]

[4:39 午後]

W18-3: 行数とコンティグ数

①velvetディレクトリに移動。
contigs.faの②行数は75,235、③
配列(コンティグ)数は8,398個。
このあたりも手打ち

```
iu@bielinux[backup] pwd [ 4:45 午後]
/home/iu/Desktop/backup
iu@bielinux[backup] cd velvet [ 4:45 午後]
iu@bielinux[velvet] pwd [ 4:46 午後]
/home/iu/Desktop/backup/velvet
iu@bielinux[velvet] ls [ 4:46 午後]
contigs.fa LastGraph PreGraph Sequences
Graph Log Roadmaps stats.txt
iu@bielinux[velvet] wc contigs.fa [ 4:46 午後]
 75235 75235 4077589 contigs.fa
iu@bielinux[velvet] grep -c ">" contigs.fa [ 4:46 午後]
8398
iu@bielinux[velvet] [ 4:46 午後]
```

W18-3: 行数とコンティグ数

```
iu@bielinux[backup] pwd  
/home/iu/Desktop/backup  
iu@bielinux[backup] cd velvet  
iu@bielinux[velvet] pwd  
/home/iu/Desktop/backup/velvet  
iu@bielinux[velvet] ls  
contigs.fa LastGraph PreGraph Sequences  
Graph Log Roadmaps stats.txt  
iu@bielinux[velvet] wc contigs.fa  
75235 75235 4077589 contigs.fa  
iu@bielinux[velvet] grep -c ">" contigs.fa  
8398  
iu@bielinux[velvet]
```

[4:45 午後]
[4:45 午後]
[4:46 午後]

①

②

③

④

Contig # : 8,398
al contig size : 3,710,829
Maximum contig size : 6,572
Minimum contig size : 261
N50 contig size : 506

W18-4: フィルタリング

iu@bielinux[~/Desktop/backup/velvet]

```
iu@bielinux[backup] pwd  
/home/iu/Desktop/backup  
iu@bielinux[backup] cd velvet  
iu@bielinux[velvet] pwd  
/home/iu/Desktop/backup/velvet  
iu@bielinux[velvet] ls  
contigs.fa LastGraph PreGraph Sequences  
Graph Log Roadmaps stats.txt  
iu@bielinux[velvet] wc contigs.fa  
 75235 75235 4077589 contigs.fa  
iu@bielinux[velvet] grep -c ">" contigs.fa  
8398  
iu@bielinux[velvet] fastaLengthFilter.py contigs.fa 300 > hoge.fa  
iu@bielinux[velvet] grep -c ">" hoge.fa  
2449  
iu@bielinux[velvet]
```

DDBJ Pipeline実行結果ファイルを入力として
、①指定した配列長閾値未満の配列を取り
除くPythonプログラムfastalengthFilter.pyを
実行。300 bp以上の配列は②2,449個となり
、Bio-Linux上で実行した結果と同じ(W12-9)

[4:45 午後]

[4:46 午後]

[4:46 午後]

[4:46 午後]

[4:46 午後]

[5:01 午後]

[5:02 午後]



①

②

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W19-2: Select Tools

Preprocessing Start

- step-1 Preprocessing
- Mapping / de novo Assembly
- step-2 Workflow
 - Genome (SNP/Short Indel)
 - RNA-seq (Tag count)
 - ChIP-seq

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly
- step2-All status

HELP

- HELP
- TUTORIAL
- Contact Us. DDBJ Read Annotation Pipeline. Development Team.

① 

Tool	Help	Version	Base space	Color space	Paired-end	MSS (WGS)	Comment
SOAPdenovo		1.05	✓		✓		
ABySS		1.3.2	✓		✓		Maximum K-mer value is 64.
Velvet		1.2.10	✓		✓	✓	We severe recommend when performing Velvet, total length of those reads is up to 22G bp. Maximum K-mer value is 64.
Trinity		r2013-02-25	✓		✓		RNA-Seq De novo Assembly
<input checked="" type="checkbox"/> Platanus		1.2.2	✓		✓		
HGAP		Protocol3 (v 2.2.0)					HGAP Pipeline for PacBio Sequence based on SMRT Analysis v2.2.0. For bax.h5 file only. (Beta version)

Mapping Contigs by de novo Assemble to Reference Sequences.
The contigs will be aligned to reference genome.

Tool	Comment
<input checked="" type="radio"/> BLAT	Single-end analysis only

② 

BACK **NEXT**

W19-4 : Set Ass. Options

Setting for De Novo Assembly

platanus

Set optional parameters of the paired-end analysis

Memory Usage : Low (recommended) High

If you request "High" memory usage during the time Nig super computer system is congested, you might be kept waiting long before job starts running.

Step1) Assembly : Construct contigs using the algorithm based on the de Bruijn graph.

```
platanus assemble -t 15 -m 120 -o out -f PE1.fastq PE2.fastq
```

Step2) Scaffold : Map paired-end (mate-pair) reads on contigs and construct scaffolds

```
platanus scaffold -t 8 -o out -c out_contig.fa -b out_contigBubble.fa -IP1 PE1.fastq PE2.fastq (-OP2 MP1.fastq MP2.fastq)
```

Step3) Gap Close : Map paired-end (mate-pair) reads on scaffolds and assemble reads on gaps and close gaps

```
platanus gap_close -t 8 -o out -c out_scaffold.fa -IP1 PE1.fastq PE2.fastq (-OP2 MP1.fastq MP2.fastq)
```

Step5) Create assembled sequences in FASTA file from pileupped reads to [submit WGS division of DDBJ](#).

Set filtered length for contigs

perl lengthfilter.pl pileupFile 300 out_WGS.txt

BACK **NEXT**

乳酸菌ゲノム決定論文中では、「*Platanus assembler ver 1.2 with the default settings*」と書かれている。赤枠のStep1, 2, 3の計3か所にオプションを指定する箇所があるが、とりあえずここは空白として…細かい点はすっ飛ばして①NEXT

W20-2: 結果を眺める

http://p.ddbj.nig.ac.jp/pipeline/DetailView.do?query_set_

ANALYSIS

- Data setup
- DRA Start
- FTP upload
- HTTP upload
- DRA Import
- Preprocessing Start

step-1

- Preprocessing
- Mapping / de novo Assembly

step-2

- Workflow
- Genome (SNP/Short Indel)
- RNA-seq (Tag count)
- ChIP-seq

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly
- step2-All status

HELP

- HELP
- TUTORIAL
- Contact Us.
- DDJB Read Annotation Pipeline Development Team.

Job info

ID	21211
Tool (Version)	Platanus (1.2.2)
RunAccession or Filename	QC.1.trimmed.fastq.gz
Download	QC.1.trimmed.fastq.gz
Read length	N.A. bp
Alias	L.hokka

Download modified queries

- [QC.1.trimmed.fastq.gz \(Original size 189.4 MB\)](#)
- [QC.2.trimmed.fastq.gz \(Original size 189.6 MB\)](#)

Download wgs file

- [out_WGS.fasta.gz \(Original size 2.3 MB\)](#)

Assembly statistics

Contig # : 117
Total contig size : 2,356,019
Maximum contig size : 257,728
Minimum contig size : 101
N50 contig size : 92,304

Time

Wait time	Start time	End time
0: 0.9	2016-01-20 18:33:36	2016-01-21 10:10:06

Command

Command	Start time	End time	Log1	Log2	Result	MD5
platanus assemble -m 120 -f QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-20 18:33:37	2016-01-21 10:08:51		View	Download(2.2 MB)	MD5
platanus scaffold -c out_contig.fa -b out_contigBubble.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:02	2016-01-21 10:09:12		View	Download(2.2 MB)	MD5
platanus gap_close -c out_scaffold.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:23	2016-01-21 10:09:34		View	Download(2.2 MB)	MD5

①

②

BACK

Top of page

①Platanusは、主に3ステップからなる(W19-4)が、そのコマンドの実体がわかる。②実行ログを眺めると、k=32, 42, 52などをいろいろ調べているのだろうと想像がつく。最後のほうで、k=117, 120, 121, 122, 123で終わっている。

Platanusはこれらのk値の結果を全て取り込んだアセンブリ結果を返すmulti-k genome assemblerのカテゴリーに属する。Velvetと違ってk値を指定するオプションが存在しないのは、k値の探索範囲も自動的に決められるから

W20-2: 結果を眺める

http://p.ddbj.nig.ac.jp/pipeline/DetailView.do?query_set_

ANALYSIS

- Data setup
- DRA Start
- FTP upload
- HTTP upload
- DRA Import
- Preprocessing Start

step-1

- Preprocessing
- Mapping / *de novo* Assembly

step-2

- Workflow
 - Genome (SNP/Short Indel)
 - RNA-seq (Tag count)
 - ChIP-seq

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. *de novo* Assembly
- step2-All status

HELP

- HELP □
- TUTORIAL
- Contact Us.
DDJB Read Annotation
Pipeline.
Development Team.

Job info

ID	21211
Tool (Version)	Platanus (1.2.2)

RunAccession or Filename	Download	Read length	Alias
QC.1.trimmed.fastq.gz	QC.1.trimmed.fastq.gz	N.A. bp	L.hokkaidonensis_MiSeq_denovo

Download modified queries

- [QC.1.trimmed.fastq.gz \(Original size 189.4 MB\)](#)
- [QC.2.trimmed.fastq.gz \(Original size 189.6 MB\)](#)

Download wgs file

- [out_WGS.fasta.gz \(Original size 2.3 MB\)](#)

Assembly statistics

Contig # : 117
Total contig size : 2,356,019
Maximum contig size : 257,728
Minimum contig size : 101
N50 contig size : 92,304

Time

Wait time	Start time	End time
0: 0.9	2016-01-20 18:33:36	2016-01-21 10:10:06

Command

Command	Start time	End time	Log1	Log2	Result	MD5	
platanus assemble -m 120 -f QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-20 18:33:37	2016-01-21 10:08:51			View	Download(2.2 MB)	MD5
platanus scaffold -c out_contig.fa -b out_contigBubble.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:02	2016-01-21 10:09:12			View	Download(2.2 MB)	MD5
platanus gap_close -c out_scaffold.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:23	2016-01-21 10:09:34			View	Download(2.2 MB)	MD5

①フィルタリング前のPlatanus実行結果ファイルは、ここからダウンロードできる。*(de novo*アセンブリの場合は)おそらくどのダウンロードボタンを押しても同じものが得られると思うが、②無難に最後のステップのところのものを選択。このあたりはプログラムごとに若干仕様が異なるため、統一的に見せるべくこのようになっているのだろう

W20-2: 結果を眺める

①Platanus実行結果の基本情報はここからみられる。パッと見、フィルタリング前の段階ですでにコンティグ数も明らかに少なくよさそう。赤枠部分を拡大して、Velvetの結果と比較

http://p.ddbj.nig.ac.jp/pipeline/DetailView.do?query_set_

ANALYSIS

- Data setup
- DRA Start
- FTP upload
- HTTP upload
- DRA Import
- Preprocessing Start

step-1

- Preprocessing
- Mapping / de novo Assembly

step-2

- Workflow**
 - Genome (SNP/Short Indel)
 - RNA-seq (Tag count)
 - ChIP-seq

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly
- step2-All status

HELP

- HELP
- TUTORIAL
- Contact Us.

Job info

ID	21211		
Tool (Version)	Platanus (1.2.2)		
RunAccession or Filename	Download	Read length	Alias
QC.1.trimmed.fastq.gz	QC.1.trimmed.fastq.gz	N.A. bp	L.hokkaidonensis_MiSeq_denovo

Download modified queries

- QC.1.trimmed.fastq.gz (Original size 189.4 MB)
- QC.2.trimmed.fastq.gz (Original size 189.6 MB)

Download wgs file

- out_WGS.fasta.gz (Original size 2.3 MB)

Assembly statistics

①

Contig # : 117
Total contig size : 2,356,019
Maximum contig size : 257,728
Minimum contig size : 101
N50 contig size : 92,304

Time

Wait time	Start time	End time
0: 0.9	2016-01-20 18:33:36	2016-01-21 10:10:06

Command

Command	Start time	End time	Log1	Log2	Result	MD5	
platanus assemble -m 120 -f QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-20 18:33:37	2016-01-21 10:08:51			View	Download(2.2 MB)	MD5
platanus scaffold -c out_contig.fa -b out_contigBubble.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:02	2016-01-21 10:09:12			View	Download(2.2 MB)	MD5
platanus gap_close -c out_scaffold.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:23	2016-01-21 10:09:34			View	Download(2.2 MB)	MD5

[BACK](#) [Top of page](#)

W20-3: Velvetと比較

k-mer	(a) フィルタリング前			(b) フィルタリング後		
	コンティグ数	総塩基数	ウェブ資料	コンティグ数	総塩基数	ウェブ資料
31	29502	4077679	W10-4	168	2381523	W10-5
61	15445	3886574	W10-4	2942	2381523	W12-5
91	8583	3412266	W10-4	2449	2151400	W12-9
111	23761	5718204	W10-5	1306	2599377	W12-9
121	15776	4690144	W10-5	168	2381523	W12-9
131	8398	3710829	W10-5	336	2405431	W12-9
151	1306	2599377	W10-5			
171	① 168	2381523	W10-5			
181	198	2386048	W10-1			
191	336	2405431	W10-5			

左表で調べた範囲のVelvet実行結果(W12-10)よりも、特にk値を意識すらせずに実行したPlatanusの結果のほうが明らかに優れていることがわかる。具体的には、Velvetでの最少配列数は①168個(k=171)であったが、②Platanusはフィルタリング前の状態で117個。そして、③Minimum contig sizeが101 bpであることから、300 bp未満でフィルタリングを行うと、間違いなく配列数が減ると予想する

Contig # : 117
 Total contig size : 2,356,019
 Maximum contig size : 257,728
 ③ Minimum contig size : 101
 N50 contig size : 92,304

W20-4:ダウンロード

http://p.ddbj.nig.ac.jp/pipeline/DetailView.do?query_set_ Detail view

ANALYSIS

- Data setup
- DRA Start
- FTP upload
- HTTP upload
- DRA Import
- Preprocessing Start

step-1

- Preprocessing
- Mapping / de novo Assembly

step-2

- Workflow
 - Genome (SNP/Short Indel)
 - RNA-seq (Tag count)
 - ChIP-seq

JOB STATUS

- step1. Preprocessing
- step1. Mapping
- step1. de novo Assembly
- step2-All status

HELP

- HELP
- TUTORIAL
- Contact Us.
 - DDJB Read Annotation Pipeline Development Team.

Job info

ID	21211		
Tool (Version)	Platanus (1.2.2)		
RunAccession or Filename	Download	Read length	Alias
QC.1.trimmed.fastq.gz	QC.1.trimmed.fastq.gz	N.A. bp	L.hokkaidonensis_MiSeq_denovo

Download modified queries

- QC.1.trimmed.fastq.gz (Original size 189.4 MB)
- QC.2.trimmed.fastq.gz (Original size 189.6 MB)

Download wgs file

- out_WGS.fasta.gz (Original size 2.3 MB)

Assembly statistics

Contig # : 117
Total contig size : 2,356,019
Maximum contig size : 257,728
Minimum contig size : 101
N50 contig size : 92,304

Time

Wait time	Start time	End time
0: 0.9	2016-01-20 18:33:36	2016-01-21 10:10:06

Command

Command	Start time	End time	Log1	Log2	Result	MD5
platanus assemble -m 120 -f QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-20 18:33:37	2016-01-21 10:08:51			View Download(2.2 MB)	MD5
platanus scaffold -c out_contig.fa -b out_contigBubble.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:02	2016-01-21 10:09:12			View Download(2.2 MB)	MD5
platanus gap_close -c out_scaffold.fa -IP1 QC.1.trimmed.fastq QC.2.trimmed.fastq	2016-01-21 10:09:23	2016-01-21 10:09:34			View Download(2.2 MB)	MD5

1

Top of page

①Platanus実行結果ファイル(platanusResult.zip)を、共有フォルダにダウンロード。とオリジナル(第6回ウェブ資料)はなっているが、講習会では、~/Desktop/backup上にあるplatanusResult.zipを取り扱うので、以降の内容は若干異なる

W20-5: Bio-Linuxで解凍

①~/Desktop/backupにある、②
platanusResult.zipの存在確認。こ
のあたりは手打ちでやってください

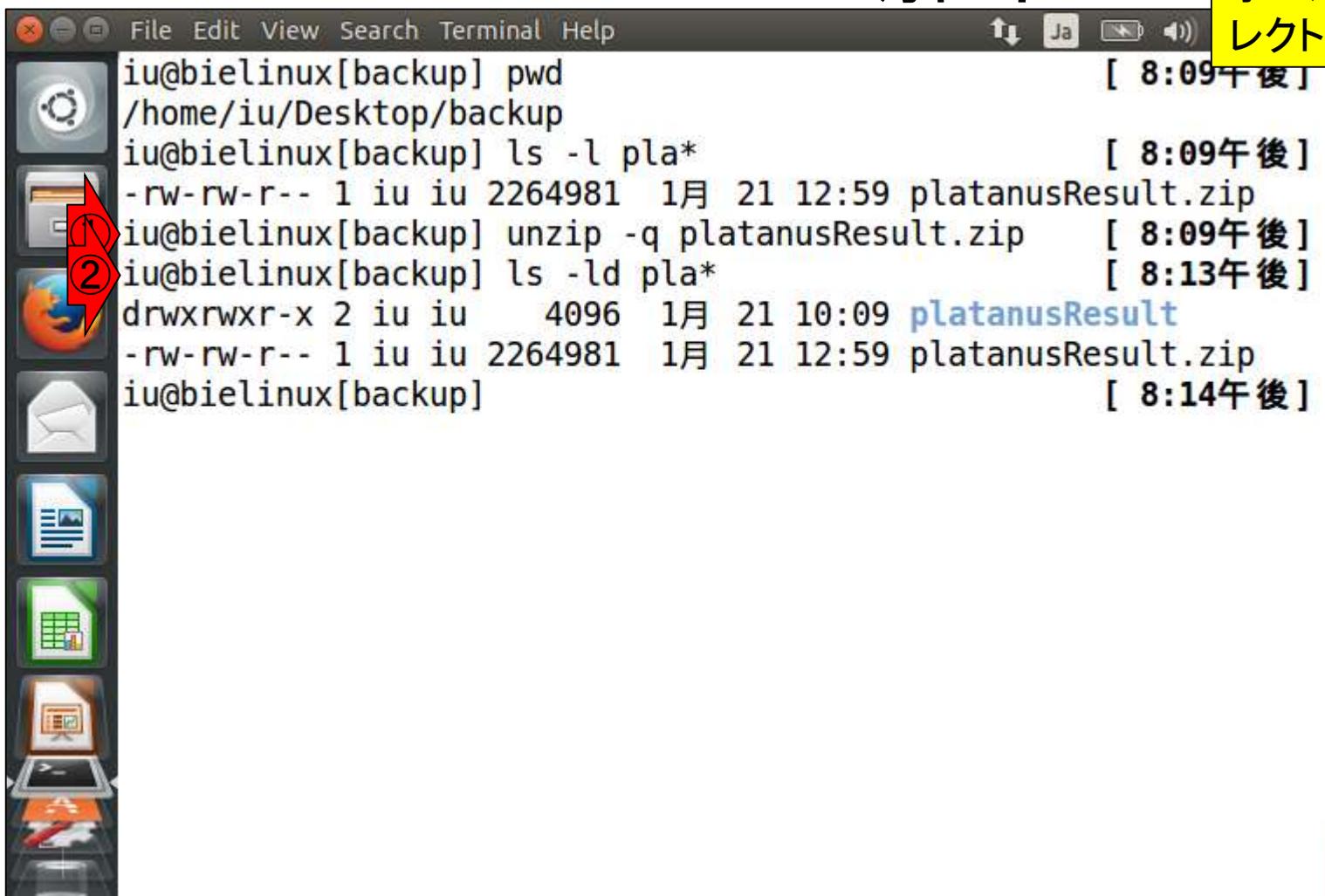
The screenshot shows a terminal window with a dark theme. On the left is a vertical dock with icons for various applications like a terminal, file manager, and browser. The terminal window has a title bar with 'File Edit View Search Terminal Help' and a system tray with icons for network, battery, and time ('20:10').

```
iu@bielinux[backup] pwd [ 8:09午後]
/home/iu/Desktop/backup
iu@bielinux[backup] ls -l pla*
-rw-rw-r-- 1 iu iu 2264981 1月 21 12:59 platanusResult.zip [ 8:09午後]
iu@bielinux[backup] [ 8:09午後]
```

Two red arrows point to specific elements: arrow ① points to the 'Desktop' icon in the dock, and arrow ② points to the 'platanusResult.zip' file listed in the terminal output.

W20-5: Bio-Linuxで解凍

①unzipコマンドで解凍。`-q`をつけて`quiet`で解凍。②`ls`で確認。`d`オプションをつけることで、ディレクトリの中身を表示させない



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "backup". The terminal content is as follows:

```
iu@bielinux[backup] pwd [ 8:09 午後]
/home/iu/Desktop/backup
iu@bielinux[backup] ls -l pla*
-rw-rw-r-- 1 iu iu 2264981 1月 21 12:59 platanusResult.zip
iu@bielinux[backup] unzip -q platanusResult.zip [ 8:09 午後]
iu@bielinux[backup] ls -ld pla* [ 8:13 午後]
drwxrwxr-x 2 iu iu 4096 1月 21 10:09 platanusResult
-rw-rw-r-- 1 iu iu 2264981 1月 21 12:59 platanusResult.zip
iu@bielinux[backup] [ 8:14 午後]
```

A red arrow points to the "dock" icon in the bottom-left corner of the desktop, which contains various application icons. A red circle with the number "2" is overlaid on the arrow.

W20-5: Bio-Linuxで解凍

①platanusResultディレクトリに移動し、②ls。Platanusの最終結果ファイルは③out_gapClosed.fa

```
iu@bielinux[backup] pwd [ 8:09 午後]
/home/iu/Desktop/backup
iu@bielinux[backup] ls -l pla*
-rw-rw-r-- 1 iu iu 2264981 1月 21 12:59 platanusResult.zip [ 8:09 午後]
iu@bielinux[backup] unzip -q platanusResult.zip [ 8:09 午後]
iu@bielinux[backup] ls -ld pla* [ 8:13 午後]
drwxrwxr-x 2 iu iu 4096 1月 21 10:09 platanusResult
-rw-rw-r-- 1 iu iu 2264981 1月 21 12:59 platanusResult.zip
iu@bielinux[backup] cd platanusResult [ 8:14 午後]
iu@bielinux[platanusResult] pwd [ 8:21 午後]
/home/iu/Desktop/backup/platanusResult
iu@bielinux[platanusResult] ls [ 8:21 午後]
out_32merFrq.tsv    out_lib1_insFreq.tsv
out_contigBubble.fa  out_scaffoldBubble.fa
out_contig.fa        out_scaffoldComponent.tsv
out_gapClosed.fa     out_scaffold.fa
iu@bielinux[platanusResult] [ 8:21 午後]
```

- ①
- ②
- ③

W20-6：配列数確認

```
File Edit View Search Terminal Help  
iu@bielinux[platanusResult] pwd  
/home/iu/Desktop/backup/platanusResult  
iu@bielinux[platanusResult] ls -l *.fa [ 8:27 午後]  
-rw-rw-r-- 1 iu iu 325 1月 21 10:08 out_contigBubble.fa  
-rw-rw-r-- 1 iu iu 2436380 1月 21 10:08 out_contig.fa  
-rw-rw-r-- 1 iu iu 2387680 1月 21 10:09 out_gapClosed.fa  
-rw-rw-r-- 1 iu iu 0 1月 21 10:09 out_scaffoldBubble.fa  
-rw-rw-r-- 1 iu iu 2389243 1月 21 10:09 out_scaffold.fa  
iu@bielinux[platanusResult] grep -c ">" *.fa [ 8:27 午後]  
out_contigBubble.fa:1  
out_contig.fa:349  
out_gapClosed.fa:117 (3)  
out_scaffoldBubble.fa:0  
out_scaffold.fa:117  
iu@bielinux[platanusResult]
```

①拡張子が.faのファイルを表示。②*.faの配列数を一気に出力。③確かにPlatanusの最終結果ファイルであるout_gapClosed.faの配列数は117個であり、④DDBJ Pipelineのウェブサイト上の数値と一致する

[8:27 午後]
Contig # : 117
Total contig size : 2,356,019
Maximum contig size : 257,728
Minimum contig size : 101
N50 contig size : 92,304 (4)

W20-6：配列数確認

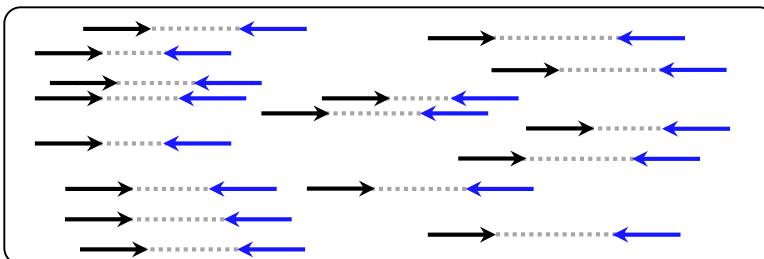
①赤下線の配列数は、2016.07.20のスライド50と同じです。この3つは、Platanusによる3ステップからなる*de novo*アセンブリの、各ステップ実行後の出力ファイルでした

```
iu@bielinux[platanusResult] pwd [ 8:27 午後]
/home/iu/Desktop/backup/platanusResult
iu@bielinux[platanusResult] ls -l *.fa
-rw-rw-r-- 1 iu iu    325 1月 21 10:08 out_contigBubble.fa
-rw-rw-r-- 1 iu iu 2436380 1月 21 10:08 out_contig.fa
-rw-rw-r-- 1 iu iu 2387680 1月 21 10:09 out_gapClosed.fa
-rw-rw-r-- 1 iu iu      0 1月 21 10:09 out_scaffoldBubble.fa
-rw-rw-r-- 1 iu iu 2389243 1月 21 10:09 out_scaffold.fa
iu@bielinux[platanusResult] grep -c ">" *.fa [ 8:27 午後]
out_contigBubble.fa:1
out contig.fa:349
out gapClosed.fa:117
out scaffoldBubble.fa:0
out scaffold.fa:117
iu@bielinux[platanusResult] [ 8:27 午後]
```



de novoアセンブリ

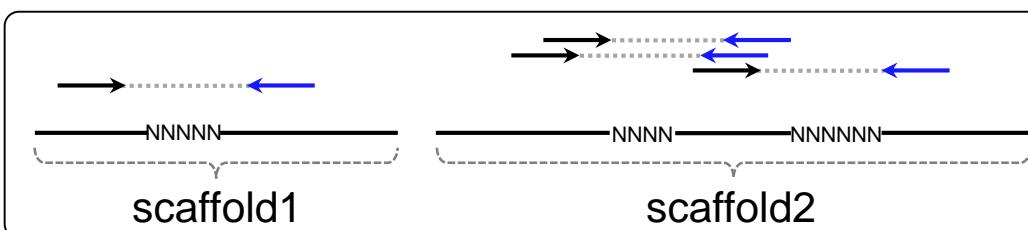
入力: paired-end FASTQファイル



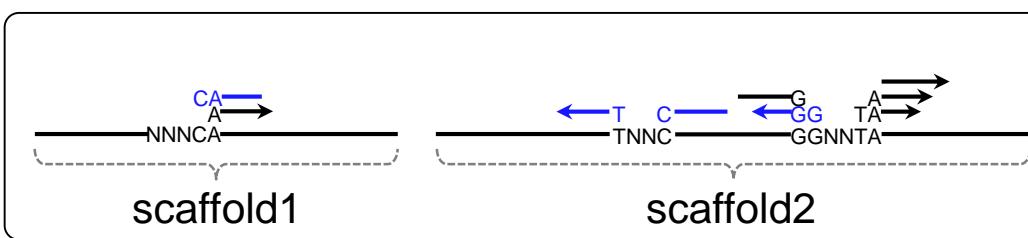
Step1: Assembly



Step2: Scaffold



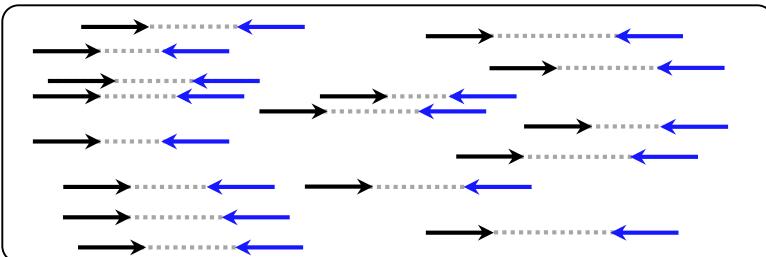
Step3: Gap close



おさらい: *de novo*アセンブリの手順
は大まかに3つのステップからなる

de novoアセンブリ

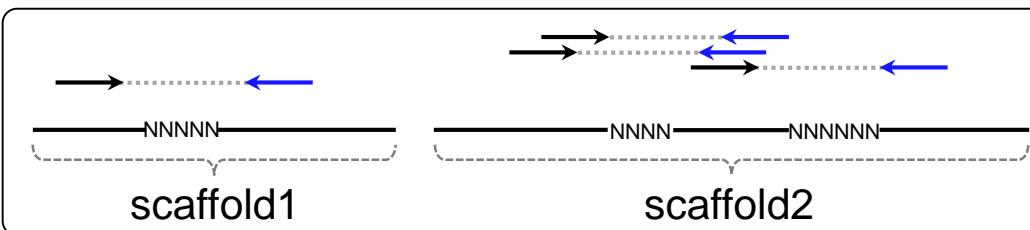
入力: paired-end FASTQファイル



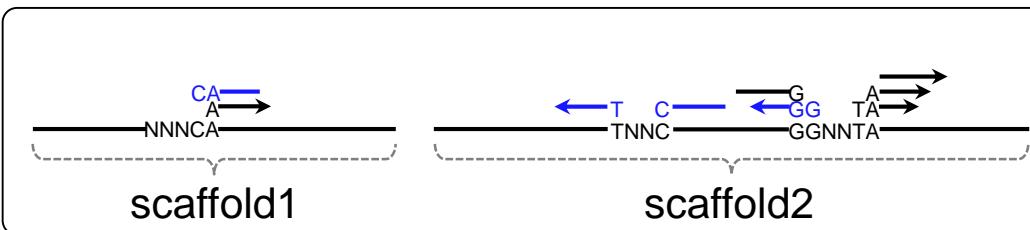
Step1: Assembly → out_contig.fa (349配列) ①



Step2: Scaffold → out_scaffold.fa (117配列) ②



Step3: Gap close → out_gapClosed.fa (117配列) ③



①②③の3つは、各ステップ実行後のファイルだったことを思い出そう。配列数の変遷が妥当である、という議論を2016.07.20にしました



W20-7: フィルタリング

out_gapClosed.faを入力として、①300 bp未満の配列を除去するプログラムを実行。②配列数は52個。③原著論文中の結果(53配列)と似た個数が得られたことがわかる

```
iu@bielinux[platanusResult] pwd
/home/iu/Desktop/backup/platanusResult
iu@bielinux[platanusResult] ls -l *.fa [ 8:27 午後]
-rw-rw-r-- 1 iu iu 325 1月 21 10:08 out_contigBubble.fa
-rw-rw-r-- 1 iu iu 2436380 1月 21 10:08 out_contig.fa
-rw-rw-r-- 1 iu iu 2387680 1月 21 10:09 out_gapClosed.fa
-rw-rw-r-- 1 iu iu 0 1月 21 10:09 out_scaffoldBubble.fa
-rw-rw-r-- 1 iu iu 2389243 1月 21 10:09 out_scaffold.fa
iu@bielinux[platanusResult] grep -c ">" *.fa [ 8:27 午後]
out_contigBubble.fa:1
out_contig.fa:349
out_gapClosed.fa:117
out_scaffoldBubble.fa:0
out_scaffold.fa:117
iu@bielinux[platanusResult] fastaLengthFilter.py out_gapClosed.
fa 300 > hoge.fa
iu@bielinux[platanusResult] grep -c ">" hoge.fa [ 8:31 午後]
52
iu@bielinux[platanusResult] [ 8:31 午後]
```

①

②

③

W20-7：総塩基数

```

File Edit View Search Terminal Help [ 8 ]
iu@bielinux[platanusResult] pwd
/home/iu/Desktop/backup/platanusResult [ 8 ]
iu@bielinux[platanusResult] ls -l *.fa
-rw-rw-r-- 1 iu iu 2347176 6月 16 20:31 hoge.fa
-rw-rw-r-- 1 iu iu      325 1月 21 10:08 out_contigBub
-rw-rw-r-- 1 iu iu 2436380 1月 21 10:08 out_contig.fa
-rw-rw-r-- 1 iu iu 2387680 1月 21 10:09 out_gapClosed.fa
-rw-rw-r-- 1 iu iu      0 1月 21 10:09 out_scaffoldBubble.fa
-rw-rw-r-- 1 iu iu 2389243 1月 21 10:09 out_scaffold.fa
iu@bielinux[platanusResult] grep -v ">" out_gapClosed.fa | wc
29506 29506 2385525
iu@bielinux[platanusResult] grep -v ">" hoge.fa | wc
52      52 2346552
iu@bielinux[platanusResult] [ 8:39午後 ]

```

「grep -v ">"」は、">"を含まない行を出力する(W8-2; W10; W12-8など)。①300 bp未満をフィルタリングする前は総塩基数が(2,385,525 - 29506) = 2,356,019 bpであった。②フィルタリングして配列数が117 → 52個に減ったあとの総塩基数は(2,346,552 - 52) = 2,346,500 bp



Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W21-2:ACGTカウント2

• ACGTカウント2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffold時にNが挿入したい。テンプレートとしては、「解析 | 一般 | GC含量(GC contents)」を用

```
pwd
ls *.fa
R -q
library(Biostrings)
```



```
fastas <- readDNAStringSet("out_contig.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_scaffold.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_gapClosed.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)
```

スライド120まで省略

①赤枠部分がA, C, G, T, Nの出現回数をカウントするRコード。Step1, 2, 3の配列長フィルタリング前の出力結果ファイルを入力している。
②のkopipe実行結果が次のスライド。
2016.07.20のスライド30-49で、ホストOSのR上で行った作業と同じことがLinux上でもできることがわかつてもらえばそれでよし

スライド120まで省略

W21-2:ACGTカウント2

```
File Edit View Search Terminal Help [ 9:48 午前 ]  
iu@bielinux[platanusResult] pwd  
/home/iu/Desktop/backup/platanusResult  
iu@bielinux[platanusResult] ls *.fa [ 9:48 午前 ]  
hoge.fa          out_contig.fa    out_scaffoldBubble.fa  
out_contigBubble.fa  out_gapClosed.fa  out_scaffold.fa  
iu@bielinux[platanusResult] R -q [ 9:48 午前 ]  
> library(Biostrings) [ ② ]
```

ACGTのカウントを調べたいファイルがあるディレクトリ上で、①Rを起動(第5回W9-8)。②library(Biostrings)と打って、リターンキーを押す

W21-2:ACGTカウント2

File Edit View Search Terminal Help

The following objects are masked from 'package:base' :

```
anyDuplicated, append, as.data.frame, as.vector, cbind, col  
names,  
do.call, duplicated, eval, evalq, Filter, Find, get, inters  
ect,  
is.unsorted, lapply, Map, mapply, match, mget, order, paste  
, pmax,  
pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce, re  
p.int,  
rownames, sapply, setdiff, sort, table, tapply, union, uniq  
ue,  
unlist, unsplit
```

Loading required package: S4Vectors

Loading required package: stats4

Creating a generic function for 'nchar' from package 'base' in
package 'S4Vectors'

Loading required package: IRanges

Loading required package: XVector

> █

エラーなく読み込み完了。これで
Biostringsパッケージが提供するACGTの
文字列をカウントするalphabetFrequency
関数などを利用可能な状態になった

まずは、①PlatanusのStep1実行結果ファイル([out_contig.fa](#))を入力として、「A, C, G, T, Nの出現回数をカウントするRコード」をコピペ実行

W21-2:ACGTカウント

ACGTカウント 2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffold時にNが挿入されているのかなどを確認したい。テンプレートとしては、「[解析 | 一般 | GC含量\(GC contents\)](#)」を用いた。連載第5回のW9-8。

```
pwd
ls *.fa
R -q
library(Biostrings)

fastas <- readDNAStringSet("out_contig.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_scaffold.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_gapClosed.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)
```

①

W21-2:ACGTカウント2

コピペ実行結果。①の実行結果部分が、A, C, G, T, Nの塩基ごとの出現回数。②Nは1つもないことがわかる

ACGTカウント2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffold時にNが挿入されているのかなどを確認したい。テンプレートとしては、「解析 | 一般 | GC含量(GC contents)」を用いた。連載第5回のW9-8。

```

pwd
ls *.fa
R -q
library(Biostrings)

fastas <- readDNAStringSet("out_contig.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_contig.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_contig.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

```

```

pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce, re
p.int,
rownames, sapply, setdiff, sort, table, tapply, union, uniq
ue,
unlist, unsplit

Loading required package: S4Vectors
Loading required package: stats4
Creating a generic function for 'nchar' from package 'base' in
package 'S4Vectors'
Loading required package: IRanges
Loading required package: XVector
> fastas <- readDNAStringSet("out_contig.fa", format="fasta")
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
      A        C        G        T        N
739836  469377  446806  742714      0
> sum(hoge)
[1] 2398733
>

```

W21-2:ACGTカウント

ACGTカウント 2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffold時にNが認識したい。テンプレートとしては、「[解析 | 一般 | GC含量\(GC contents\)](#)」を

①の実行結果部分は、総塩基数を計算しているところ。②alphabetFrequency関数実行によって得られた結果をhogeというものに格納している。このhogeを入力として総和を計算するsum関数を適用して全塩基数とみなしている

```

pwd
ls *.fa
R -q
library(Biostrings)

fastas <- readDNAStringSet("out_contig.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge) ①

fastas <- readDNAStringSet("out_contig.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fastas <- readDNAStringSet("out_contig.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

          pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce, re
          p.int,
          rownames, sapply, setdiff, sort, table, tapply, union, uniq
          ue,
          unlist, unsplit

Loading required package: S4Vectors
Loading required package: stats4
Creating a generic function for 'nchar' from package 'base' in
package 'S4Vectors'
Loading required package: IRanges
Loading required package: XVector
> fastas <- readDNAStringSet("out_contig.fa", format="fasta") ②
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
      A      C      G      T      N
    739836 469377 446806 742714      0
> sum(hoge)
[1] 2398733
>

```

W21-2:ACGTカウント2

ACGTカウント2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffold時にNが挿入されるか確認したい。テンプレートとしては、「解析 | 一般 | GC含量(GC contents)」を用いた

①Step2実行結果ファイル(out_scaffold.fa)を入力として実行した結果。Scaffoldingによってコンティグ間が未知塩基Nで埋められたギャップで表現されるため、②Nが存在(491個)するのは妥当

```

pwd
ls *.fa
R -q
library(Biostrings)

fasta <- readDNAStringSet("out_contig.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fasta <- readDNAStringSet("out_scaffold.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)


```

```

Creating a generic function for 'nchar' from package 'base' in
package 'S4Vectors'
Loading required package: IRanges
Loading required package: XVector
> fasta <- readDNAStringSet("out_contig.fa", format="fasta")
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
      A      C      G      T      N
739836 469377 446806 742714     0
> sum(hoge)
[1] 2398733

```

①

```

> fasta <- readDNAStringSet("out_scaffold.fa", format="fasta")
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
      A      C      G      T      N
729635 458119 440510 727306    491
> sum(hoge)
[1] 2356061

```

②

①Step3実行結果ファイル(out_gapClosed.fa)を入力として実行した結果。Gap closingのおかげで、この場合は②Nが0個になっている。③総塩基数2,356,019 bpという数値は、W20-7のwcコマンドから得られる結果と同じ。同じ目的を達成する上でも様々な手段がある

W21-2:ACGTカウント

ACGTカウント 2[W20-11]

それぞれのA, C, G, T, Nのカウント数を調べて、本当にscaffoldとして認識したい。テンプレートとしては、「解析 | 一般 | GC含量(GC content)」

```

pwd
ls *.fa
R -q
library(Biostrings)

fasta <- readDNAStringSet("out_scaffold.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fasta <- readDNAStringSet("out_scaffold.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fasta <- readDNAStringSet("out_gapClosed.fa")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

```

	A	C	G	T	N
out_scaffold.fa	739836	469377	446806	742714	491
out_gapClosed.fa	729772	458211	440585	727451	0

W21-2: Rの終了

- ACGTカウント 2[W20-11]

それぞれの A, C, G, T, N のカウント数を調べて、本当に scaffold 時に N が挿入されているのかなどを確認したい。テンプレートとしては、「解析 | 一般 | GC含量(GC contents)」を用いた。連載第5回の W9-8。

```
R -q
library(Biostrings)

fasta <- readDNAStringSet("out_scaffold.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

fasta <- readDNAStringSet("out_gapClosed.fa", format="fasta")
hoge <- alphabetFrequency(fasta)
obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
colSums(hoge)[obj]
sum(hoge)

q(save="no")
```

739836 469377 446806 742714 0
> sum(hoge)
[1] 2398733
> fasta <- readDNAStringSet("out_scaffold.fa", format="fasta")
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
A C G T N
729635 458119 440510 727306 491
> sum(hoge)
[1] 2356061
> fasta <- readDNAStringSet("out_gapClosed.fa", format="fasta")
> hoge <- alphabetFrequency(fasta)
> obj <- is.element(colnames(hoge), c("A", "C", "G", "T", "N"))
> colSums(hoge)[obj]
A C G T N
729772 458211 440585 727451 0
> sum(hoge)
[1] 2356019
> q(save="no")
iu@bielinux[platanusResult]

[10:06午前]

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



おさらい

①乳酸菌ゲノム配列決定論文は、2種類のNGS機器から得られたデータを併用している。第6回はIllumina MiSeqデータ(DRR024501)を、連載第7回はPacBioデータを取り扱っている

PacBio RS IIデータ(DRR024500)

- DRR024500は登録内容に問題があったことが判明し消滅
- 4セル分のデータ。DRR054113–054116に差し替えられている
- セルあたり約15万リード。4セル分なので約60万リード

Illumina MiSeqデータ(DRR024501)

- paired-endゲノムデータ
- リード長は、forward側とreverse側共に250 bp
- オリジナルは2,971,310リード。最初の300,000リードを解析
- forward側(DRR024501sub_1.fastq.gz)
- reverse側(DRR024501sub_2.fastq.gz)

FaQCs実行結果(第6回W5–4)

- 300,000リード → 297,633リード (W5–2)
- forward側(QC.1.trimmed.fastq.gz)
- reverse側(QC.2.trimmed.fastq.gz)
- ファイルの場所: ~/Documents/DRR024501/result



NGS連載第7回は

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランскриプトーム、正規化、発現変動、統計、モ^テ
(last modified 2016/06/03, since 2011)

What's new?

- このウェブページ
リソースRと必要
法(Windows2015
ホリキオ / 2015)

- 書籍 | 日本乳酸菌学会誌 | 第5回アセンブル、マッピング、そしてQC (last modified 2016/06/03, since 2011)
- 書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ (last modified 2016/06/17) NEW
- 書籍 | 日本乳酸菌学会誌 | 第7回ロングリードアセンブリ (last modified 2016/05/12) ①
- イントロ | 一般 | ランダムに行を抽出 (last modified 2014/07/17)
- イントロ | 一般 | 任意の文字列を行の最初に挿入 (last modified 2014/07/17)
- イントロ | 一般 | 任意の文字列を行の最後に挿入 (last modified 2014/07/17)
- イントロ | 一般 | ランダムに行を抽出 (last modified 2014/07/17)
- イントロ | 一般 | 任意の文字列を行の最初に挿入 (last modified 2014/07/17)

書籍 | 日本乳酸菌学会誌 | 第7回ロングリードアセンブリ

日本乳酸菌学会誌の第7回分です。Linuxコマンドのリンク先は主に日経BP社様です。

- 原稿PDF
- ウェブ資料PDF
 - Windows用(2016.06.03)
 - Macintosh用

Linuxコマンド

- ant-cache (パッケージ調査)

②このあたりからスタート。ロングリードの代表格であるPacBioデータとその周辺の話。オリジナルの第7回ウェブ資料PDFと若干順番を入れ替えながら、まずは事実の羅列から話を進めます

①

②

PacBioのファイル形式とデータ解析の概要

- W1-1: PacificBiosciencesのYouTubeサイト
 - Introduction to SMRT Sequencing
 - Single Molecule Real Time Sequencing
- W2-1: PacBioデータ(原著論文中のDRR IDだが削除されている)
 - DRR024500: Tanizawa et al., BMC Genomics, 2015
- W2-3: DRR024501 -> DRP002401 -> DRX022185
- W2-4: DRR024501 -> DRA002643
- W2-5: PacBioデータ概観
 - DRR054113
 - DRR054114
 - DRR054115
 - DRR054116
- W2-6: SMRT Portal(PacBio提供のHGAPを含む解析ソフトウェア群)の場所

おさらい(NGS連載第3回)

①の第3回で行った議論は、
基本的に Illumina データの話。
PacBio データには通用しない

全部読んで説明

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

第3回W20からW21

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- FASTQ提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、**FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる**(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。



W2-7:bax.h5ファイル

PacBioのファイル形式とデータ解析の概要

- W1-1: PacificBiosciencesの YouTube サイト
 - [Introduction to SMRT Sequencing](#)
 - [Single Molecule Real Time Sequencing](#)
- W2-1: PacBioデータ(原著論文中のDRR IDだが削除されている)
 - [DRR024500: Tanizawa et al., BMC Genomics, 2015](#)
- W2-3: [DRR024501](#) -> [DRP002401](#) -> [DRX022185](#)
- W2-4: [DRR024501](#) -> [DRA002643](#)
- W2-5: PacBioデータ概観
 - [DRR054113](#)
 - [DRR054114](#)
 - [DRR054115](#)
 - [DRR054116](#)
- W2-6: SMRT Portal(PacBio提供のHGAPを含む解析ソフトウェア群)の場所
 - [PacBio](#) -> [DevNet](#) -> [SMRT Analysis](#)
 - SMRT Analysis 2.3までは、HGAPを実行するためには**bax.h5**ファイルが必須。
 - SMRT Analysis 3.0からは、BAMファイルが入力フォーマットになる。但しここでのBAMファイルは、マッピングデータではなく、シークエンス生データ。
 - PacBio RSIIの後継機である[Sequel](#)の出力ファイル形式はBAM。
 - PacBioのファイル形式の説明については[こちら](#) (<http://pacbiofileformats.readthedocs.io/en/latest/>)^(3.0)。
- W2-7: [DRR054113](#)の**bax.h5**ファイル (下記3ファイル合わせてDRR054113に相当)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.1.bax.h5](#) (747 MB; 784,301,199 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.2.bax.h5](#) (766 MB; 803,938,042 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.3.bax.h5](#) (901 MB; 945,597,712 bytes)

まず、①PacBioの生データは、sraでもFASTQでなく、bax.h5という形式のファイル(正確にはPacBio RS IIという機器が出力するファイル形式)。②サイズも巨大。具体的には、1セル分のみでも(747MB + 766MB + 901MB) = 2,414MB(約2.4GB)に達する。③これはDRR054113の1セル分のデータだが、1ファイル/セルではなく、3つに分割されている点も最初は戸惑うポイント

おさらい

PacBio RS IIデータ(DRR024500)

- DRR024500は登録内容に問題があったことが判明し、削除
- 4セル分のデータ。DRR054113–054116に差し替えられている
- セルあたり約15万リード。4セル分なので約60万リード

①乳酸菌ゲノム配列決定論文は、②PacBio RS IIで4セル分のデータを取得し、*de novo*アセンブリを行った。DRA上では、セルごとにDRR054113–054116という計4つのIDで登録されている。その内の1つであるDRR054113について解説しました

②

Illumina MiSeqデータ(DRR024501)

- paired-endゲノムデータ
- リード長は、forward側とreverse側共に250 bp
- オリジナルは2,971,310リード。最初の300,000リードを解析
- forward側(DRR024501sub_1.fastq.gz)
- reverse側(DRR024501sub_2.fastq.gz)

FaQCs実行結果(第6回W5–4)

- 300,000リード → 297,633リード (W5–2)
- forward側(QC.1.trimmed.fastq.gz)
- reverse側(QC.2.trimmed.fastq.gz)
- ファイルの場所: ~/Documents/DRR024501/result

①

W2-7:bax.h5ファイル

PacBioのファイル形式とデータ解析の概要

- W1-1: PacificBiosciencesのYouTubeサイト
 - [Introduction to SMRT Sequencing](#)
 - [Single Molecule Real Time Sequencing](#)
- W2-1: PacBioデータ(原著論文中のDRR IDだが削除されている)
 - [DRR024500: Tanizawa et al., BMC Genomics, 2015](#)
- W2-3: [DRR024501](#) -> [DRP002401](#) -> [DRX022185](#)
- W2-4: [DRR024501](#) -> [DRA002643](#)
- W2-5: PacBioデータ概観
 - [DRR054113](#)
 - [DRR054114](#)
 - [DRR054115](#)
 - [DRR054116](#)
- W2-6: SMRT Portal(PacBio提供のHGAPを含む解析ソフトウェア群)の場所
 - [PacBio](#) -> [DevNet](#) -> [SMRT Analysis](#)
 - SMRT Analysis 2.3までは、HGAPを実行するためにはbax.h5ファイルが必須。
 - SMRT Analysis 3.0からは、BAMファイルが入力フォーマットになる。但しここでのBAMファイルは、マッピングデータではなく、シークエンス生データ。
 - PacBio RSIIの後継機である[Sequel](#)の出力ファイル形式はBAM。
 - PacBioのファイル形式の説明については[こちら](#) (<http://pacbiofileformats.readthedocs.io/en/3.0/>)。
- W2-7: [DRR054113](#)のbax.h5ファイル (下記3ファイル合わせてDRR054113に相当)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.1.bax.h5](#) (747 MB; 784,301,199 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.2.bax.h5](#) (766 MB; 803,938,042 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.3.bax.h5](#) (901 MB; 945,597,712 bytes)

DDBJ Pipeline上では、PacBio用の*de novo*アセンブリ用プログラムHGAPを利用可能。しかし、① HGAPはbax.h5ファイルのみしか受け付けない。HGAPを内包するPacBio提供のSMRT Analysisシステムのインストールは超高難易度(個人の感想です)。しかも数百GBメモリ搭載マシンでないと動かせない(DDBJ Pipeline上でDRR054113のみを入力としてHGAPを実行しても120GB程度のメモリを要する)。共同研究などで他人に頼むなど以外の場合、通常はDDBJ Pipeline上でHGAP実行一択



おさらい(NGS連載第3回)

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- FASTQ提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、**FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる**(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

①公共DB上では、bax.h5ファイルは公開されていません。sraやFASTQファイルが手元にあっても実質的には(DDBJ PipelineでHGAPを実行できないので)無意味。しかも②(クオリティ値がPacBioよりも高い) Illuminaデータの場合は、sraからFASTQへの変換時にリード数が**若干減る程度**だが…

①

②

連載第7回W3

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

①PacBioデータの場合は**相当**減る。具体的には、②例えばDRA上で同じDRR054113をダウンロードしても、sraファイルは163,482リード(**実際には若干異なる?!**)であるのに対し、FASTQファイルだと915リードという結果になる。これは減ったリード数ではなく、生き残ったリード数

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- FASTQ提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、**FASTQファイル中のリード数は、元のsraファイル中のリード数よりも**若干**少なくなる**(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

連載第7回W3

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- FASTQ提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

①これはおそらく、fastq-dump実行時に指定しているオプションの閾値が、Illuminaデータを合理的にフィルタリングするための条件のままで統一的にPacBioデータに対しても適用しているためであろう(推測の域を出ないが、論理的に考えればそのはず)



この後の展開は…

- 3大公共DB(NGS分野)の提供ファイル形式
 - DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
 - EMBL-EBI ENA (ENA): FASTQ形式①gzip圧縮)
 - NCBI SRA (SRA): sra形式
- 大元はsra形式なので…
 - FASTQファイルはsraファイルから作成する(sra → FASTQ)
 - SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
 - FAS②提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

①DRA上でダウンロードしたDRR054113のFASTQファイルが915リードになるのをFastQCで確認します(第7回W3-2)。次に、(第4回W4-5, W13-5, W14, W15でも紹介した)apt-getを用いたプログラムインストールの応用編(第7回W4)として、②SRA Toolkitを解説

念のため説明

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- FAS①**提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

①SRA Toolkitは、第7回のオリジナルウェブ資料(W5)ではFASTQファイルを生成してFastQCで確認する用途として利用しています。しかし前述のように、PacBioは入力がbax.h5ファイルなので「PacBioデータの性質を確認できてよかったです」の域を出ません。そして Illuminaの場合は最初からFASTQファイルを使えばよいので、sraファイルを取り扱う意義は見出せません(第3回原稿PDFのp36左下)

念のため説明

■ 3大公共DB(NGS分野)の提供ファイル形式

- DDBJ SRA (DRA): sra形式とFASTQ形式(bzip2圧縮)
- EMBL-EBI ENA (ENA): FASTQ形式(gzip圧縮)
- NCBI SRA (SRA): sra形式

■ 大元はsra形式なので…

- FASTQファイルはsraファイルから作成する(sra → FASTQ)
- SRA Toolkit中のfastq-dumpで「sra → FASTQ」の変換を行う
- **FAS**①提供サイト(DRAとENA)は、fastq-dump実行時にクオリティフィルタリングなどを行っている。このため、FASTQファイル中のリード数は、元のsraファイル中のリード数よりも若干少なくなる(第3回W24)。実際、SRR616268のsraファイルは135,073,834リード、FASTQファイルは134,755,996リードであった(第3回W24-3)。

①SRA Toolkitは、あくまでもapt-getを用いたプログラムインストール(応用編; 第7回W4)の例題という位置づけ。apt-get(およびapt-cache)を使いこなして効率的にインストールを行うスキルがあれば、*de novo transcriptome assembly*プログラムの1つであるTrinityのインストールも行えます(2016.08.04)

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6: FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W3-1 : FASTQダウンロード

DRASearch

DRR054113 [FASTQ](#) [SRA](#)

Run Detail

Alias	DRR054113
Instrument model	
Date of run	
Run center	
Number of spots	163,482
Number of bases	360,244,590

READS (joined)

qual	date	count	file
>DRR054113.1	01/27/2016 02:32午後	2,392,259	DRR054113.fastq.bz2
CCTATGCTGTCAGCATTGATTGCTAGTTGAT	01/27/2016 02:32午後	2,901,836	DRR054114.fastq.bz2
GCTACTAGTCTGAGTCTGCCTGACTGATTGA	01/27/2016 02:33午後	3,858,646	DRR054115.fastq.bz2
GTACGGCACGCATGTAGTACTGGTGATGACC	01/27/2016 02:33午後	4,455,155	DRR054116.fastq.bz2
GCAGGGCATGATACCAAGGTCGATTCACTATTGATGTCTAGACTTAGCTGACATAGCAGATTGATCTCTGATTACAGG			
CGATATCGCACTGCGTCATACGATTACAGTCA			
>DRR054113.2			
TCATATACTCGGCACAATGTGTGATCGTAAAGGGATGTCATTGTGTAGTATTGTATTCTATATGTCGAGCATCAGCG			
TTCTACTGCTGAGATGATATATTCTGAGTATTATGGTTATGTATTTCACGTGAAACCTGGATTATGTCGTGGACGGACGT			
TGTACGGATTCTAACTGTTAGTATCGAGCATTGATCGTCGATGGATTGATAGTGTCTCCGTTGAGTCGTAATGATTGTT			
CAGTTAGTCGATCAGTCGTTGAGTCAGTGAATTGATTTGTAAGCGAAGATTATGAATCTTACGATCCTTATGGCTGAGTTGAA			
TGATCTGCTGCTAGTGTCTGTATGTTGATGATCACATGACGATACGTGATATTATTGTCACGCATCGATTGAG			

Navigation

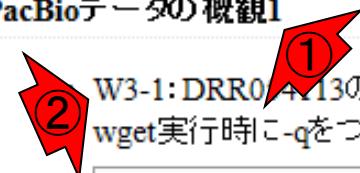
- Submission DRP002401
- Study DRP002401
- Experiment DRX022185 [FASTQ](#)

① DRR054113、②FASTQ、③bzip2
圧縮FASTQファイルをダウンロード。
右クリックで「ショートカットのコピー」などでURL情報を取得(第4回
W9-2やW18-1)してwgetしてもよい
し、共有フォルダ経由でBio-Linux上
に置いてもよい。とオリジナルのウ
ェブ資料は書いてあるが、講習会用
に既に~/Desktop/backupにダウ
ンロード済み。それをコピーして利用

② **③**

W3-1:FASTQダウンロード

PacBioデータの概観1



① W3-1: DRR054113のFASTQファイルをダウンロード

wget実行時に-qをつけて途中経過を非表示。URL情報取得については、連載第4回W9-2+

```
cd ~/Documents  
pwd  
mkdir DRR054113  
cd DRR054113  
pwd  
#wget -cq ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/DRA002/DRA002643/DRX022185/DRR054113.fastq.bz2  
cp ~/Desktop/backup/DRR054113.fastq.bz2 .  
ls -l  
ls -lh
```



- ①今はこのあたりの話をしている。
- ②「W3-1」という情報を頼りに探す。
- ③赤枠をそのまま実行すればwgetは実行されないが、大量同時アクセスでDDBJに迷惑をかけないように、~/Desktop/backup上にあるファイルのコピーにしてください

W3-2:FastQC (ver. 0.11.4)実行

FastQC ver. 0.11.4は、第4回W9-2でインストールし、fastqc2というコマンドでパスを通している。

```
cd ~/Documents/DRR054113  
  
pwd  
ls -l  
fastqc2 -v  
fastqc2 -q DRR054113.fastq.bz2 --outdir=/home/iu/Desktop/mac_share  
ls -l ~/Desktop/mac_share
```

W3-3:結果を見る

[DRR054113_fastqc.html](#)

W3-4:配列長分布

[DRR054113_fastqc.html](#)

W3-1:FASTQダウンロード

```
File Edit View Search Terminal Help  
iu@bielinux[iu] cd ~/Documents  
iu@bielinux[Documents] pwd  
/home/iu/Documents  
iu@bielinux[Documents] mkdir DRR054113  
iu@bielinux[Documents] cd DRR054113  
iu@bielinux[DRR054113] pwd  
/home/iu/Documents/DRR054113  
iu@bielinux[DRR054113] wget -cq ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/DRA002/DRA002643/DRX022185/DRR054113.fastq.bz2  
iu@bielinux[DRR054113] ls -l  
total 2340  
-rw-rw-r-- 1 iu iu 2392259 3月 22 12:32 DRR054113.fastq.bz2  
iu@bielinux[DRR054113] ls -lh  
total 2.3M  
-rw-rw-r-- 1 iu iu 2.3M 3月 22 12:32 DRR054113.fastq.bz2  
iu@bielinux[DRR054113] [redacted]
```

① ファイルマネージャー
② ウェブブラウザ
③ テキストエディタ
④ ディレクトリ作成

つまり、①~/Documents/DRR054113で、②wgetの部分はcpコマンドのほうのコピペにしてください、ということ。

④ディレクトリ作成は絶対に行い、指定された場所で作業を行うこと！



W3-2: FastQC

```
File Edit View Search Terminal Help  
iu@bielinux[DRR054113] pwd  
/home/iu/Documents/DRR054113  
iu@bielinux[DRR054113] ls -l  
total 2340  
-rw-rw-r-- 1 iu iu 2392259 3月 22 12:32 DRR054113.fastq.bz2  
iu@bielinux[DRR054113] fastqc2 -v  
FastQC v0.11.4  
iu@bielinux[DRR054113] fastqc2 -q DRR054113.fastq.bz2 --outdir=  
/home/iu/Desktop/mac_share  
iu@bielinux[DRR054113] ls -l ~/Desktop/mac_share [12:35午後]  
total 751  
-rwxrwxrwx 1 iu iu 392406 3月 22 12:35 DRR054113_fastqc.html  
-rwxrwxrwx 1 iu iu 375781 3月 22 12:35 DRR054113_fastqc.zip  
iu@bielinux[DRR054113] █
```

①FastQC ver. 0.11.4は、第4回W9-2でインストールし、fastqc2というコマンドでパスを通している。②FastQCを実行し、共有フォルダ(~/Desktop/mac_share)に保存している。③ファイルの確認。ヒトによっては他のものも見えるだろうが、最低限④が見えていれば問題ない

[12:35午後]

[12:35午後]

[12:35午後]

[12:35午後]

①

②

③

④

W3-3: 結果を眺める

①共有フォルダに保存することで、使いなれた
ホストOS(この場合Windows)上で②FastQC
実行結果ファイルを眺めることができる

The screenshot shows a Windows desktop environment. On the left, there is a file explorer window with two red numbered arrows pointing to specific items: arrow 1 points to the 'share' folder icon, and arrow 2 points to the 'DRR054113_fastqc.html' file icon. Both items have a small dog icon next to them. The main window is a 'FastQC Report' application. The title bar shows the path 'C:\Users\kadota\Desktop\share\DRR054113_fastqc.html'. The application interface includes a 'Summary' section on the left with various quality check links, and a 'Basic Statistics' table on the right. The 'Basic Statistics' table contains the following data:

Measure	Value
Filename	DRR054113.fastq.bz2
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	915
Sequences flagged as poor quality	0
Sequence length	923-8076
%GC	38

Below the table is a 'Per base sequence quality' chart showing quality scores across all bases (Sanger / Illumina 1.9 encoding). The y-axis ranges from 50 to 80, and the x-axis shows sequence segments. The chart is mostly green, indicating good quality.

Produced by FastQC (version 0.11.4)

W3-3: 結果を眺める

FastQC実行結果の解説は第4回W8とW17、および第6回W4にもあり。①入力ファイル。②リード数は915、③配列長は923-8076 bpの範囲であることがわかる

FastQC Report

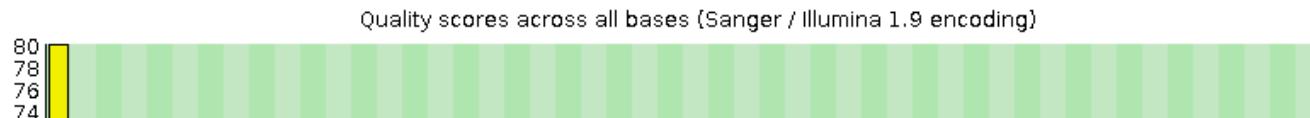
Summary

- ✓ Basic Statistics
- ✗ Per base sequence quality
- ✗ Per sequence quality scores
- ✗ Per base sequence content
- ! Per sequence GC content
- ✓ Per base N content
- ! Sequence Length Distribution
- ✓ Sequence Duplication Levels
- ! Overrepresented sequences
- ✓ Adapter Content
- ! Kmer Content

Basic Statistics

Measure	Value
Filename	DRR054113.fastq.bz2
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	915
Sequences flagged as poor quality	0
Sequence length	923-8076
%GC	38

✗ Per base sequence quality



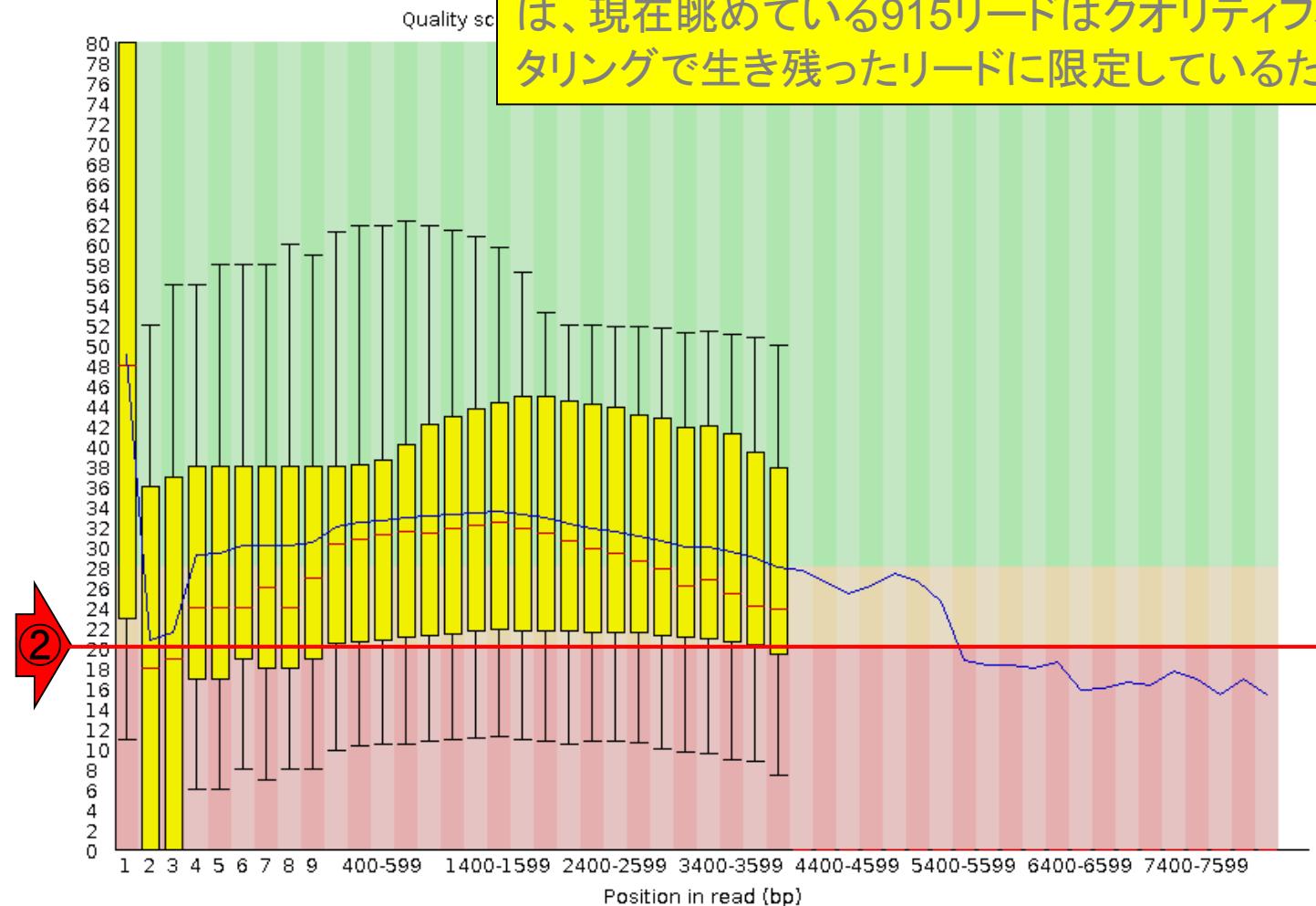
W3-3: 結果を眺める

FastQC Report

Summary

- ✓ Basic Statistics
- ✗ Per base sequence quality ①
- ✗ Per sequence quality scores
- ✗ Per base sequence content
- ! Per sequence GC content
- ✓ Per base N content
- ! Sequence Length Distribution
- ✓ Sequence Duplication Levels
- ! Overrepresented sequences
- ✓ Adapter Content
- ! Kmer Content

✗ Per base sequence quality



①Per base sequence quality。この図の縦軸はクオリティスコア。②赤線のスコア20を超えているかどうかが1つの目安。Illumina HiSeq2000 (第4回W8)やMiSeq (第6回W4)とは傾向が異なる。第7回W6では詳述しているが、PacBioの公称スペックよりも全体的なクオリティが高い。この理由は、現在眺めている915リードはクオリティフィルタリングで生き残ったリードに限定しているため

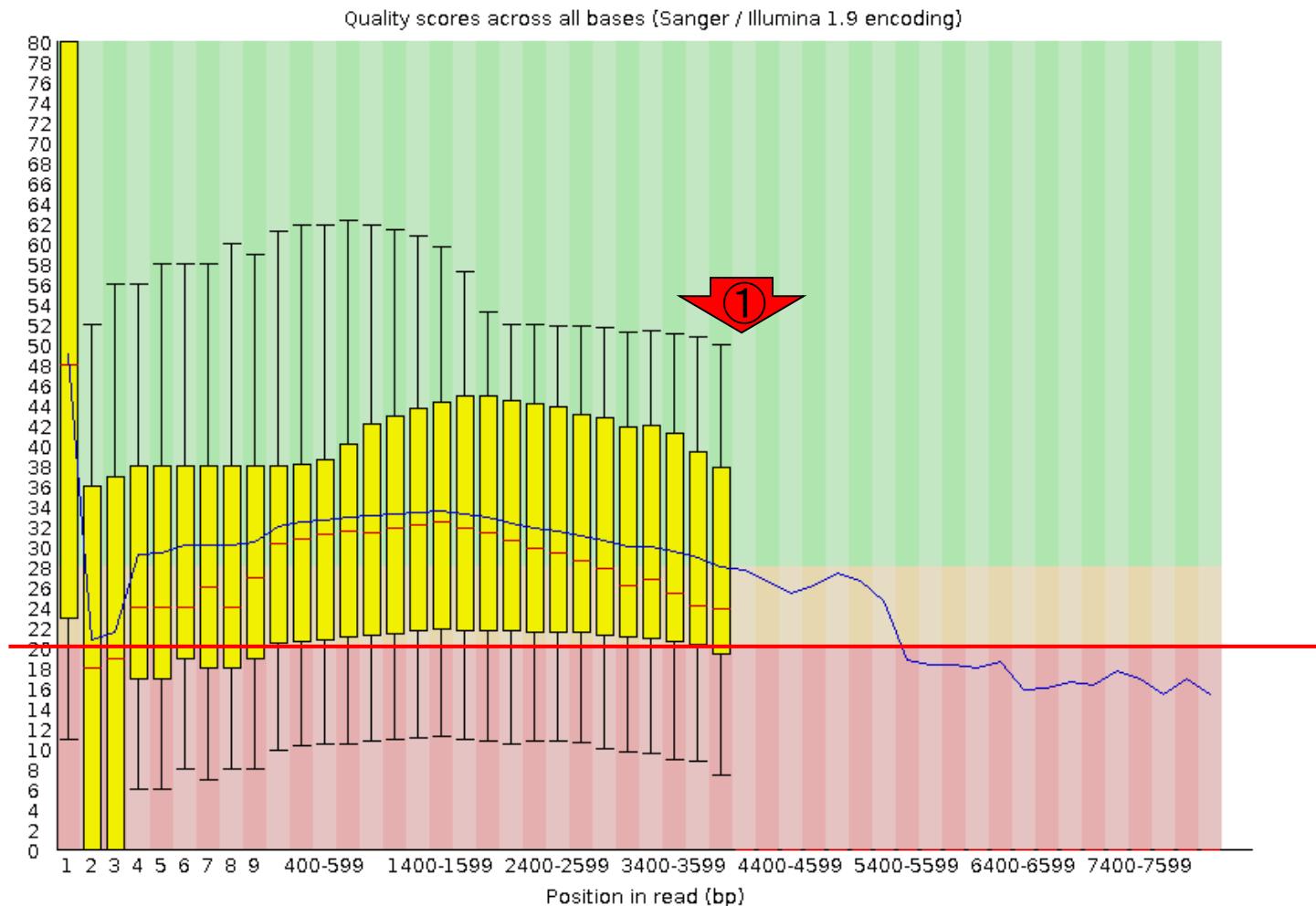
W3-3: 結果を眺める

FastQC Report

Summary

- ✓ Basic Statistics
- ✗ Per base sequence quality
- ✗ Per sequence quality scores
- ✗ Per base sequence content
- ! Per sequence GC content
- ✓ Per base N content
- ! Sequence Length Distribution
- ✓ Sequence Duplication Levels
- ! Overrepresented sequences
- ✓ Adapter Content
- ! Kmer Content

✗ Per base sequence quality



①横軸のリードポジションが4000 bpあたりで黄色の縦棒がなくなっている。この理由は、4000 bp以上のリードが少数だからと思われる。それは②の配列長分布(Sequence Length Distribution)でも確認できる

W3-4：配列長分布

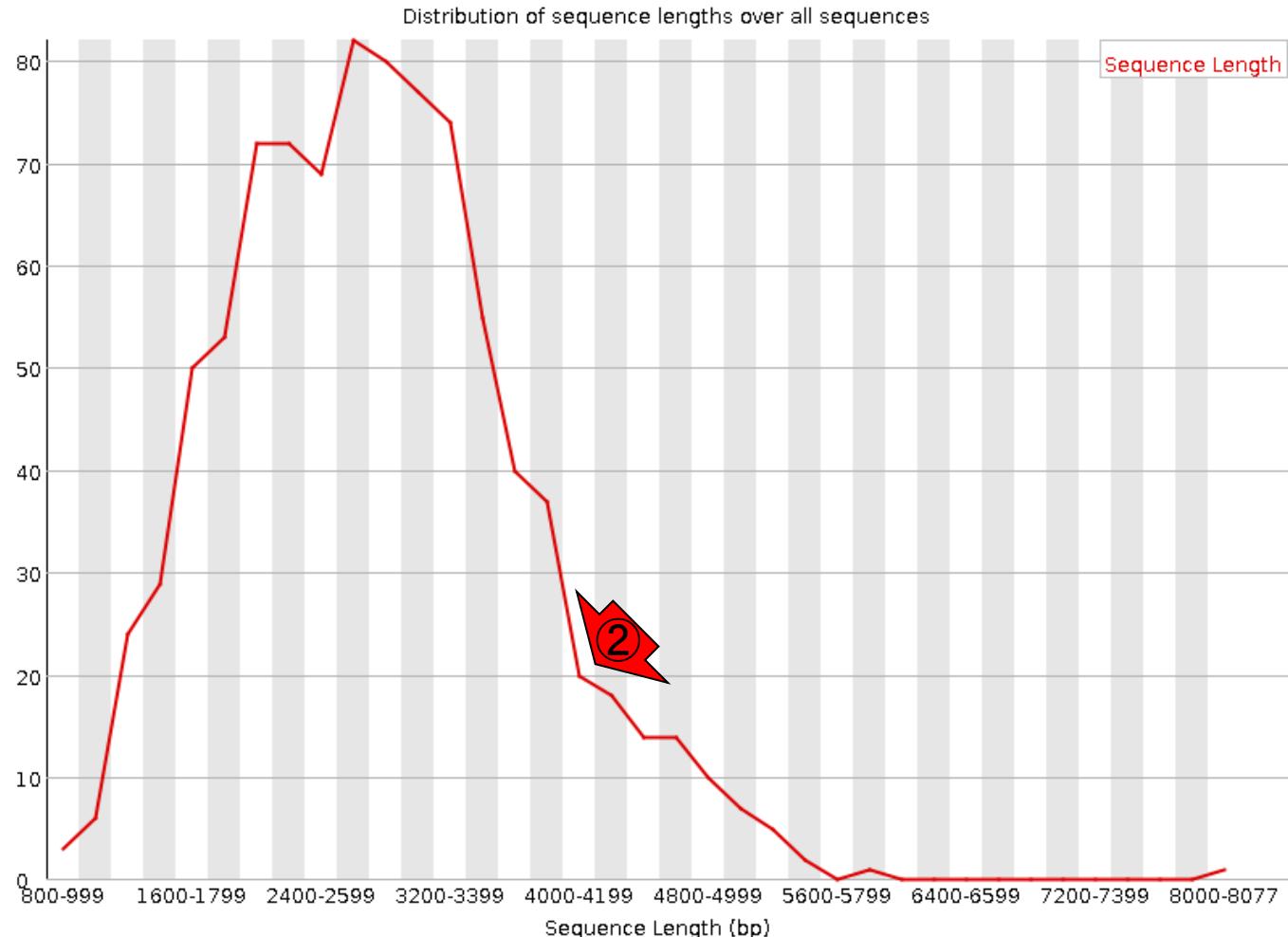
FastQC Report

①配列長分布(Sequence Length Distribution)。②このあたりで黄色の縦棒がなくなっているので、おそらく20リードが黄色の縦棒の有無の閾値なのだろう

Summary

- Basic Statistics
- Per base sequence quality
- Per sequence quality scores
- Per base sequence content
- Per sequence GC content
- Per base N content
- Sequence Length Distribution (1)
- Sequence Duplication Levels
- Overrepresented sequences
- Adapter Content
- Kmer Content

Sequence Length Distribution



Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6:FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



NGS連載第7回の...

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランскриプトーム、正規化、発現変動、統計、モラフ (last modified 2016/06/03, since 2011)

What's new?

- このウェブページリソース Rと必要な方法(Windows2015)

- 書籍 | 日本乳酸菌学会誌 | 第5回アセンブル、マッピング、そしてQ
- 書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ (last modified 2015/06/03, since 2011)
- 書籍 | 日本乳酸菌学会誌 | 第7回ロングリードアセンブリ (last modified 2016/06/03, since 2011)
- ・ イントロ | 一般 | ランダムに行を抽出 (last modified 2014/07/17)
- ・ イントロ | 一般 | 任意の文字列を行の最初に挿入 (last modified 2014/07/17)
- ・ イントロ | 一般 | 任意の文字列を行の最後に挿入 (last modified 2014/07/17)
- ・ イントロ | 一般 | ランダムに行を抽出 (last modified 2014/07/17)
- ・ イントロ | 一般 | 任意の文字列を行の最初に挿入 (last modified 2014/07/17)
- ・ イントロ | 一般 | ランダムに行を抽出 (last modified 2014/07/17)
- ・ イントロ | 一般 | 任意の文字列を行の最後に挿入 (last modified 2014/07/17)

②場所はこのあたり。次のスライドは③のリンク先ですが、基本的にコマンド打ち込み部分以外は、リンク先にアクセスせずに、スライド眺めるだけでよい。ネットワーク経由の作業なのでうまくいかない可能性もあるが、その場合はスライド眺めながらのエアーハンズオンに切り替え。インストールに失敗しても、その後に影響しないような構成にします



書籍 | 日本乳酸菌学会誌 | 第7回ロングリードアセンブリ

日本乳酸菌学会誌の第7回分です。Linuxコマンドのリンク先は主に日経BP社様です。

- 原稿PDF

- ウェブ資料

- Wind...
Macintosh

Linuxコマンド

- apt-cache (18)

SRA Toolkit (ver. 2.5.7)のインストールと利用

- W4-1: SRA Toolkit

2016年3月22日現在、ver. 2.5.7が最新。

- wget利用時のソフトウェアURL情報
- SRA Toolkit Documentation
- Installation and Configuration Guide

- W4-2: apt-cache

「apt-cache -n search キーワード」で任意のキーワードを含むソフトウェア名をリストアップ。
-nオプションは、パッケージ名のみ表示させよという意味。--names-onlyでもよい。

```
cd ~/Documents/DRR054113
```

```
pwd
```

```
apt-cache -n search SRA  
apt-cache -n search SRA | wc
```

W4-1 : SRA Toolkit

①2016年3月22日現在の最新版はver. 2.5.7。②プログラムはOSの種類ごとに用意されている。Bio-Linuxの実体はUbuntuなので③ここ。話についてこれないヒトは、連載第1回を復習。④のリンク先が次のスライド

The screenshot shows the NCBI SRA Software page. At the top, there's a navigation bar with links for Main, Browse, Search, Download, Submit, Documentation, Software, Trace Archive, Trace Assembly, Trace Home, and Trace BLAST. Below this is a sub-navigation bar with Download, Toolkit Documentation, and XML Schema. The main content area is titled "SRA Toolkit". It contains a link "For Toolkit documentation [click here.](#)" with a red arrow labeled ④ pointing to it. Below this, a numbered list provides download links:

1. NCBI SRA Toolkit latest release (December 23 2015, version 2.5.7 release) compiled binaries and [md5 checksums](#):
 - [CentOS Linux 64 bit architecture](#)
 - [Ubuntu Linux 64 bit architecture](#)
 - [MacOS 64 bit architecture](#)
 - [MS Windows 64 bit architecture](#)
 - [vdb-view Windows Installer](#) is a spreadsheet-like browser for viewing SRA and vdb objects - Windows only
2. NCBI Decryption Tools latest release binaries and [md5 checksums](#):
 - [CentOS Linux 64 bit architecture](#)
 - [CentOS Linux 32 bit architecture](#)
 - [Ubuntu Linux 64 bit architecture](#)
 - [Ubuntu Linux 32 bit architecture](#)
 - [MacOS 64 bit architecture](#)
 - [MacOS 32 bit architecture](#)
 - [MS Windows 64 bit architecture](#)
 - [MS Windows 32 bit architecture](#)
3. Latest Source Code:
 1. [NGS Software Development Kit](#) – November 24 2015, version 1.2.3 release
 2. [NCBI VDB Software Development Kit](#) – December 23 2015, version 2.5.7 release
 3. [NCBI SRA Toolkit](#) – December 23 2015, version 2.5.7 release

W4-1 : SRA Toolkit

①目的のfastq-dumpプログラムは、②よく使われるツール群(Frequently Used Tools)の最初に位置する。fastq-dumpを利用したいがために、SRA Toolkitをインストールするヒトがほとんどであろう。基本的には③を参考にインストールする。クリック

http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=toolkit_doc

NCBI Site map All databases Search

Sequence Read Archive

Main Browse Search Download Submit Documentation Software Trace Archive Trace Assembly Trace Home Trace BLAST

Download Toolkit Documentation XML Schema

SRA Toolkit Documentation

[SRA Toolkit Installation and Configuration Guide](#) ③

[Protected Data Usage Guide](#) ②

Frequently Used Tools:

fastq-dump: Convert SRA data into fastq format ①

[prefetch](#): Allows command-line downloading of SRA, dbGaP, and ADSP data

[sam-dump](#): Convert SRA data to sam format

[sra-pileup](#): Generate pileup statistics on aligned SRA data

[vdb-config](#): Display and modify VDB configuration information

[vdb-decrypt](#): Decrypt non-SRA dbGaP data ("phenotype data")

Additional Tools:

[abi-dump](#): Convert SRA data into ABI format (csfasta / qual)

[illumina-dump](#): Convert SRA data into Illumina native formats (qseq, etc.)

[sff-dump](#): Convert SRA data to sff format

[sra-stat](#): Generate statistics about SRA data (quality distribution, etc.)

[vdb-dump](#): Output the native VDB format of SRA data.

[vdb-encrypt](#): Encrypt non-SRA dbGaP data ("phenotype data")

[vdb-validate](#): Validate the integrity of downloaded SRA data

W4-1 : SRA Toolkit

SRA Toolkit Installation and Configuration Guide

Table of Contents

1. Downloading and installing the SRA Toolkit
2. Testing the Toolkit configuration
3. Configuring the Toolkit
4. Links and help documents

Contact: sra-tools@ncbi.nlm.nih.gov

The following guide will outline the download, installation, and configuration of the SRA Toolkit. Detailed information regarding the usage of individual tools in the SRA Toolkit can be found on the tool-specific documentation pages.

The NCBI SRA Toolkit enables reading ("dumping") of sequencing files from the SRA database and writing ("loading") files into the .sra format (Note that this is not required for submission). The Toolkit source code is provided in the form of the [SRA SDK](#), and may be compiled with GCC. However, pre-built software executables are available for Linux, Windows, and Mac OS X, and we highly recommend using these pre-built executables whenever possible.

Downloading and installing the SRA Toolkit

Download the Toolkit from the SRA website

1. If you are using a web browser, the following page contains download links to the most current version of the toolkit for each of the supported platforms: SRA Toolkit download page: <http://www.ncbi.nlm.nih.gov/Traces/sra/?view=software>
2. If you are instead working from a command line interface, you may use FTP or wget to obtain the software from the following directory: "://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current". Example:

`wget "://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current/sratookit.current-centos_linux64.tar.gz"`

①

Unpack the Toolkit:

1. For Linux, use tar:
`tar -xzf sratookit.current-centos_linux64.tar.gz`

②

①wgetでtar.gzをダウンロードし、②解凍するやり方が示されているが…折角なので第4回W4-5, W13-5, W14, W15で紹介した「`sudo apt-get install ソフトウェア名`」でSRA Toolkitのインストールを行うやり方を伝授

W4-2: apt-cache

```
File Edit View Search Terminal Help  
iu@bielinux[DRR054113] pwd  
/home/iu/Documents/DRR054113  
iu@bielinux[DRR054113] apt-cache -n search SRA      [ 8:16午後 ]  
libsratom-0-0 - library for serialising LV2 atoms to/from Turtle  
libsratom-dev - library for serialising LV2 atoms to/from Turtle  
- development files  
libsratom-doc - library for serialising LV2 atoms to/from Turtle  
- documentation  
sra-toolkit - utilities for the NCBI Sequence Read Archive ②  
sra-toolkit-libs-dev - Development files for the NCBI SRA Toolkit's libraries  
sra-toolkit-libs0 - Libraries for the SRA Toolkit  
iu@bielinux[DRR054113] █
```

目的:「sudo apt-get install ソフトウェア名」で指定する名前を知りたい！やり方:「apt-cache -n search キーワード」で任意のキーワードを含むソフトウェア名をリストアップする。ここでは、①SRAを含むソフトウェア名をリストアップ。②欲しいソフトウェア名は、sra-toolkitだということがわかる

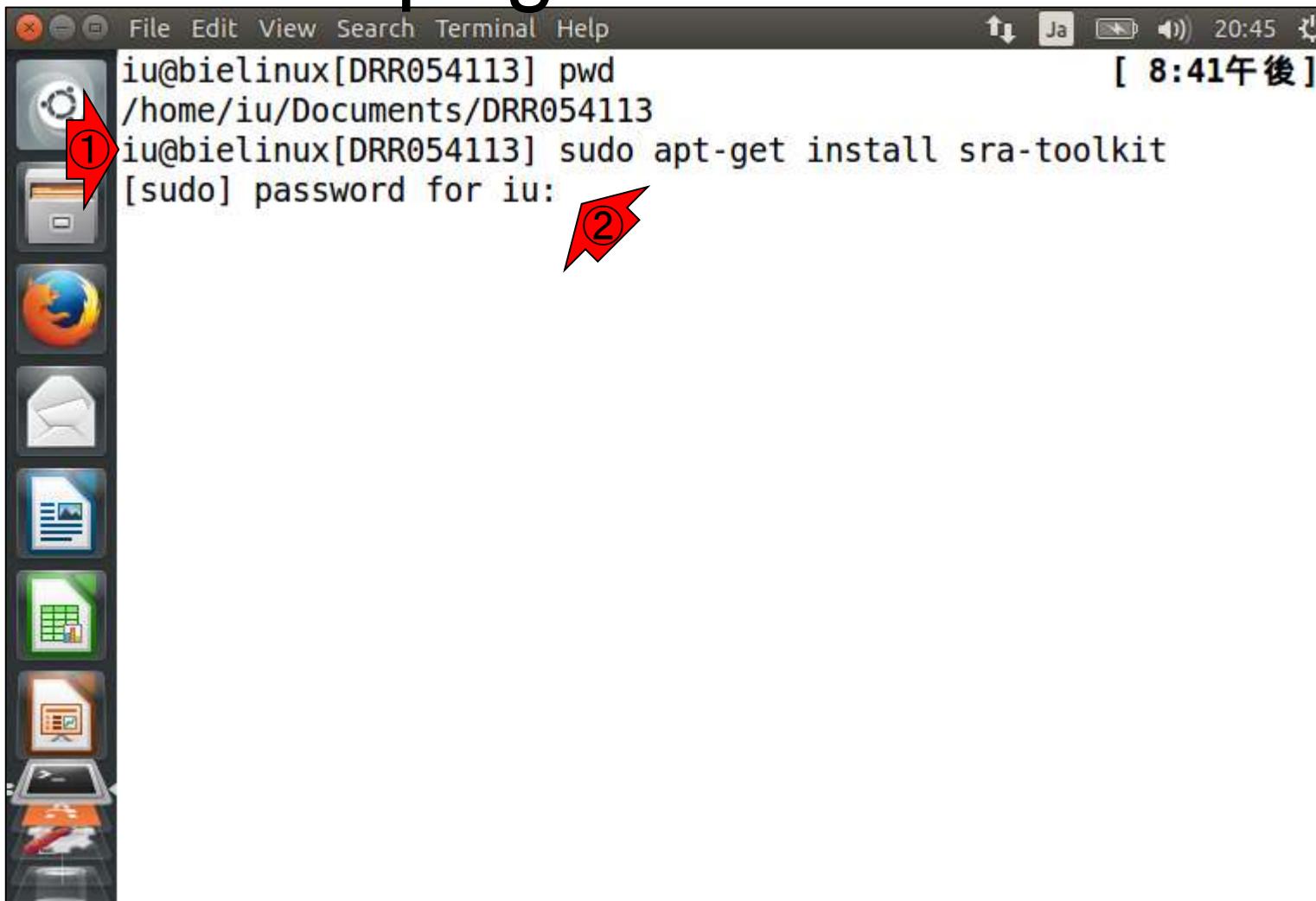
W4-2: apt-cache

```
File Edit View Search Terminal Help
iu@bielinux[DRR054113] pwd
/home/iu/Documents/DRR054113
iu@bielinux[DRR054113] apt-cache -n search SRA      [ 8:16 午後 ]
libsratom-0-0 - library for serialising LV2 atoms to/from Turtle
libsratom-dev - library for serialising LV2 atoms to/from Turtle
- development files
libsratom-doc - library for serialising LV2 atoms to/from Turtle
- documentation
sra-toolkit - utilities for the NCBI Sequence Read Archive
sra-toolkit-libs-dev - Development files for the NCBI SRA Toolkit's libraries
sra-toolkit-libs0 - Libraries for the SRA Toolkit
iu@bielinux[DRR054113] apt-cache -n search SRA | wc [ 8:16 午後 ]
      6      58     418
iu@bielinux[DRR054113]                                [ 8:29 午後 ]
```

おまけ。①「apt-cache -n search SRA」実行結果として、小文字のsraを含むソフトウェア名もリストアップされたことから、キーワード部分は、大文字でも小文字でもどちらでもいいのだろうと学習する。また、②「| wc」を追加することで、ソフトウェア名が6個だったと認識

①「sudo apt-get install sra-toolkit」。rootのパスワードを聞かれたら打ち込む(推奨手順通りだとpass1409)

W4-3: apt-get



```
iu@bielinux[DRR054113] pwd [ 8:41午後]
/home/iu/Documents/DRR054113
iu@bielinux[DRR054113] sudo apt-get install sra-toolkit
[sudo] password for iu:
```

W4-3: apt-get

①Do you want to continue?と聞かれるので、y。イチイチ聞かれたくない場合は、「sudo apt-get -y install sra-toolkit」と-yオプションをつけておけばよい(第4回W15-1)

```
iu@bielinux:~/Documents/DRR054113] Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-3.13.0-55 linux-headers-3.13.0-55-generic
  linux-headers-3.13.0-68 linux-headers-3.13.0-68-generic
  linux-headers-3.13.0-71 linux-headers-3.13.0-71-generic
  linux-image-3.13.0-55-generic linux-image-3.13.0-68-generic
  linux-image-3.13.0-71-generic linux-image-extra-3.13.0-55-generic
  linux-image-extra-3.13.0-68-generic linux-image-extra-3.13.0-71-generic
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  sra-toolkit-libs0
The following NEW packages will be installed:
  sra-toolkit sra-toolkit-libs0
0 upgraded, 2 newly installed, 0 to remove and 138 not upgraded.
Need to get 2,311 kB of archives.
After this operation, 6,065 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```



W4-3: apt-get

```
File Edit View Search Terminal Help
used.
Do you want to continue? [Y/n] y
Get:1 http://jp.archive.ubuntu.com/ubuntu
oolkit-libs0 amd64 2.1.7a-1ubuntu2 [924 kB]
Get:2 http://jp.archive.ubuntu.com/ubuntu
oolkit amd64 2.1.7a-1ubuntu2 [1,387 kB]
Fetched 2,311 kB in 0s (5,838 kB/s)
Selecting previously unselected package
(Reading database ... 439956 files and
talled.)
Preparing to unpack .../sra-toolkit-libs0_2.1.7a-1ubuntu2_amd64.
deb ...
Unpacking sra-toolkit-libs0 (2.1.7a-1ubuntu2) ...
Selecting previously unselected package sra-toolkit.
Preparing to unpack .../sra-toolkit_2.1.7a-1ubuntu2_amd64.deb ...
.
Unpacking sra-toolkit (2.1.7a-1ubuntu2) ...
Setting up sra-toolkit-libs0 (2.1.7a-1ubuntu2) ...
Setting up sra-toolkit (2.1.7a-1ubuntu2) ...
Processing triggers for libc-bin (2.19-0ubuntu6.7) ...
iu@bielinux[DRR054113] [ 8:48 午後 ]
```

①インストール完了後の状態。2016年06月19日に気づいたこととして、赤下線部分とこの後のスライドを眺めればわかるが、バージョンがかなり古い。2016年7月公開済みの原稿PDFではapt-getの手順を推奨しているが、“最新版”をインストールしたい場合は、「wget URL」などプログラムのバージョンを自分で把握なり取り扱える手段でやりましょう。SRA Toolkitはただのapt-getの練習台程度の位置づけしかなかったが、これもやったおかげで上記事実を知り、「こういうこともあるから気をつけねば」という経験を積むことができた。いろいろやるのは大事



W4-4：確認

The screenshot shows a terminal window open in a desktop environment. The terminal window title is 'File Edit View Search Terminal Help'. The terminal content is as follows:

```
iu@bielinux[DRR054113] pwd  
/home/iu/Documents/DRR054113  
iu@bielinux[DRR054113] fastq-dump  
  
Usage:  
  fastq-dump [options] [ -A ] <accession>  
  fastq-dump [options] <path [path...]>  
  
Use option --help for more information  
  
fastq-dump : 2.1.7  
  
iu@bielinux[DRR054113] where fastq-dump  
/usr/bin/fastq-dump  
iu@bielinux[DRR054113]
```

On the left side of the terminal window, there is a vertical docked application bar with several icons. Three specific icons are highlighted with red arrows and numbered 1, 2, and 3:

- Icon 1: A small terminal window icon.
- Icon 2: A file folder icon.
- Icon 3: A document with a grid icon.

①インストール作業は「~/Documents/DRR054113」で行ったが、「sudo apt-get install ソフトウェア名」でやる場合は、基本的にどの作業ディレクトリ上でもよい。②インストール後は、fastq-dumpを使用可能。③パスも既に通されている

[12:31午後]

[12:31午後]

[12:31午後]

W4-5: パスが通っている

SRA Toolkit Installation and Configuration Guide



Table of Contents

1. Downloading and installing the SRA Toolkit
2. Testing the Toolkit configuration
3. Configuring the Toolkit
4. Links and help documents

Contact: sra-tools@ncbi.nlm.nih.gov

The following guide will outline the usage of individual tools in the SRA Toolkit.

The NCBI SRA Toolkit contains files in the .sra format (Note that may be compiled with GC). We recommend using these pre-compiled packages.

Downloading and installing the Toolkit

Download the Toolkit from:

1. If you are using a web browser, download the supported platform's package.
2. If you are instead working in a terminal window, download the directory: "<http://ftp-trace.ncbi.nlm.nih.gov/sra/sratoolkit>"

Unpack the Toolkit:

1. For Linux, use tar:
`tar -xzf sratoolkit.current-centos_linux64.tar.gz`

Unpack the Toolkit:

1. For Linux, use tar:
`tar -xzf sratoolkit.current-centos_linux64.tar.gz`
2. For Mac OS X, double-click on the .tar.gz file and the Archive Utility will unpack it. Alternatively, command-line tar will also work (see Linux example, above).
3. For Windows, either use an archiving and compression utility (e.g., Winzip, 7-Zip, etc.), or simply double-click on the .zip file and drag the 'sratoolkit...' folder to the preferred install location.

Note: For most users, the Toolkit functions (fastq-dump, sam-dump, etc.) will not be located in their [PATH environmental variable](#). This may require providing directory information about the location of the Toolkit. See the below examples for how 'fastq-dump' would be called in different circumstances:

- `~/[user_name]/sra-toolkit/fastq-dump`
YES: The Toolkit "bin" directory has been placed in the user-specified directory "sra-toolkit"
- `./fastq-dump`
YES: The Toolkit components are in the current working directory
- `fastq-dump`
NO: If the toolkit location is not specified in your \$PATH variable, then the OS cannot locate the fastq-dump program, even if it is in the current directory. NOTE: Windows users should be able to enter only "fastq-dump.exe" if you have navigated to the Toolkit "bin" directory.



「`sudo apt-get install sra-toolkit`」で無事インストール完了したあとは、W4-4で示したようにパスを通し終わった状態。それゆえ①SRA Toolkit Installation and Configuration Guide中の、②パスに関する注意書きは、気にしなくてもよい

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6: FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W5-6: デフォルトで実行

```
File Edit View Search Terminal Help
/home/iu/Documents/DRR054113
iu@bielinux[DRR054113] ls
DRR054113_DRA.fastq.bz2 DRR054113.fastq
DRR054113_DRA.fastq.gz DRR054113.sra
iu@bielinux[DRR054113] rm -f *.fastq
iu@bielinux[DRR054113] ls -l
total 1389816
-rw-rw-r-- 1 iu iu 2392259 3月 22 12:32 DRR054113_DRA.fastq.bz2
-rw-rw-r-- 1 iu iu 2720482 3月 22 12:38 DRR054113_DRA.fastq.gz
-rw-rw-r-- 1 iu iu 1418046334 3月 22 15:23 DRR054113.sra
iu@bielinux[DRR054113] fastq-dump ./DRR054113.sra ① [ 5:14 午後 ]
Written 163380 spots for ./DRR054113.sra
Written 163380 spots total
iu@bielinux[DRR054113] ls -l ③ [ 5:15 午後 ]
total 2102944
-rw-rw-r-- 1 iu iu 2392259 3月 22 12:32 DRR054113_DRA.fastq.bz2
-rw-rw-r-- 1 iu iu 2720482 3月 22 12:38 DRR054113_DRA.fastq.gz
-rw-rw-r-- 1 iu iu 730238332 3月 23 17:15 DRR054113.fastq ②
-rw-rw-r-- 1 iu iu 1418046334 3月 22 15:23 DRR054113.sra
iu@bielinux[DRR054113] wc DRR054113.fastq ④ [ 5:15 午後 ]
 653520   980280  730238332 DRR054113.fastq
iu@bielinux[DRR054113] [ 5:19 午後 ]
```

ここはダイジェスト版のみで眺めるだけ(詳細は第7回ウェブ資料PDF)。①sraファイルを入力としてfastq-dumpを実行すると、②fastqファイルが得られる。リード数情報は、③を眺めるのでもわかる。④fastqファイルの行数(653,520)から辿れるリード数($653,520 / 4 = 163,380$ 個)と同じ。DRA上で見られる数値(163,482リード; W2-5)と若干異なる理由は不明

W6-3: 結果を眺める

FastQC実行結果。①入力は、さきほどのgzip圧縮ファイル。②リード数は163,380、③配列長は116-28874 bpの範囲であることがわかる

FastQC Report

Thu 24 Mar 2016
DRR054113.fastq.gz

Summary

Basic Statistics

Per base sequence quality

Per sequence quality scores

Per base sequence content

Per sequence GC content

Per base N content

Sequence Length Distribution

Sequence Duplication Levels

Overrepresented sequences

Adapter Content

Basic Statistics

Measure	Value
Filename	DRR054113.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	163380
Sequences flagged as poor quality	0
Sequence length	116-28874
%GC	44

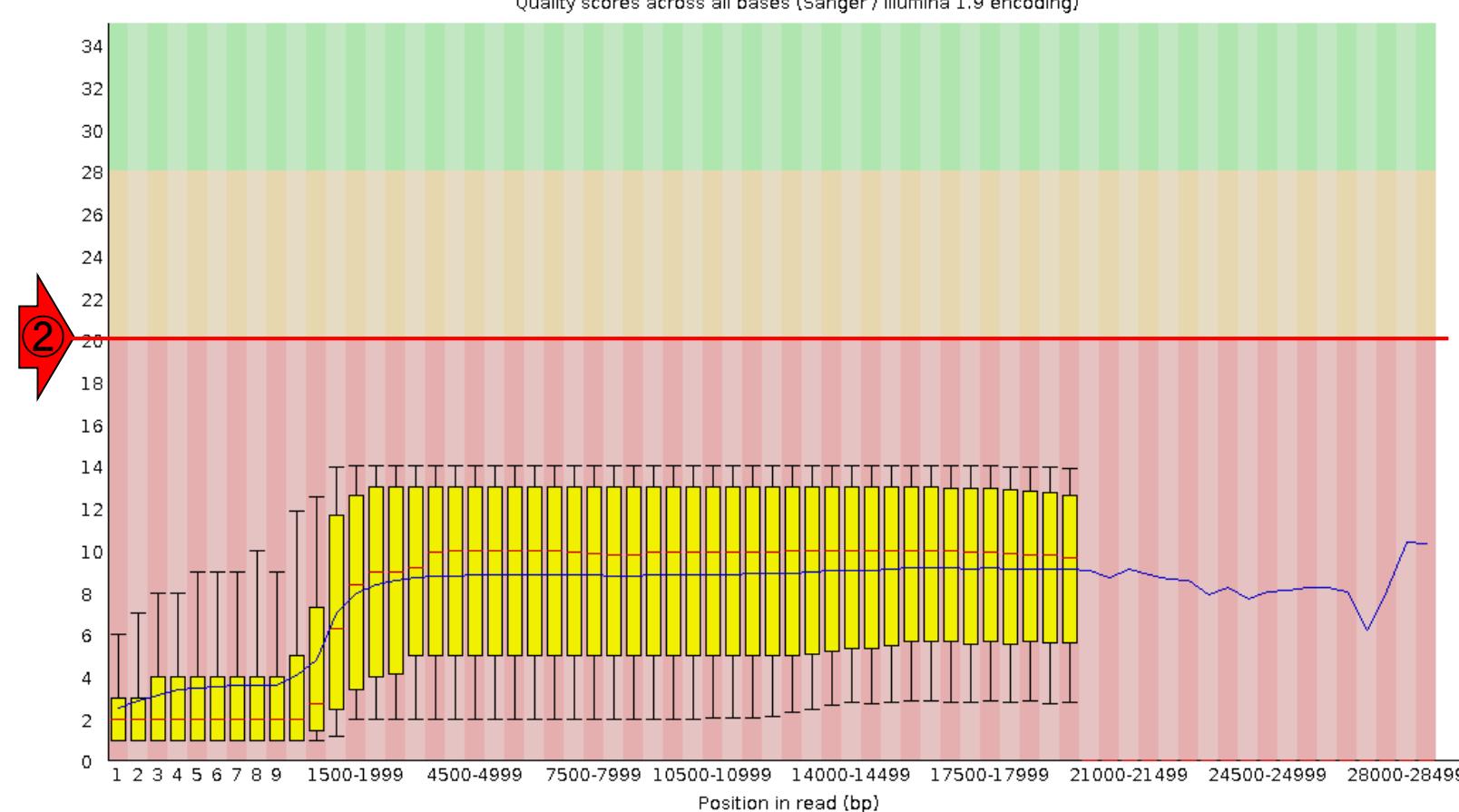
W6-3: 結果を眺める

FastQC Report

Summary

- Basic Statistics
- Per base sequence quality ①
- Per sequence quality scores
- Per base sequence content
- Per sequence GC content
- Per base N content
- Sequence Length Distribution
- Sequence Duplication Levels
- Overrepresented sequences
- Adapter Content
- Kmer Content

Per base sequence quality



W6-4: Rで計算

塩基配列決定精度(エラー率)からクオリティスコアをRで計算する。①エラー率13%のときは、スコア8.86となるので、実際のスペック通りで安心

```
File Edit View Search Terminal Help [ 4:32 午後 ]
iu@bielinux[DRR054113] R -q
> -10*log10(0.01)      #エラー率1%のときのクオリティスコア
[1] 20
> -10*log10(0.10)      #エラー率10%のときのクオリティスコア
[1] 10
> -10*log10(0.13)      #エラー率13%のときのクオリティスコア
[1] 8.860566
> q(save="no")
iu@bielinux[DRR054113] [ 4:33 午後 ]
```



W6は

①のあたりです。②FastQC実行結果のhtmlファイルはここにあるので、じっくり眺めたいヒトはどうぞ。~/Desktop/backupにもあります

PacBioデータの概観2

- W6-1: [FastQC ver. 0.11.4](#)実行

FastQC ver. 0.11.4は、第4回W9-2でインストールし、`fastqc2`というコマンドでパスを通している。

```
cd ~/Documents/DRR054113
```

```
pwd  
ls -l  
fastqc2 -v  
fastqc2 -q DRR054113.fastq.gz  
ls -l
```

- W6-2: 改名して移動

共有フォルダ([~/Desktop/mac_share](#))内にW3-2で作成した同じファイル名のものが存在する。このため、`mv`コマンドで共有フォルダ([~/Desktop/mac_share](#))に移動させる際に、(163380リードからなるファイルの結果という意味で) [_163380](#)をファイル名に追加している。`.zip`ファイルは実質的に不要ではあるが、一応同時に移動させている。

```
pwd  
ls -l *fastqc*  
ls -l ~/Desktop/mac_share  
mv DRR054113_fastqc.html ~/Desktop/mac_share/DRR054113_163380_fastqc.html  
mv DRR054113_fastqc.zip ~/Desktop/mac_share/DRR054113_163380_fastqc.zip  
ls -l ~/Desktop/mac_share
```

- W6-3: 結果を眺める

[DRR054113_163380_fastqc.html](#)

- W6-4: Rで計算

塩基配列決定精度が87%(エラー率13%)のときのクオリティスコアは8.86。

```
R -q  
-10*log10(0.01)    #エラー率1%のときのクオリティスコア  
-10*log10(0.10)    #エラー率10%のときのクオリティスコア  
-10*log10(0.13)    #エラー率13%のときのクオリティスコア
```

Contents

- W11:ゲノムサイズ推定(KmerGenieのインストールと利用)
 - インストール、single-endで実行(結果の解説)、paired-endで実行(結果の解説)
- W12:配列長によるフィルタリング(Pythonプログラムの利用)
- DDBJ Pipeline(W13からW17まではほぼ省略)
 - W18:k=131でのVelvet実行結果の解析
 - W19:Platanusの実行、W20:結果の解析、W21:ACGTカウント(塩基ごとの出現頻度解析)
- ロングリード(PacBio)データと公共DB
 - W2: PacBio生データはbax.h5形式、公共DBはsra形式とFASTQ形式
 - W3: 公共DB (DRA)のFASTQファイルを入力としてFastQC
 - W4: NCBI SRA (SRA)が提供するSRA Toolkitのインストール
 - W5: 利用(.sra → .fastqへの変換)、W6: FastQC
- DDBJ PipelineでHGAPを実行
 - W7:DDBJ Pipelineに解析したい生データファイル(.bax.h5)をアップロード
 - W8:DDBJ PipelineでHGAPを実行
 - W9:HGAPアセンブリ結果を眺める



W2-7:bax.h5ファイル

PacBioのファイル形式とデータ解析の概要

- W1-1: PacificBiosciencesの YouTube サイト
 - [Introduction to SMRT Sequencing](#)
 - [Single Molecule Real Time Sequencing](#)
- W2-1: PacBioデータ(原著論文中のDRR IDだが削除されている)
 - [DRR024500: Tanizawa et al., BMC Genomics, 2015](#)
- W2-3: [DRR024501](#) → [DRP002401](#) → [DRX022185](#)
- W2-4: [DRR024501](#) → [DRA002643](#)
- W2-5: PacBioデータ概観
 - [DRR054113](#)
 - [DRR054114](#)
 - [DRR054115](#)
 - [DRR054116](#)
- W2-6: SMRT Portal(PacBio提供のHGAPを含む解析ソフトウェア群)の場所
 - [PacBio](#) → [DevNet](#) → [SMRT Analysis](#)
 - SMRT Analysis 2.3までは、HGAPを実行するためにはbax.h5ファイルが必須。
 - SMRT Analysis 3.0からは、BAMファイルが入力フォーマットになる。但しここでのBAMファイルは、マッピングデータではなく、シークエンス生データ。
 - PacBio RSIIの後継機である[Sequel](#)の出力ファイル形式はBAM。
 - PacBioのファイル形式の説明については[こちら](#) (<http://pacbiofileformats.readthedocs.io/en/3.0/>)。
- W2-7: [DRR054113](#)のbax.h5ファイル (下記3ファイル合わせてDRR054113に相当)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.1.bax.h5](#) (747 MB; 784,301,199 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.2.bax.h5](#) (766 MB; 803,938,042 bytes)
 - [m130821_065825_42195_c100539522550000001823089611241356_s1_p0.3.bax.h5](#) (901 MB; 945,597,712 bytes)

おさらい。DDBJ Pipeline上では、PacBio用の*de novo*アセンブリ用プログラム HGAPを利用可能。しかし、①HGAPは bax.h5ファイルのみしか受け付けない

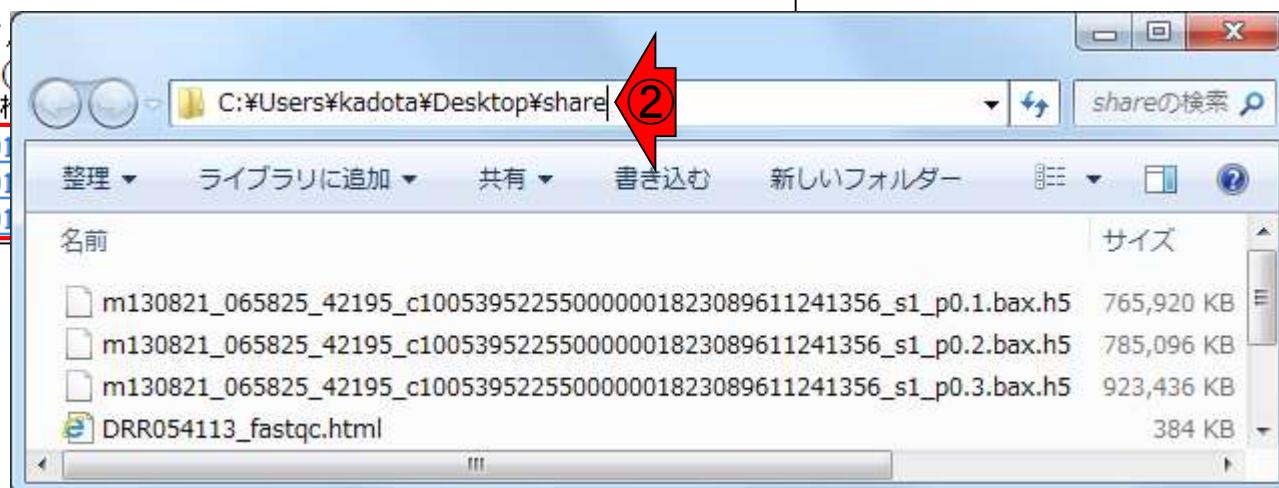


W7-1:bax.h5ファイル準備

PacBioのファイル形式とデータ解析の概要

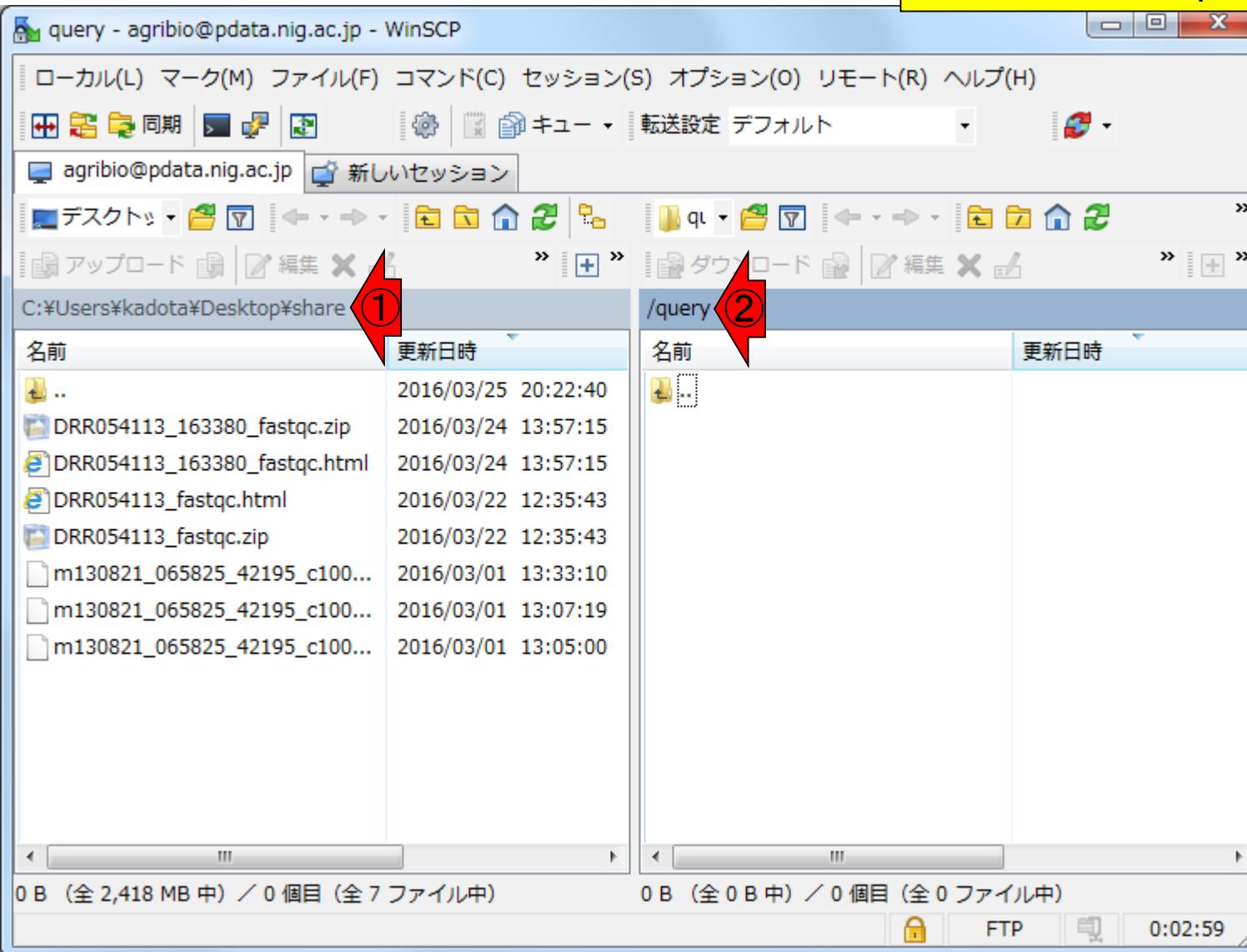
- W1-1: PacificBiosciencesの YouTube サイト
 - [Introduction to SMRT Sequencing](#)
 - [Single Molecule Real Time Sequencing](#)
- W2-1: PacBioデータ(原著論文中のDRR IDだが削除されている)
 - [DRR024500: Tanizawa et al., BMC Genomics, 2015](#)
- W2-3: [DRR024501](#) → [DRP002401](#) → [DRX022185](#)
- W2-4: [DRR024501](#) → [DRA002643](#)
- W2-5: PacBioデータ概観
 - [DRR054113](#)
 - [DRR054114](#)
 - [DRR054115](#)
 - [DRR054116](#)
- W2-6: SMRT Portal(PacBio提供のHGAPを含む解析ソフトウェア群)の場所
 - [PacBio](#) → [DevNet](#) → [SMRT Analysis](#)
 - SMRT Analysis 2.3までは、HGAPを実行するためには**bax.h5**ファイルが必須。
 - SMRT Analysis 3.0からは、BAMファイルが入力フォーマットになる。但しここでのBAMファイルは、マッピングデータではなく、シークエンス生データ。
 - PacBio RSIIの後継機である[Sequel](#)の出力ファイル
 - PacBioのファイル形式の説明については[こちら](#)
- W2-7: [DRR054113](#)の**bax.h5**ファイル (下記3ファイル合計)
 - [m130821_065825_42195_c1005395225500000001.bax.h5](#)
 - [m130821_065825_42195_c1005395225500000001823089611241356_s1_p0.1.bax.h5](#)
 - [m130821_065825_42195_c1005395225500000001823089611241356_s1_p0.2.bax.h5](#)
 - [m130821_065825_42195_c1005395225500000001823089611241356_s1_p0.3.bax.h5](#)
 - [DRR054113_fastqc.html](#)

①DDBJ Pipelineにアップロードしたい**bax.h5**ファイルを手元のPCにダウンロードしておく。②ここではDesktop上のshareフォルダにダウンロード。ファイルサイズは約2.4GBに達するため、それなりに時間はかかるだろう。もちろん講習会ではやりません！見るだけ！



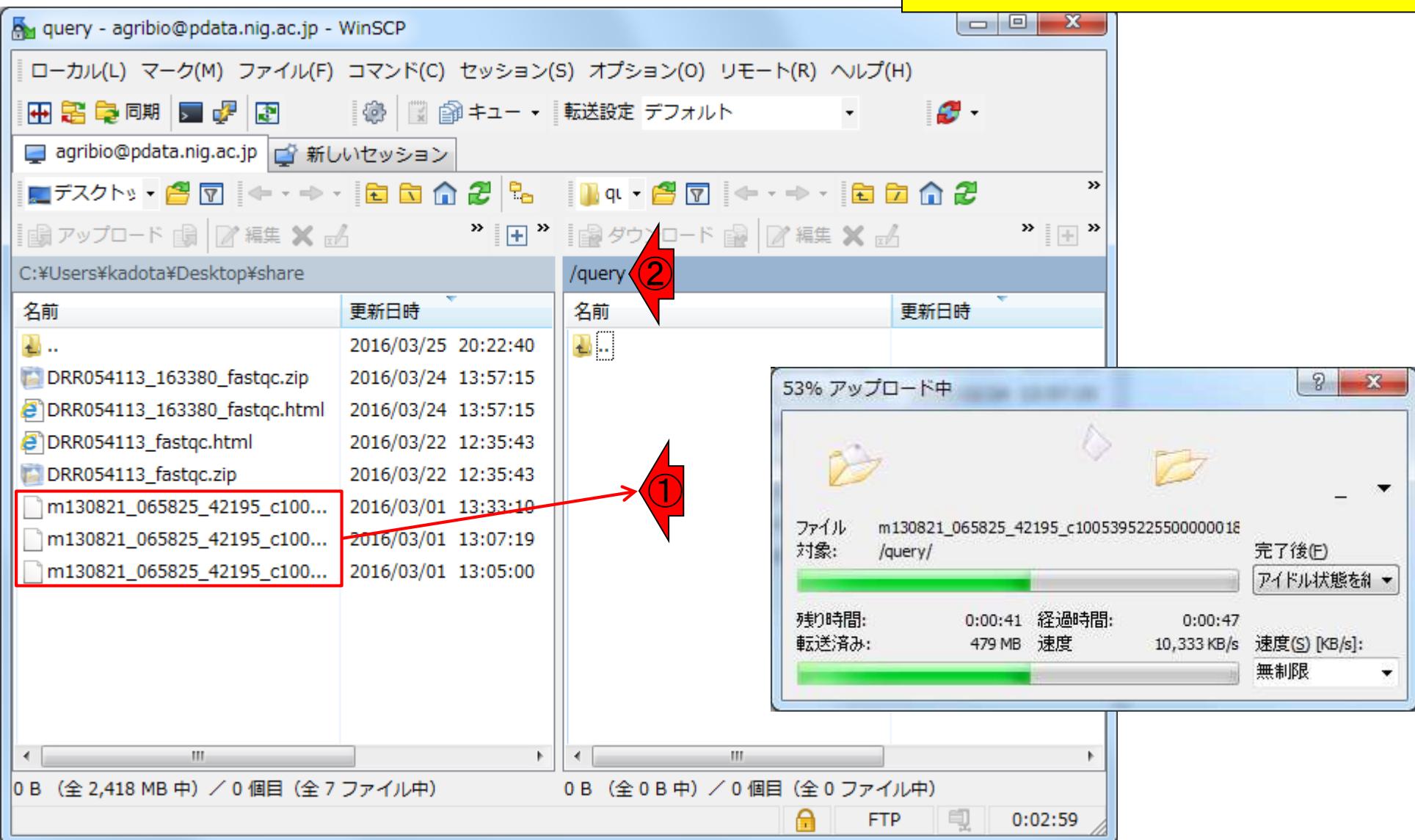
W7-2: アップロード

WinSCPを用いてDDBJ Pipelineの実体である遺伝研スパコンにログインし、①「デスクトップ – share」、②queryフォルダに移動した状態



W7-2: アップロード

①アップロードしたい3つのファイルを
②queryフォルダにドラッグ&ドロップ。
アップロードもそれなりに時間がかかる



W8-1 : HGAP実行

② de novo Assembly

③ HGAP

④ NEXT

Tool	Help	Version	Base space	Color space	Paired-end	MSS (WGS)	Comment
SOAPdenovo		2.04-r240	✓		✓		
ABySS		1.3.2	✓		✓		Maximum K-mer value is 64.
Velvet		1.2.10	✓		✓	✓	We severely recommend when performing Velvet, total length of those reads is up to 22G bp. Maximum K-mer value is 64.
Trinity		2.1.1	✓		✓		RNA-Seq De novo Assembly
Platanus		1.2.2	✓		✓		
HGAP		Protocol3 (v 2.2.0)					HGAP Pipeline for PacBio Sequence based on SMRT Analysis v2.2.0. For bax.h5 file only. (Beta version)

Mapping Contigs by de novo Assemble to Reference Sequences.
The contigs will be aligned to reference genome.

Tool	Comment
BLAT	Single-end analysis only

BACK NEXT

W8-1 : HGAP実行

① 解析したい3つのファイルに
チェックを入れて、②confirm

http://p.ddbj.nig.ac.jp/pipeline>SelectQuery.do Generating Query Sets f... x

DDBJ DNA Data Bank of Japan

Select Query Files → Select Tools → Set QuerySet → Set Ass. Options → Confirmation → Running Status

ACCOUNT login ID [agribio] Logout Change password

ANALYSIS Data setup DRA Start FTP upload HTTP upload DRA Import Preprocessing Start step-1 Preprocessing Mapping / de novo Assembly step-2 Workflow Genome (SNP/Short Indel) RNA-seq (Tag count) ChIP-seq

JOB STATUS step1. Preprocessing step1. Mapping step1. de novo Assembly step2-All status

HELP HELP TUTORIAL Contact Us. DDBJ Read Annotation

Generating Query Sets from Query Read Files

RESET BACK NEXT

Single analysis Layout of single sequence.

5' Linker(1) Target Linker(2) 3'

	Run	ACCESSION	Read length	Quality Score
<input checked="" type="checkbox"/>	m130821_065825_42195_c100539522550000001823089611241356_s1_p0.3.bax.h5	bp		
<input checked="" type="checkbox"/>	m130821_065825_42195_c100539522550000001823089611241356_s1_p0.2.bax.h5	bp		
<input checked="" type="checkbox"/>	m130821_065825_42195_c100539522550000001823089611241356_s1_p0.1.bax.h5	bp		

① (Red arrow) Check the boxes for the three files you want to analyze.
② (Red arrow) Click the "confirm" button.

QUERY SET

RESET BACK NEXT

①と②、2つの指定可能なパラメータがあります

W8-2:2つのパラメータ

The screenshot shows the DDBJ Pipeline interface for 'Setting for De Novo Assembly'. The workflow steps are: Select Query Files, Select Tools, Set QuerySet, Set Ass. Options (highlighted in orange), Confirmation, and Running Status.

Setting for De Novo Assembly

hgap

Set optional parameters for HGAP pipeline

Select UGE-node to run :

month_fat (32 CPUs and 320GB memory)
 month_medium (32 CPUs and 256GB memory)

Same results will be generated with either option.
You can check the CPU and memory usage at [NIG-SC Website](#).

1 : The approximate genome size, in base pairs.(Must be a value between 1 and 150000000)

GenomeSize =

2 : The minimum length of reads (in base pairs) to use as seeds for pre-assembly.

Minimum Seed Length :

Automatic Estimation
If the coverage exceeds 30X, the Minimum Seed Read Length that results in at least 30X coverage by the longest subreads will be calculated automatically. If the coverage is less than 30X, the user-specified value will be used.

Use Manually Specified Value (regardless of the coverage)

BACK NEXT

ACCOUNT
login ID [agribio]
Logout Change password

ANALYSIS
Data setup
DRA Start
FTP upload
HTTP upload
DRA Import
Preprocessing Start
step-1
Preprocessing
Mapping / de novo Assembly
step-2
Workflow
Genome (SNP/Short Indel)
RNA-seq (Tag count)
ChIP-seq

JOB STATUS
step1. Preprocessing
step1. Mapping
step1. de novo Assembly
step2-All status

HELP
HELP
TUTORIAL
Contact Us. DDBJ Read Annotation

W8-2: パラメータの解説

<https://github.com/PacificBiosciences/Bioinformatics-Training/wiki/HGAP%20in%20SMRT%20Analysis>

Personal Open source Business Explore Pricing

PacificBiosciences / Bioinformatics-Training

Code Issues 7 Pull requests 0 Wiki Pulse

HGAP in SMRT Analysis

Ihon edited this page on Jun 10 2014 · 2 revisions

This page contains information about the current release of HGAP.

There have been multiple iterations of the HGAP implementation in performance improvements added to each iteration. In SMRT Analysis introduced, significantly speeding up HGAP execution. In most cases HGAP.3 makes it the preferred protocol. In production environment We recommend using the latest version of SMRT Analysis to ensure performance with HGAP.

SMRT Analysis v2.1 has a new implementation of HGAP that speeds This is found in the `RS_HGAP_Assembly.2` protocol. `RS_HGAP_Assembly.2` Analysis v2.2.0 and later versions.

SMRT Analysis v2.2 contains a further improvement to HGAP, in which stage is sped up, this new protocol is named `RS_HGAP_Assembly.3`. Versions 2 and 3 is largely the same.

Important parameters

1. Genome Size

To accurately determine the Minimum Seed Read Length and the coverage of trimmed preassembled reads going into the assembly step, it is important to adjust the target genome size as accurately as possible.

2. Automatic Minimum Seed Read Length calculation

The Minimum Seed Read Length that results in at least 30X target genome coverage by the longest subreads is being calculated automatically (the default option). To use the user-selected Minimum Seed Read Length, the default option has to be **deselected**. If less than 30X coverage is being used for the HGAP process, the algorithm will use the user-selected Minimum Seed Length (6kb default), so lowering the default setting to 500bp is required to allow all-vs-all PreAssembly at lower than 30X coverage.

Genome Size

At the moment, HGAP in SMRT Analysis supports genomes up to 130 MB; further improvements to scaling the workflow will enable support for larger genomes.

Older versions of SMRT Analysis may have lower genome size limits. SMRT Analysis 2.0 was limited to a 10 Mb genome size. We do not recommend using older versions of SMRT Analysis since they can have significant performance limitations; please upgrade if possible.

Usage notes

For microbial assemblies we have seen improved assembly results using the latest workflows

W8-3: HGAP実行

①ゲノムサイズは、乳酸菌の平均的なゲノムサイズである2.5MB。②minimum lengthは、Automatic Estimation(デフォルト)を指定して③NEXT

PDF原稿p104左下

The screenshot shows the DDBJ Pipeline interface for 'SettingAssembly.do'. The main title is 'Setting for De Novo Assembly'. The navigation bar at the top includes 'Select Query Files', 'Select Tools', 'Set QuerySet', 'Set Ass. Options' (which is highlighted in orange), 'Confirmation', and 'Running Status'. On the left, there are sections for 'ACCOUNT' (login ID: agribio), 'ANALYSIS' (Data setup, DRA Start, FTP upload, HTTP upload, DRA Import, Preprocessing Start, step-1, Preprocessing, Mapping / de novo Assembly, step-2), 'Workflow' (Genome (SNP/Short Indel), RNA-seq (Tag count), ChIP-seq), 'JOB STATUS' (step1. Preprocessing, step1. Mapping, step1. de novo Assembly, step2-All status), and 'HELP' (HELP, TUTORIAL, Contact Us, DDBJ Read Annotation). The main content area is titled 'hgap' and 'Set optional parameters for HGAP pipeline'. It asks to 'Select UGE-node to run :'. Two options are shown: 'month_fat (32 CPUs and 320GB memory)' (radio button not selected) and 'month_medium (32 CPUs and 256GB memory)' (radio button selected). A note says 'Same results will be generated with either option. You can check the CPU and memory usage at [NIG-SC Website](#)'. Below this, section 1 asks for 'The approximate genome size, in base pairs. (Must be a value between 1 and 150000000)' with an input field containing '2500000'. Section 2 asks for 'The minimum length of reads (in base pairs) to use as seeds for pre-assembly.' with an input field containing '6000'. It has two radio button options: 'Automatic Estimation' (selected) and 'Use Manually Specified Value (regardless of the coverage)'. At the bottom right are 'BACK' and 'NEXT' buttons. Three red arrows point to these buttons: arrow 1 points to the 'GenomeSize' input field, arrow 2 points to the 'Minimum Seed Length' input field, and arrow 3 points to the 'NEXT' button.

W8-3: HGAP 実行

Run Confirmation

Destination of mail
When the request is completed, the system sends an email to this address.
kadota@bi.a.u-tokyo.ac.jp * Required

Result files will be deleted 60 days after submission.

Assembly [hgap]

PairedOrientation	RunAccession	RunAlias	RowLength	QualityScore1	QualityScore2
single	21144	L.hokkaidonensis.PacBio3			
single	21143	L.hokkaidonensis.PacBio2			
single	21142	L.hokkaidonensis.PacBio1			

Assembly commands
hgap

Set optional parameters for HGAP pipeline

Select UGE-node to run :

month_fat (32 CPUs and 320GB memory)
 month_medium (32 CPUs and 256GB memory)

Same results will be generated with either option.
You can check the CPU and memory usage at [NIG-SC Website](#).

1 : The approximate genome size, in base pairs.(Must be a value between 1 and 150000000)
GenomeSize = 2500000

Web ページからのメッセージ

Do you really want to execute pipeline programs?

① RUN (Red Arrow pointing up to the RUN button)

② OK (Red Arrow pointing down to the OK button)

W9-2: 結果を眺める

The screenshot shows the DDBJ Pipeline Detail View interface. On the left, there's a sidebar with navigation links like 'ACCOUNT', 'ANALYSIS' (selected), 'JOB STATUS', 'HELP', and 'TUTORIAL'. The main area has tabs for 'Detail view', 'Job info', 'Download modified queries', 'Download wgs file', 'Assembly statistics' (selected), and 'Time'. The 'Assembly statistics' tab displays the following data:

Contig #	: 4
Total contig size	: 2,433,614
Maximum contig size	: 2,289,497
Minimum contig size	: 11,372
N50 contig size	: 2,289,497

Red annotations with numbers 1 through 4 point to specific parts of the interface:

- 1**: Points to the 'Help' link in the sidebar.
- 2**: Points to the 'Job info' tab.
- 3**: Points to the 'Assembly statistics' table.
- 4**: Points to the red box highlighting the assembly statistics table.

① *de novo* Assembly、② Job ID番号(21965)を頼りにすれば、このページに辿り着ける。
 ③赤枠部分を見ると、④コンティグ数は4つ。
 このデータの正解は3つ(1 chromosome and 2 plasmids)。世間一般の評価通りのロングリード(PacBio)の長所がよく表れた結果

W9-2: 結果を眺める

①Total contig sizeは、乳酸菌の一般的なゲノムサイズと近く妥当。②Maximum contig sizeのものが全体の9割以上を占めていることから、これが乳酸菌の染色体ゲノムなのだろうと妄想する

http://p.ddbj.nig.ac.jp/pipeline/DetailView.do?query_set_id=21965

Detail view

Select Query Files → Select Tools → Set QuerySet → Set Ass. Options → Confirmation → Running Status

Detail view

Job info

- ID: 21965
- Tool (Version): HGAP (Protocol3(v 2.2.0))

RunAccession or Filename Download

m130821_065825_42195_c10053952255000001823089611241356_s1_p0.3.bax.h5	m130821_065825_42195_c10053952255000001823089611241356_s1_p0.3.bax.h5
m130821_065825_42195_c10053952255000001823089611241356_s1_p0.2.bax.h5	m130821_065825_42195_c10053952255000001823089611241356_s1_p0.2.bax.h5
m130821_065825_42195_c10053952255000001823089611241356_s1_p0.1.bax.h5	m130821_065825_42195_c10053952255000001823089611241356_s1_p0.1.bax.h5

Download modified queries

The modified query file does not exist, because of the following reasons.

- The file is expired. (about 1 months)
- Job is waiting for execution queue.
- Error in query file.

Download wgs file

- [out_WGS.fasta.gz](#) (Original size 2.4 MB)

Assembly statistics

Config # : 4
Total contig size : 2,433,614
Maximum contig size : 2,289,497
Minimum contig size : 11,372
N50 contig size : 2,289,497

Time

Wait time	Start time	End time
0: 0:11	2016-03-28 20:14:25	2016-03-29 19:13:20

Command Start time End time Log1 Log2 Result MD5

run HGAP through smrtpipe.py : GenomeSize=2500000,minSeedLength=6000	2016-03-28 20:14:25	2016-03-29 19:12:37	View	Download(13.1 MB)	MD5
---	------------------------	------------------------	----------------------	-----------------------------------	---------------------

BACK Top of page

Contig # : 4
Total contig size : 2,433,614
Maximum contig size : 2,289,497
 Minimum contig size : 11,372
 N50 contig size : 2,289,497

①

②

W9-3: ダウンロード

①result.zipというzip圧縮ファイルをダウンロードして、その後の解析へと進んでいく

Detail view

Job info

ID 21965	Tool (Version) HGAP (Protocol3(v 2.2.0))	RunAccession or Filename m130821_065825_42195_c10053952255000001823089611241356_s1_p0.3.bax.h5 m130821_065825_42195_c10053952255000001823089611241356_s1_p0.2.bax.h5	Download
-------------	---	--	----------

Command	Start time	End time	Log1	Log2	Result	MD5
run HGAP through smrtpipe.py : GenomeSize=2500000,minSeedLength=6000	2016-03-28 20:14:25	2016-03-29 19:12:37	View		Download(13.1 MB)	MD5

JOBS STATUS

step1. Preprocessing
step1. Mapping
step1. de novo Assembly

TIME

Wait time 0: 0:11	Start time 2016-03-28 20:14:25	End time 2016-03-29 19:13:20
----------------------	-----------------------------------	---------------------------------

Command	Start time	End time	Log1	Log2	Result	MD5
run HGAP through smrtpipe.py : GenomeSize=2500000,minSeedLength=6000	2016-03-28 20:14:25	2016-03-29 19:12:37	View		Download(13.1 MB)	MD5

この後の展開は…

- W10: multi-FASTAファイルの分割
 - プログラムによっては、single-FASTAのほうが取扱いやすい場合もある
- W11: FASTQファイルの分割とクオリティスコア分布
 - FASTAファイル用がうまく動かなくても、4行で1リードだということが分かっていれば、Linuxコマンドでどうにかなる、という話
 - PacBioはFASTQファイルも出力する。Rで(single-)FASTQファイルを読み込んでクオリティスコア分布を描画し、PacBioデータは両末端のクオリティが低い傾向にあることを確認
- 環状化(ゲノム解読のfinishing作業の一部)
 - アセンブリ結果として、最初と最後の末端部分が同じ配列の場合は、通常そのコンティグは環状と判断。それを確認するための基本的な考え方、手段、および環状化のノウハウを伝授
 - W12: seqinrパッケージを用いて、仮想環状コンティグのドットプロットで感覚をつかむ
 - W13: 重複配列の除去の感覚をつかむ(これが環状化作業の実体)
 - W14: dotterプログラムで実際のPacBio出力結果に対して適用し、環状状態の概要を知る
 - W15: Bio-Linux上のblastnで、環状の同一コンティグ同士をDB側とquery側にして実行
 - W16: 正確なアラインメントを眺め、切断箇所をクオリティスコア分布と合わせて判断する
 - W17: 両端切断後のコンティグに対し、クオリティスコア分布やdotterを再度実行して確認
 - W18: NCBI blastのやり方も示し、様々な解析手段を伝授