2016.07.07版

de novoアセンブリの理論の話（スライド134-147あたり）は、当日省略予定です。が、第3部初日(2016年08月01日）の、k-mer出現頻度分布の解釈や、シークエンスエラー由来k-merの除去などとも関連しますので、必要に応じて予習（または復習）しておいてください。W8(スライド159-168あたり)も状況次第で省く

第3部：NGS解析（中～上級）

～ Linux環境でのデータ解析：マッピング、トリミング、アセンブリ～

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/

忘れたらココを思い出してね

利用プログラムの簡単な解説

- Biostrings
  - 塩基配列の各種解析を行うためのRパッケージ。トリミング用として利用
- 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
- FASTX-Toolkit
  - FASTAやFASTQ形式ファイルの簡単な処理を行うためのツール群。fastx_trimmerを利用
- QuasR (Gaidatzis et al., *Bioinformatics*, 2015)
  - (主に)マッピングからカウント情報取得まで行ってくれるRパッケージ
- Rockhopper2 (Tjaden, B., *Genome Biol.*, 2015)
  - バクテリア用*de novo*トランスクリプトームアセンブラ
- Velvet (Zerbino and Birney, *Genome Res.*, 2008)
  - *de novo*ゲノムアセンブラ

おさらい

①Rockhopper2の結果に苦悩しつつ、気を取り直してRパッケージQuasRによる乳酸菌リファレンスゲノム配列へのマッピングを行うべく、Linux環境でのRの基本的な利用法を学習したのが2016年08月01日

- オリジナル(SRR616268)
  - 乳酸菌paired-end RNA-seqデータで、最初の100万リードのみ抽出
  - forward側(SRR616268sub_1.fastq.gz)のリード長は107 bp
  - reverse側(SRR616268sub_2.fastq.gz)のリード長は93 bp

- FaQCs実行結果(W1-1)
  - 1,000,000リード → 977,202リード (W1-3)
  - forward側(QC.1.trimmed.fastq)
  - reverse側(QC.2.trimmed.fastq)
  - リード長はバラバラ。FastQC上で見られるIllumina adapterは消滅状態
- *de novo*トランスクリプトームアセンブリ(Rockhopper 2)実行結果
  - paired-end(QC.1.trimmed.fastqとQC.2.trimmed.fastq):0 transcript or contig (W5-2)
  - single-end (forward側のみ; QC.1.trimmed.fastq):1 transcript (W6-2)
  - single-end (reverse側のみ; QC.2.trimmed.fastq):423 transcripts (W6-4)

①

第5回原稿PDFのp195

今日の前半は、①Rパッケージ QuasRによる乳酸菌ゲノムへのマッピングの話。原因を特定しアセンブルやマッピングが改善されたことを確認するまでが第5回原稿内容

として、FastQC によるクオリティチェックを行えばよい［W1-2］。著者らは、FastQC 実行結果ファイルの項目（Overrepresented sequences）を眺めて、トリム前に見えていた既知のアダプターやプライマー配列が、トリム後に正しく見えなくなっていることを確認して安心している［W1-3］。

このデータに関して結論からいえば、forward 側の 107 bp のリードファイル（SRR616268sub_1.fastq.gz → QC.1.trimmed.fastq）のうち、100–107 塩基付近に乳酸菌に由来しないものがトリムしきれずに多く残っている。これは、アセンブルやマッピングがうまくできない、という実害を被ることでわかる。計算時間がかかるため、できるだけ QC 段階で問題解決するという方針もあろう。しかし、やってみてはじめてわかることもある。以降の内容は、著者らが実際に行ったことを問題解決に至る思考回路とともに述べる。大まかに述べると、Rockhopper2 [18] によるトランスクリプトームアセンブリ、QuasR [19] による乳酸菌ゲノムへのマッピング、そして QC 再実行である。

トランスクリプトームアセンブリ

ゲノムのアセンブリは、断片化されたゲノム配列由来リードをつなぎ合わせて、元のゲノム配列を再構築する作業である。この再構築に相当する英語がアセンブリ（assembly）であり、再構築を行うプログラムをアセンブラ（assembler）という。デノボ（*de novo*）という言葉が同時に用いられることが多いが、これは「最初から」と

か「一から」という意味であ
入力として（つまり他の情報を一切利用せずに）アセンブルする際には、*de novo* assembly という表現がなされる。トランスクリプトームアセンブリとは、アセンブル対象がゲノムではなく解析サンプル中で発現している全転写物（トランスクリプトーム）の場合を指す。RNA-seq データのみを入力として一からアセンブルする場合は、*de novo* transcriptome assembly などと呼ばれる。

このデータに関して結論からいえば、forward 側の 107 bp のリードファイル（SRR616268sub_1.fastq.gz → QC.1.trimmed.fastq）のうち、100–107 塩基付近に乳酸菌に由来しないものがトリムしきれずに多く残っている。これは、アセンブルやマッピングがうまくできない、という実害を被ることでわかる。計算時間がかかるため、できるだけ QC 段階で問題解決するという方針もあろう。しかし、やってみてはじめてわかることもある。以降の内容は、著者らが実際に行ったことを問題解決に至る思考回路とともに述べる。大まかに述べると、Rockhopper2 [18] によるトランスクリプトームアセンブリ、QuasR [19] による乳酸菌ゲノムへのマッピング、そして QC 再実行である。

①

Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

基本はコピペ

①乳酸菌NGS連載第5回のサイト。②W14-1からスタート。例えば次のスライドは③のコピペ実行結果のスクリーンショット。私は手打ちに忙殺されて全体像の理解が追い付かないという結果になるほうが無様だと思う派ですが…、考え方はヒトそれぞれ。どうしても手打ちしたければタブ補完を有効利用して気合いでついてきましょう

(Rで)塩基配列解析

~NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統

(last m

What's

- このウ
リーン
法(W

- 書籍 | 日本乳酸菌学会誌 | について (last modified 2016/05/12
- 書籍 | 日本乳酸菌学会誌 | 第1回イントロダクション (last modif
- 書籍 | 日本乳酸菌学会誌 | 第2回GUI環境からコマンドライン環境へ (last modified 2
- 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 modi
- 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
- 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 2
- 書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ (last modified 2016/04/23)
- 書籍 | 日
- イントロ |

①

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

日本乳酸菌学会誌の第5回分です。Linuxコマンドのリンク先は主に日経BP社様です。

- 第5回分PDF
- ウェブ資料PDF(2015.12.22版; 約14MB)

Linuxコマンド

- bzip2 (bzip2圧縮、および解凍)
- cd (ディレクトリを変更)
- echo (文字列を表示)
- export (変数を追加)
- file (ファイルタイプを判定)

マッピング (R ver. 3.2.0; QuasR ver. 1.8.4)

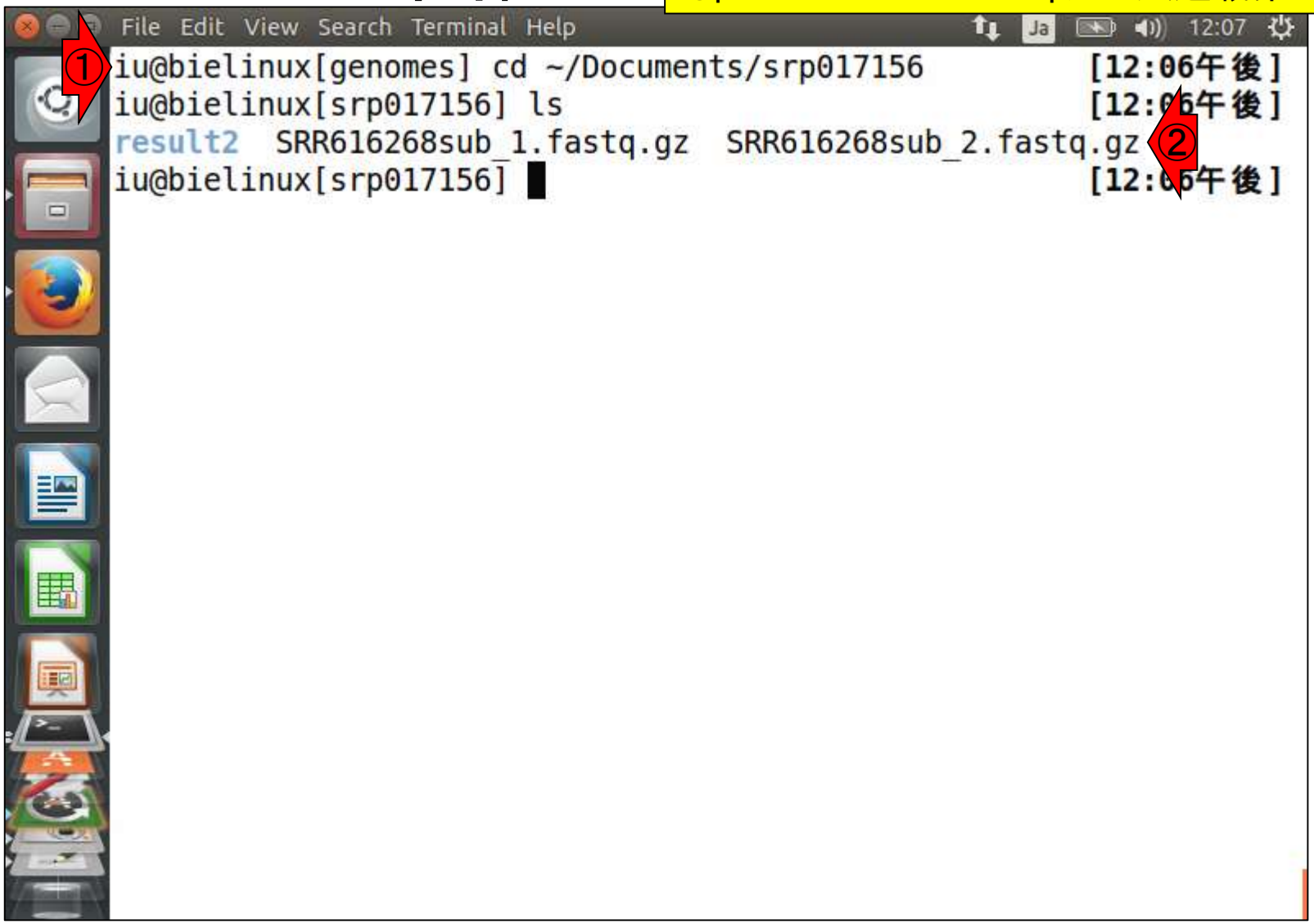
- 入力ファイルの準備と行数確認[W14-1] ②

③

```
cd ~/Documents/srp017156
ls

ls result2
gzip result2/*.fastq
ls result2

cp result2/QC.1.trimmed.fastq.gz .
cp result2/QC.2.trimmed.fastq.gz .
ls

gunzip -c SRR616268sub_1.fastq.gz | wc
gunzip -c SRR616268sub_2.fastq.gz | wc
gunzip -c QC.1.trimmed.fastq.gz | wc
gunzip -c QC.2.trimmed.fastq.gz | wc
```

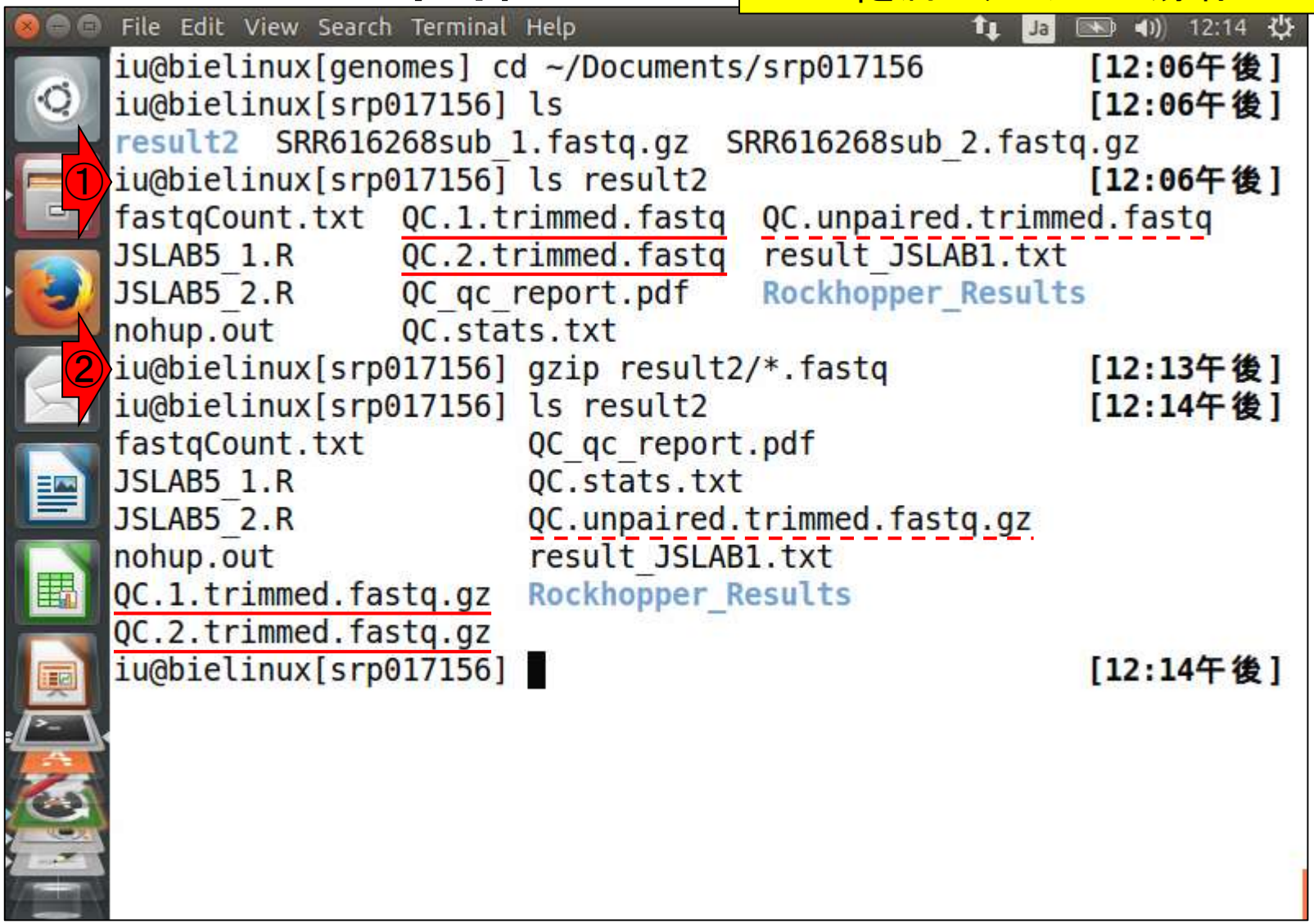
W14-1：準備

①作業ディレクトリは「~/Documents/srp017156」。②ここで見えている2つのgzファイルは、の100万リード(400万行)からなるpaired-end RNA-seqデータ(連載第3回W25あたりで作成)



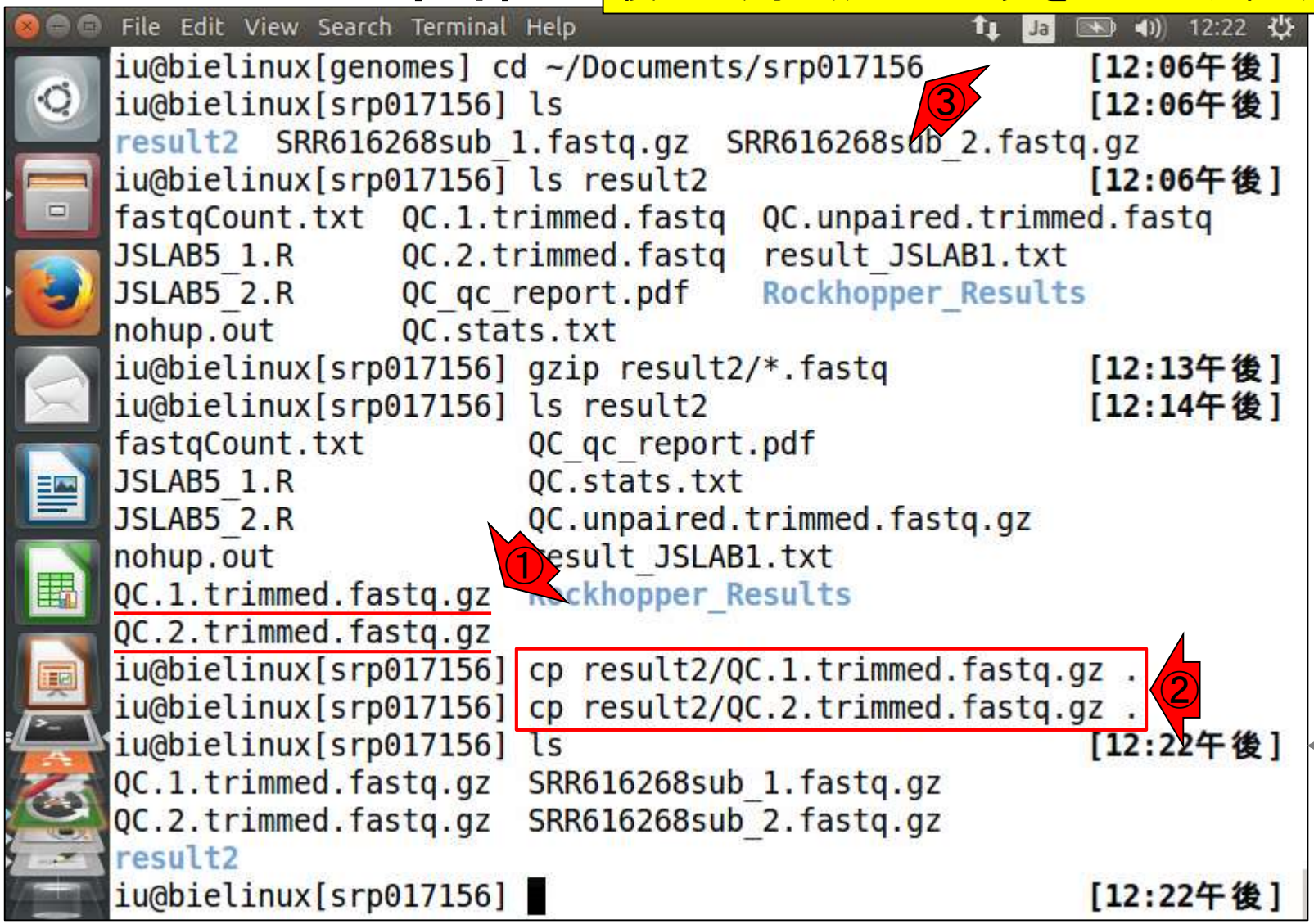
W14-1：準備

①「~/Documents/srp017156/result2」ディレクトリ上にある*.fastqファイルがFaQCs実行結果。確認。②この中の*.fastqを満たすファイル(赤線の3つ)をgzip圧縮。数分



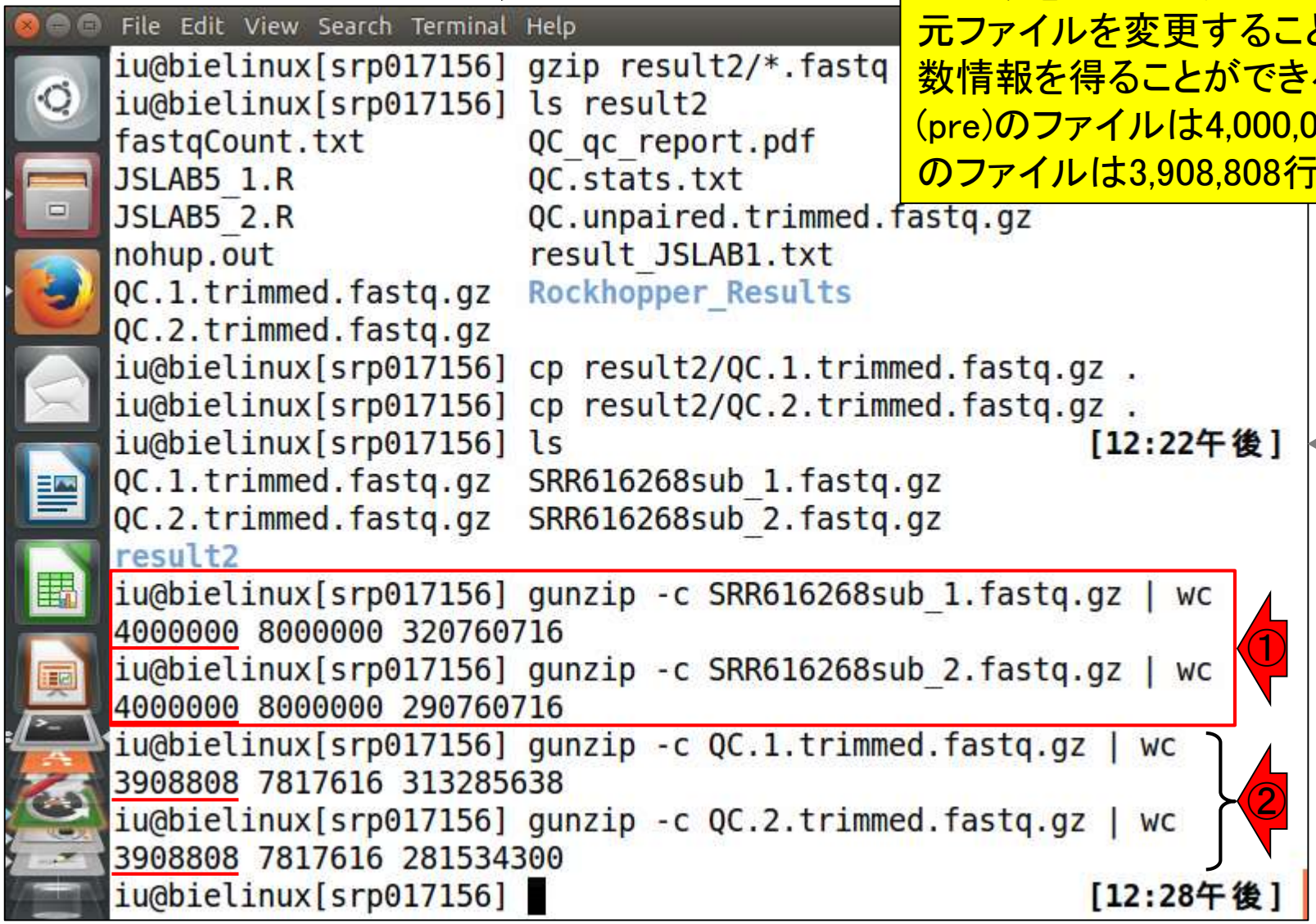
W14-1：準備

①マッピングしたいのは赤下線の2つのファイルのみ。②これらをカレントディレクトリにコピー。③つまりココ。②のcpコマンドの最後のピリオド(.)はコピー先をカレントディレクトリにするという意味



W14-1：行数確認

-cオプションをつけて元ファイルを残したままでgzip圧縮ファイルを解凍。パイプ(|)でそのまま行数をカウントするwcコマンドに流すことで、元ファイルを変更することなくgzファイルの行数情報を得ることができる。①FaQCs実行前(pre)のファイルは4,000,000行、②実行後(post)のファイルは3,908,808行であることがわかる



W14-2：リストファイル

①リストファイルの作成（正確にはダウンロード）と確認。QuasRは複数サンプルのマッピングが可能。ここでは、FaQCs実行前(pre)と実行後(post)のpaired-endファイルをリストとして与えてマッピングを実行するつもり

W14-2：wget失敗時は

①ダウンロードサイトは、アグリバイオのサーバなので、まずはwgetをトライしてみてよい。うまくいかないときは②「~/Desktop/backup」からのコピーで各自対応してください

• リストファイル[W14-2]

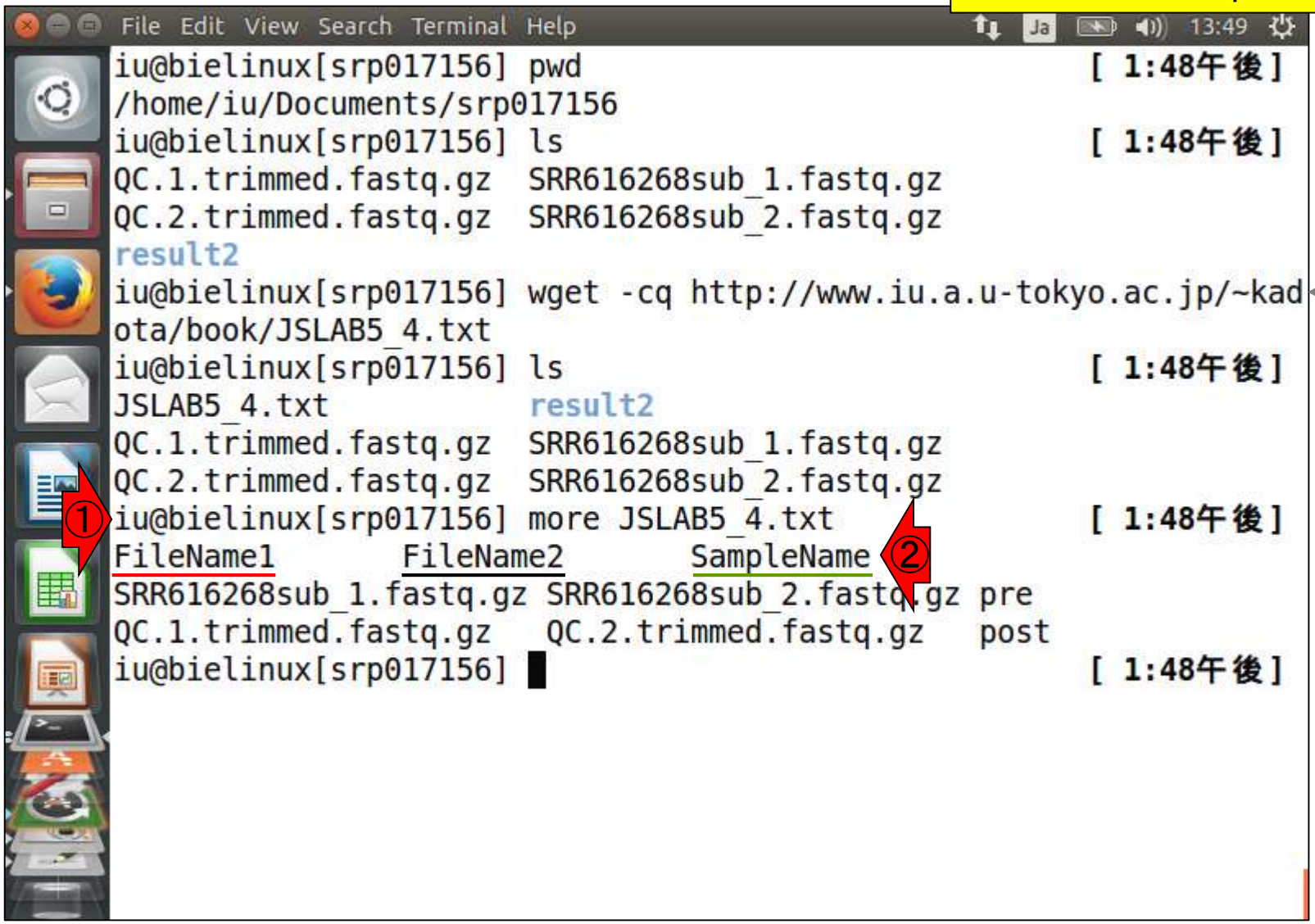
JSLAB5_4.txtの作成(正確にはダウンロード)。wgetで失敗したヒトは「cp ~/Desktop/backup/JSLAB5_4.txt .」で各自対処してください。

```
cd ~/Documents/srp017156

pwd
ls
wget -cq http://www.iu.a.u-tokyo.ac.jp/~kadota/book/JSLAB5_4.txt
#cp ~/Desktop/backup/JSLAB5_4.txt .
ls

more JSLAB5_4.txt
```

②

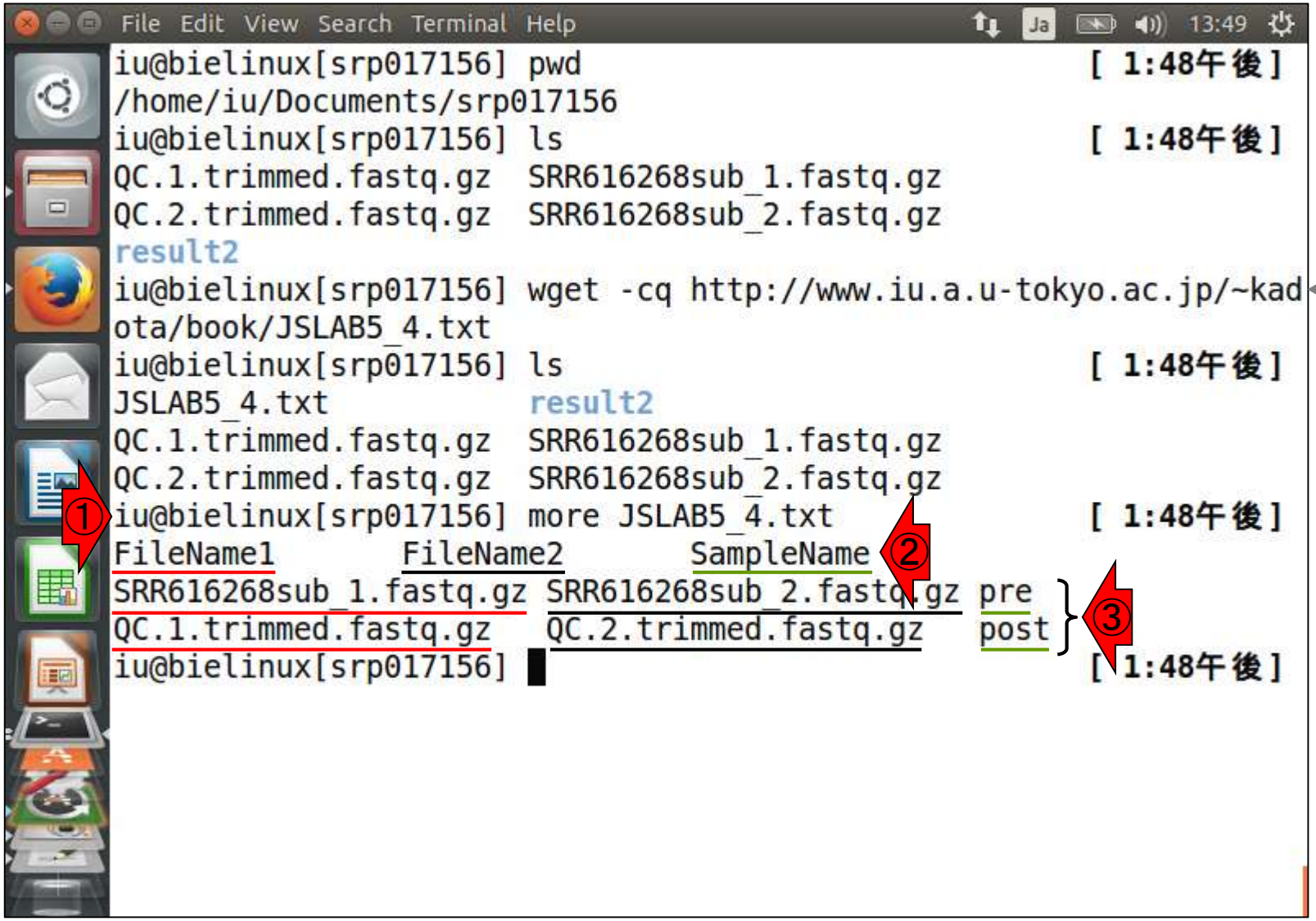
W14-2：リストファイル

①リストファイルの中身を確認。paired-endの場合は、1行目の部分は、②「FileName1 FileName2　SampleName」と書く(固定)



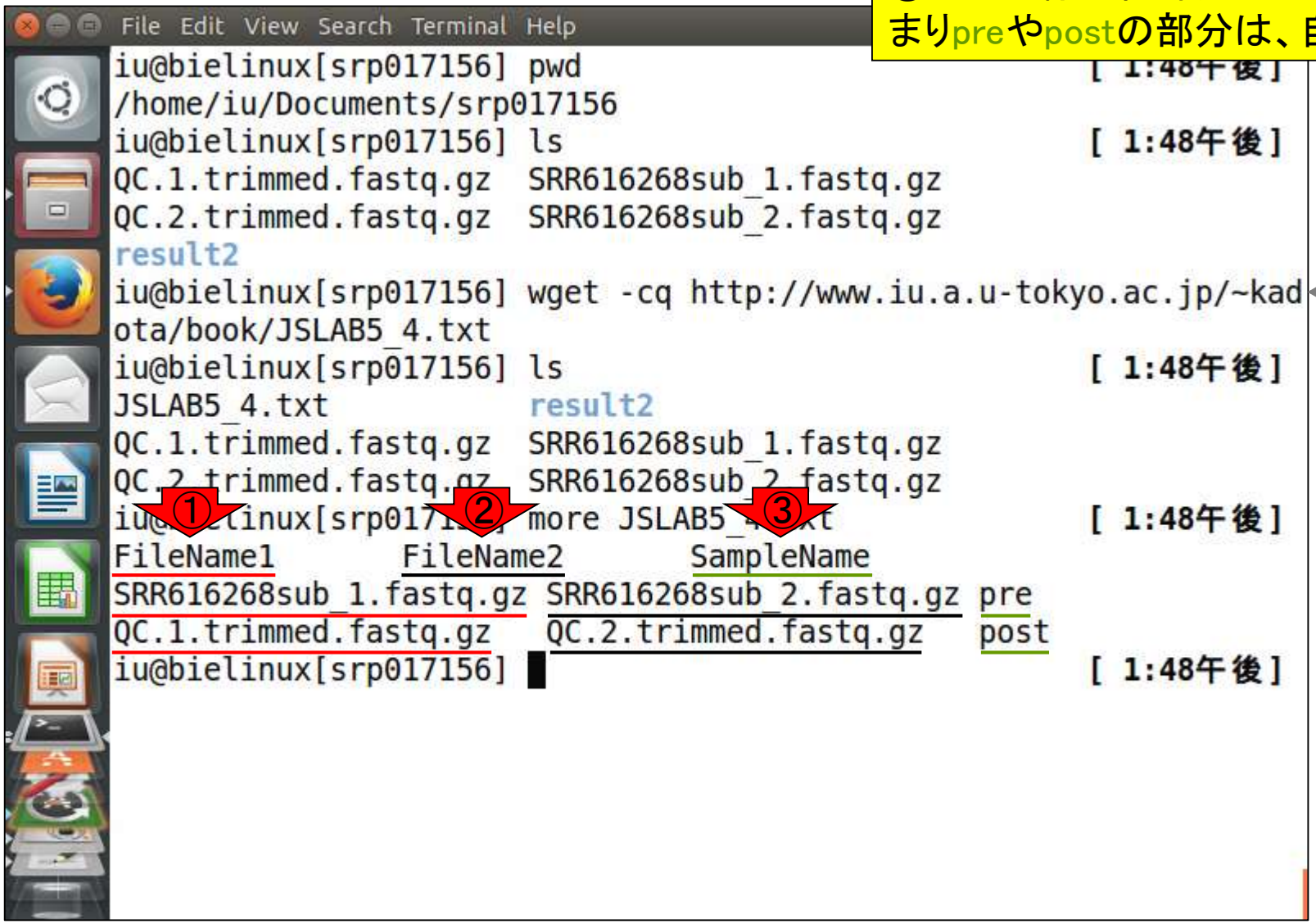
W14-2：リストファイル

③2行目以降にマッピングしたい入力ファイル名を書く



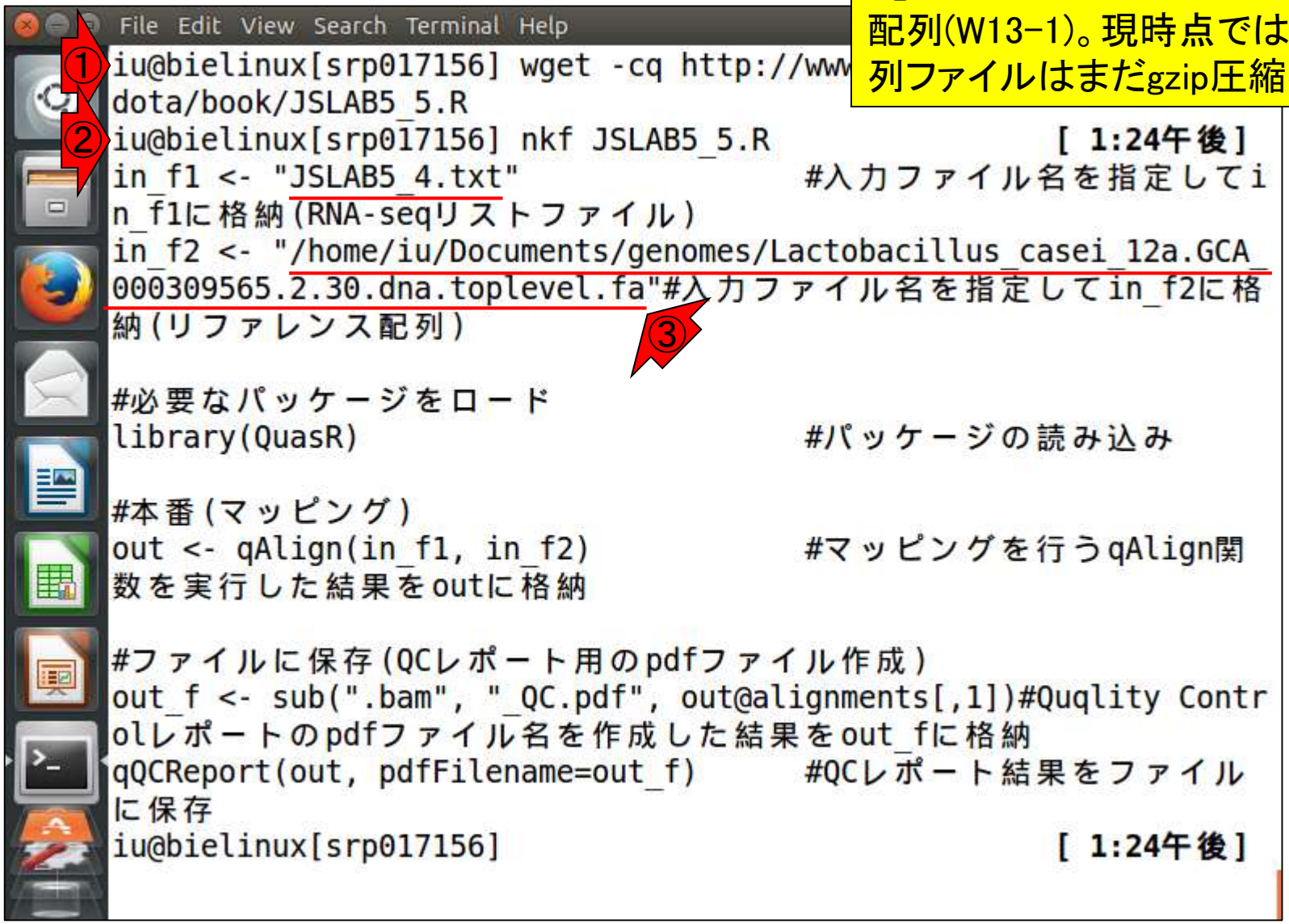
W14-2：リストファイル

①1列目（赤下線部分）はforward側のファイル、
②2列目（黒下線部分）はreverse側のファイル、
③3列目(緑下線部分)は任意のサンプル名。つまりpreやpostの部分は、自由に変えてよい



W14-3：Rスクリプト

①Rスクリプトファイル(JSLAB5_5.R)のダウンロードと、②中身の表示。nkfは文字化け回避用。in_f1がリストファイル(W14-2)。in_f2がリファレンス配列(W13-1)。現時点では、③のリファレンス配列ファイルはまだgzip圧縮状態。解凍は後で行う



W14-4：カラー表示

カラー表示。実際のコマンドはごくわずか。①qAlign関数部分がマッピング本番。②qQCReport関数は、PDFレポート作成用。これはまだコピペしない!

- カラー表示[W14-4]

Rスクリプトファイル JSLAB5_5.Rの中身を表示。moreでみたものと基本的に同じです。in_f1には、マップしたいFASTQファイルのリストをQuasRの入力形式に従って作成したファイルの名前(JSLAB5_4.txt)、in_f2にはマップされる側のリファレンス配列を指定します。ここでは作業ディレクトリ上にないRelease 30の乳酸菌ゲノムファイル(解凍したファイル)を絶対パスで指定しています。

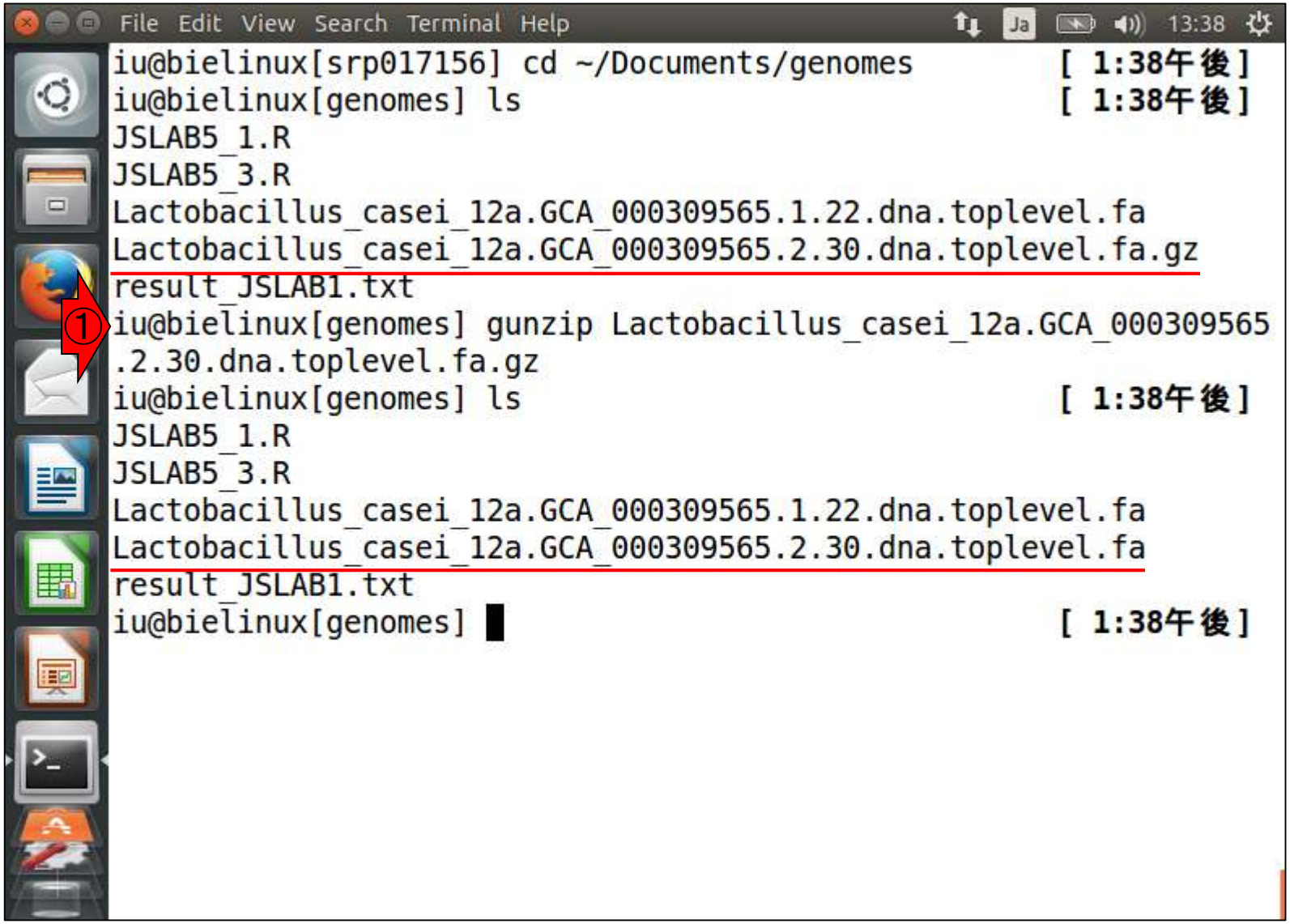
```
in_f1 <- "JSLAB5_4.txt"                          #入力ファイル名を指定してin_f1に格納(RNA-seqリ
in_f2 <- "/home/iu/Documents/genomes/Lactobacillus_casei_12a.GCA_000309565.2.30.dna

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

#本番(マッピング)
out <- qAlign(in_f1, in_f2)                      #マッピングを行うqAlign関数を実行した結果をoutに

#ファイルに保存(QCレポート用のpdfファイル作成)
out_f <- sub(".bam", "_QC.pdf", out@alignments[,1])#Quqlity Controlレポートのpdfファ
qQCReport(out, pdfFilename=out_f)                #QCレポート結果をファイルに保存
```

①

②

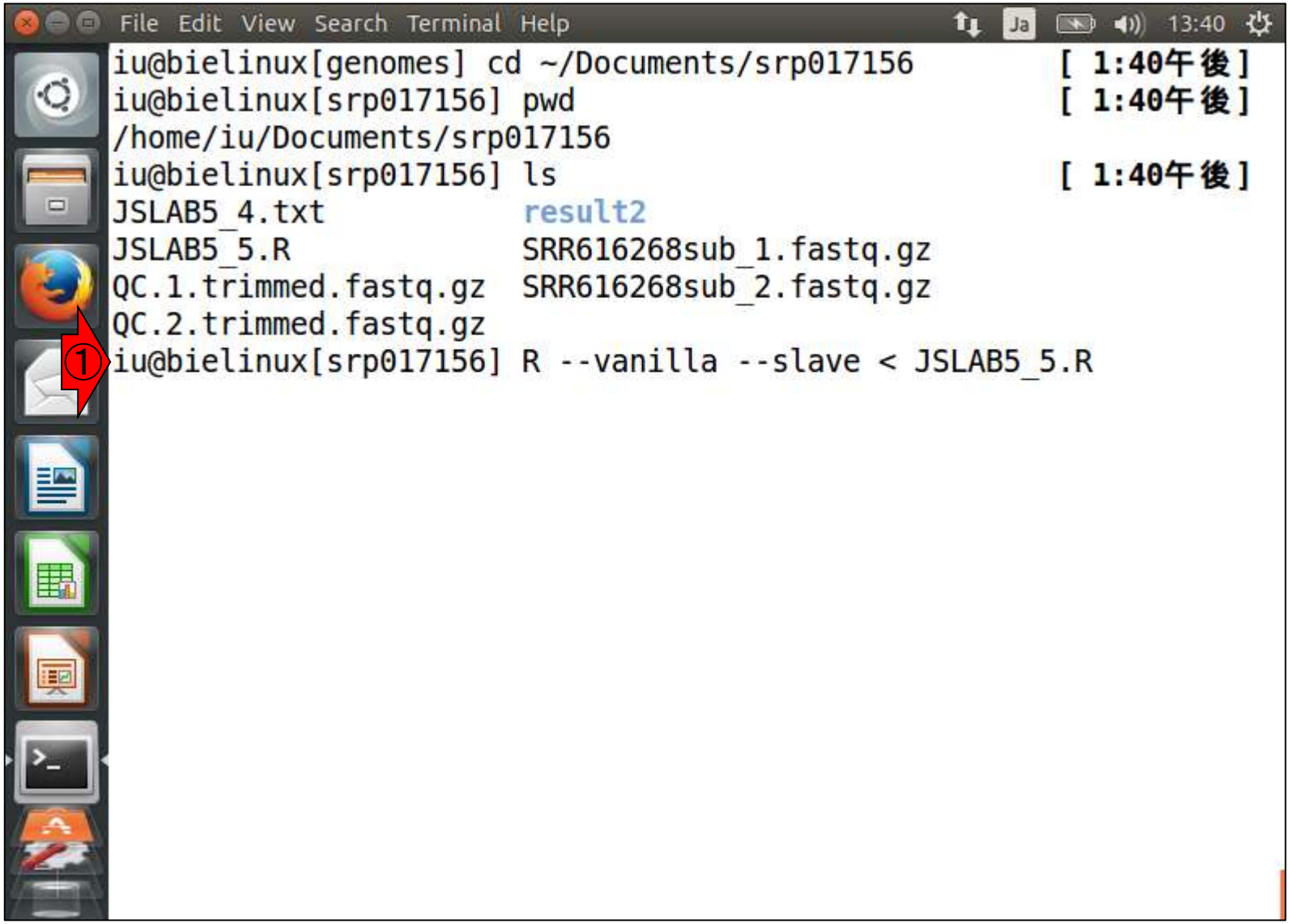
W14-5：解凍

①gzip圧縮されたリファレンスゲノム配列ファイルを解凍



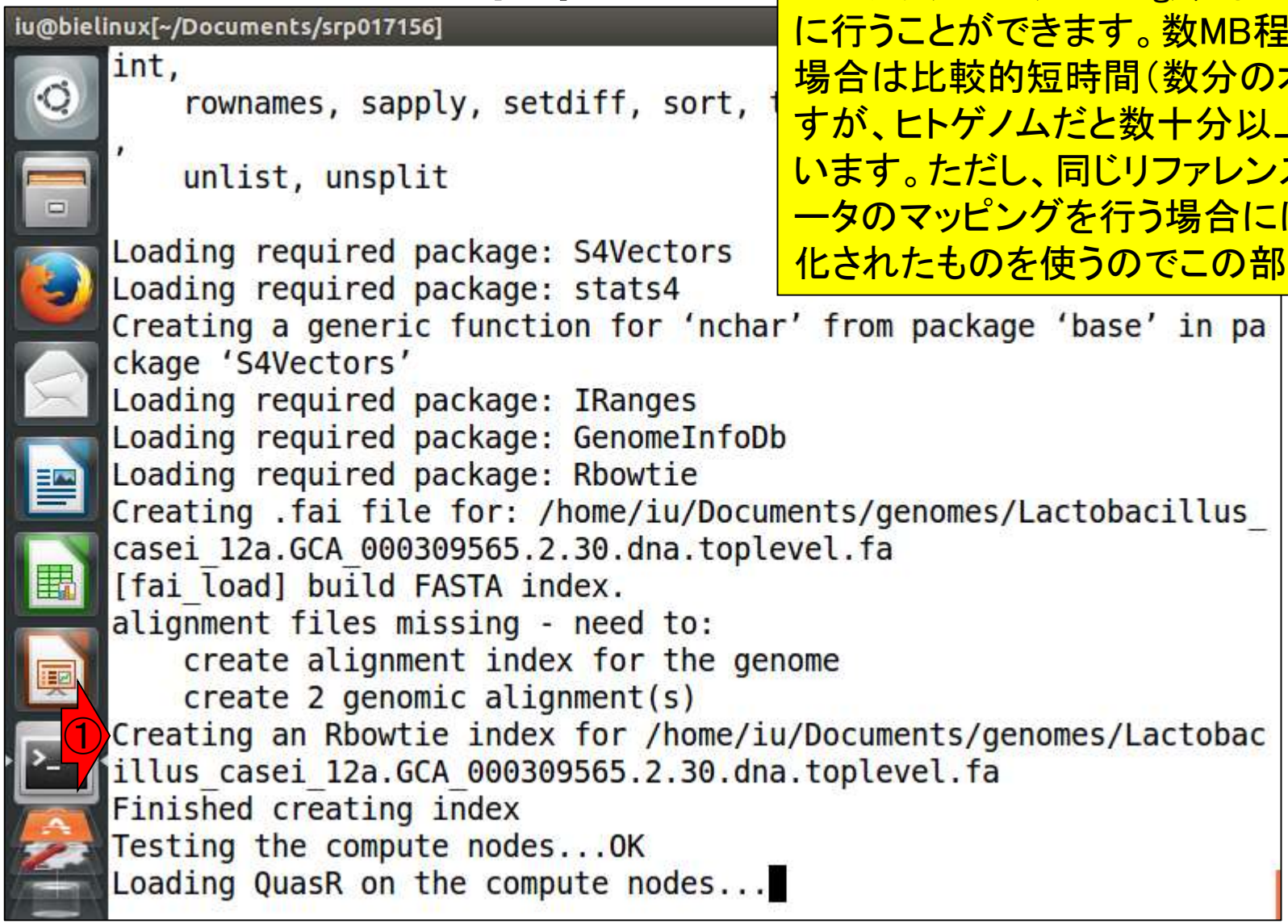
W14-5：マッピング本番

①QuasRでのマッピング用のRスクリプトファイルJSLAB5_5.R（W14-3）を実行。約15分



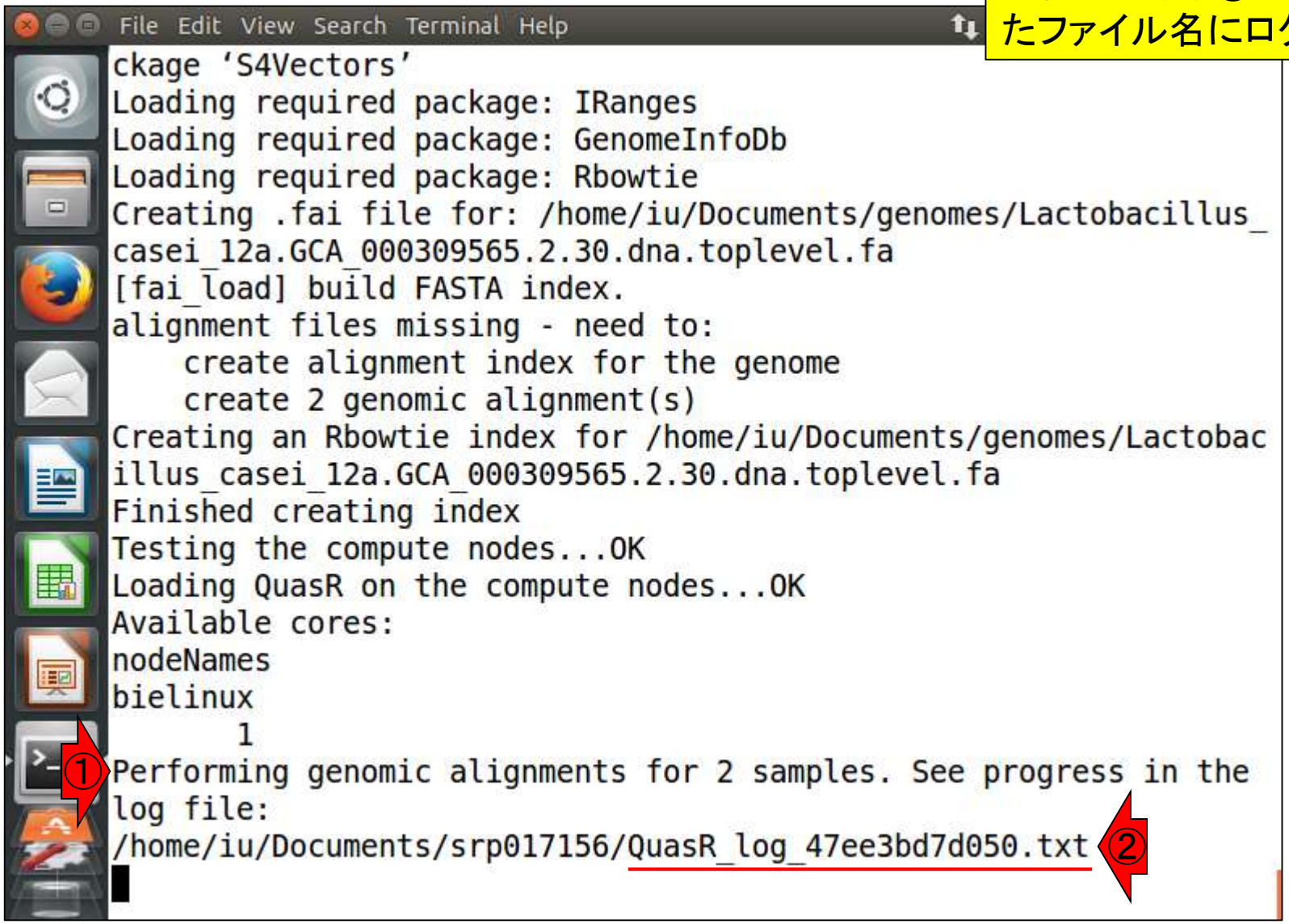
W14-5：途中経過1

リターンキーを押して数秒後の状態。①まず最初にやっているのは、リファレンス配列のインデックス化。インデックス化(indexing)することでマッピングを高速に行うことができます。数MB程度の乳酸菌ゲノムの場合は比較的短時間（数分のオーダー）で終わりますが、ヒトゲノムだと数十分以上はかかるのではと思います。ただし、同じリファレンス配列を使って別のデータのマッピングを行う場合には、既にインデックス化されたものを使うのでこの部分はスキップできます



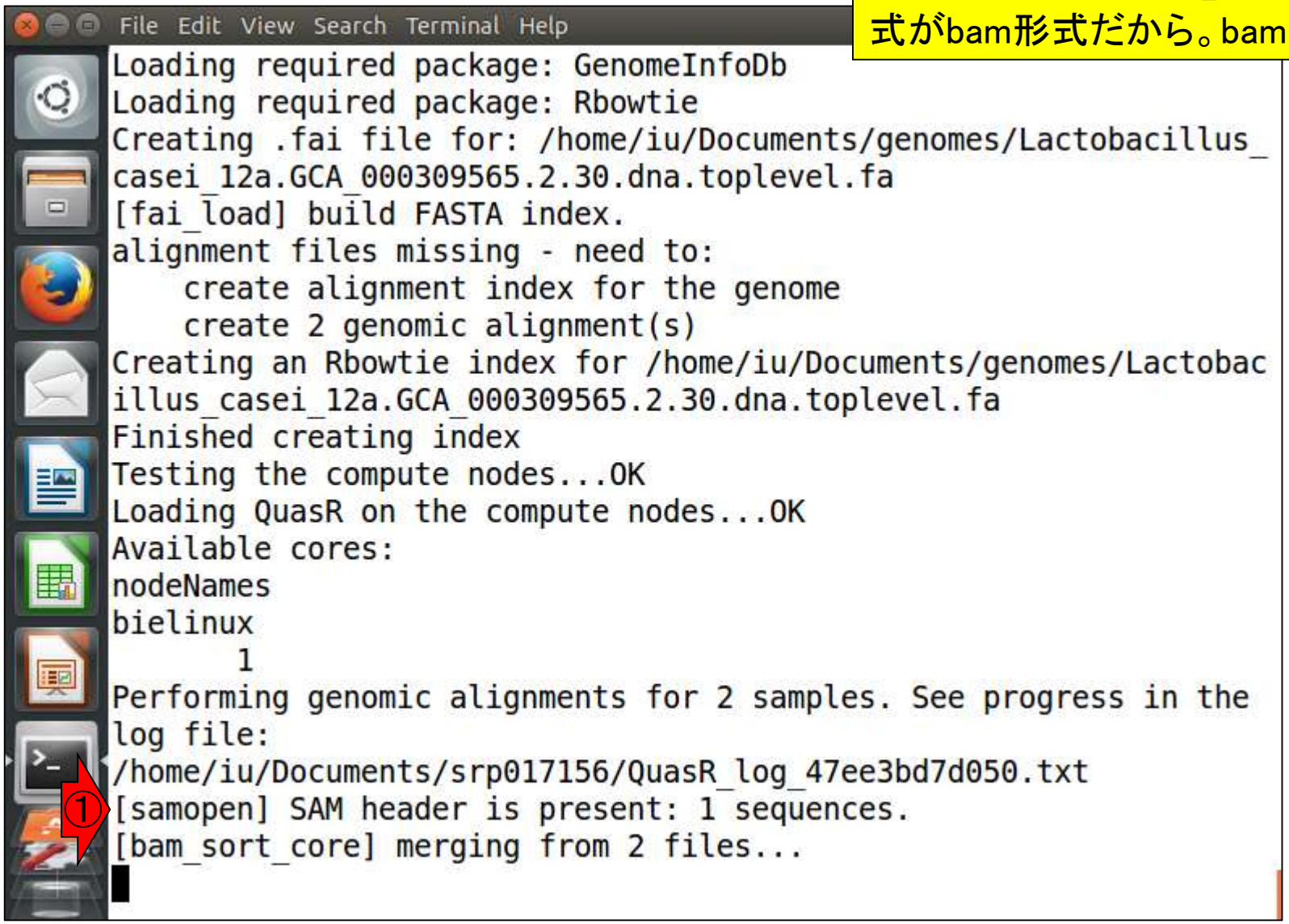
W14-5：途中経過2

①マッピングがスタート。この種のプログラムは実行ログファイルを作成する場合が多いです。②QuasRも絶対パスで示したファイル名にログを書き込んでいます



W14-5：途中経過3

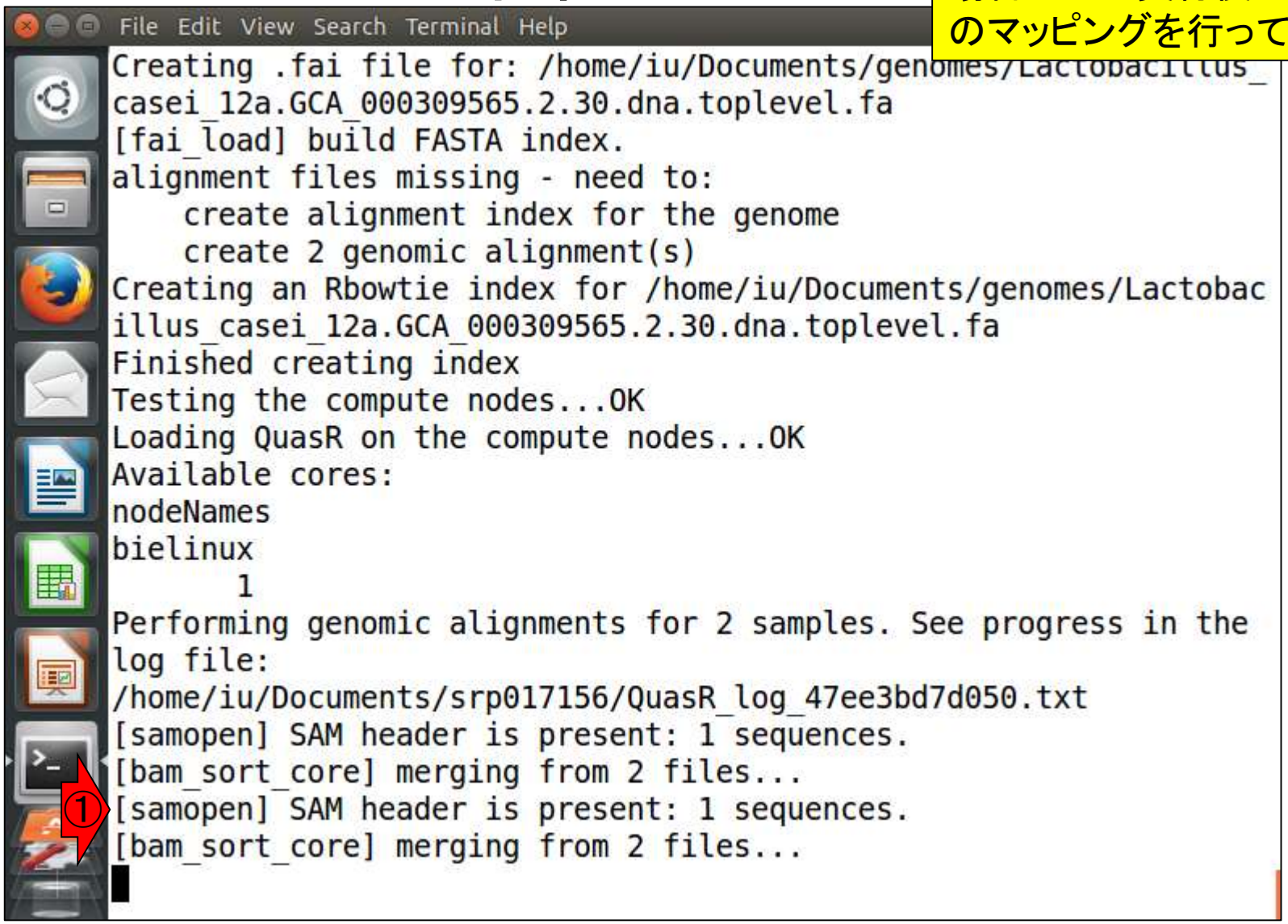
①samやbamと書かれているが、これは多くのマッピングプログラム(QuasRのデフォルトは内部的にBowtieプログラムを利用)の結果ファイルの形式がbam形式だから。bamはsamのバイナリ版



Bowtie：Langmead et al., *Genome Biol.*, **10**: R25, 2009

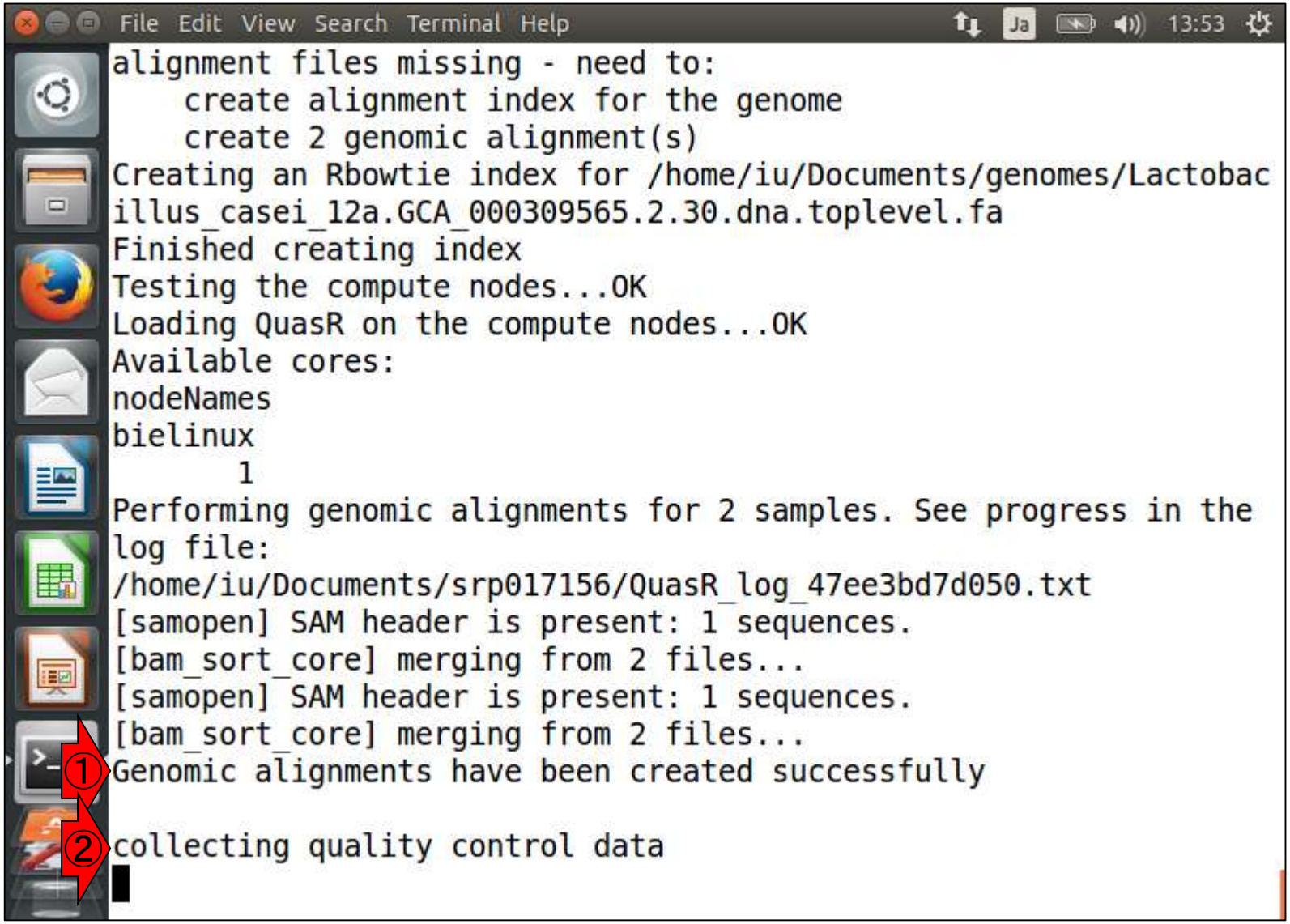
W14-5：途中経過4

①2回目のsamやbamの記述。おそらく2つめのサンプル(リストファイルの3行目。この場合FaQCs実行後のファイルQC.*.fastq.gz)のマッピングを行っているのだろう



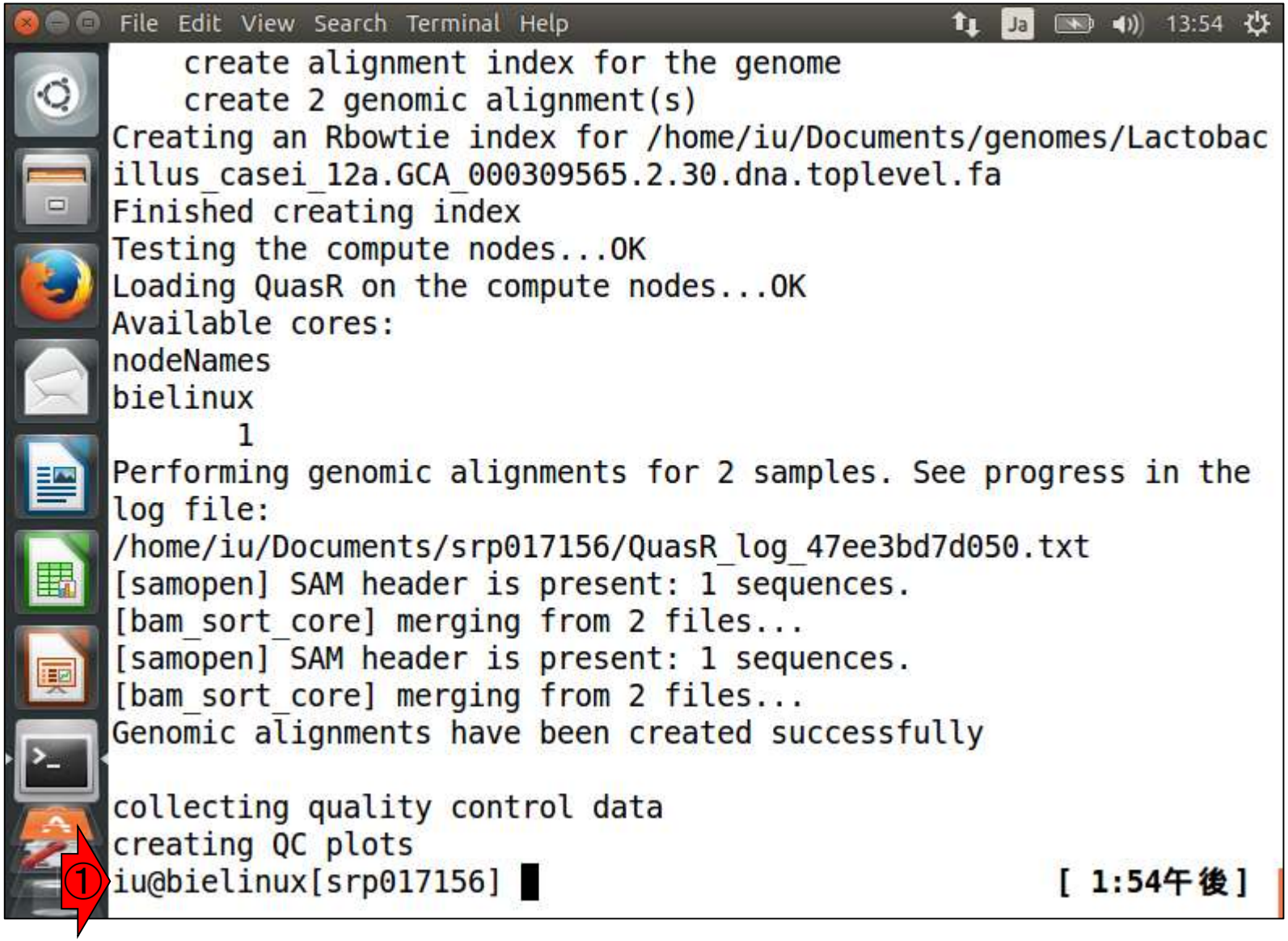
W14-5：途中経過5

①マッピングは無事に終了したようだ。
②QC情報を得ようとしているのだろう



①エラーを吐くことなく正常終了したようだ

W14-5：無事終了

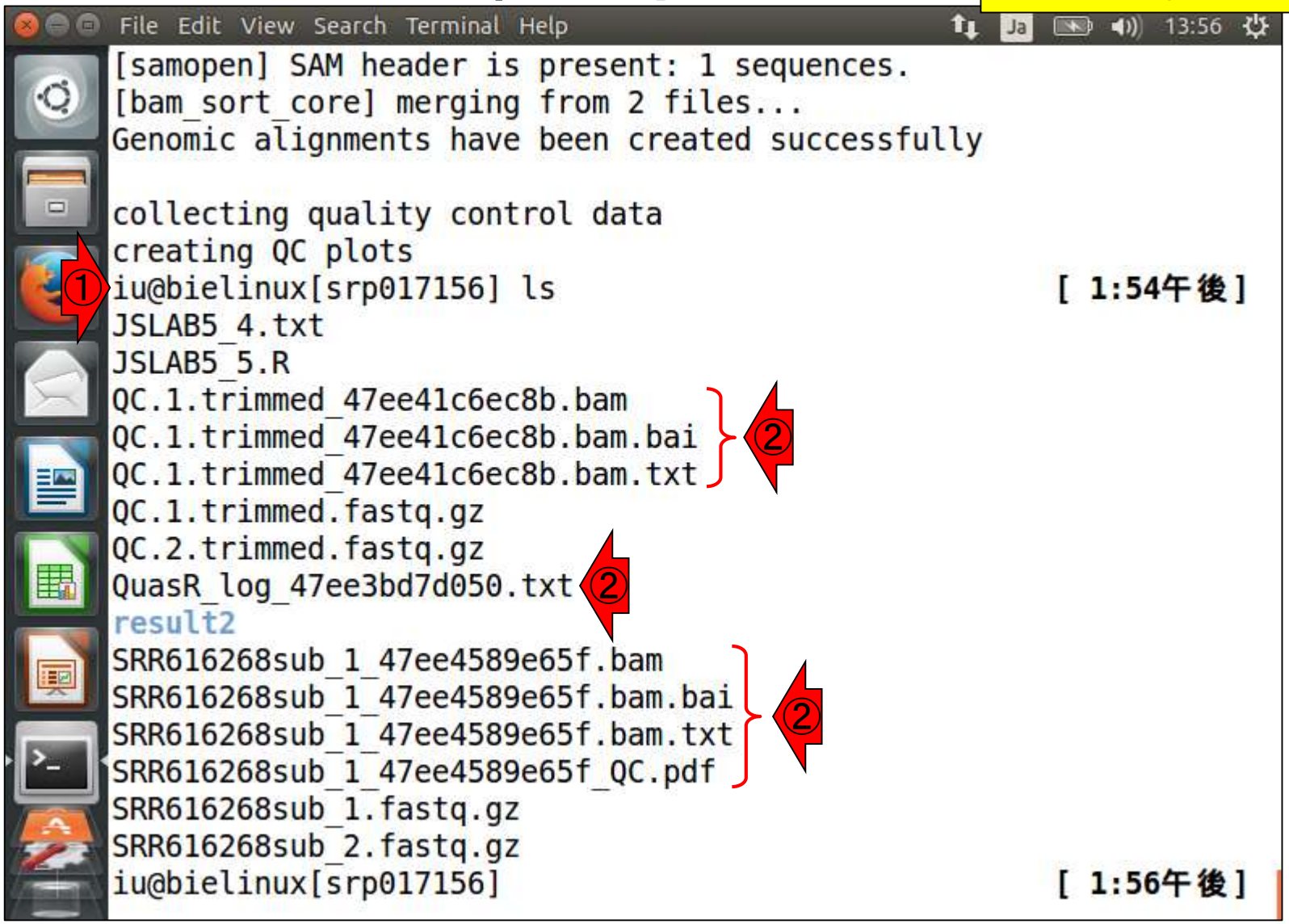


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

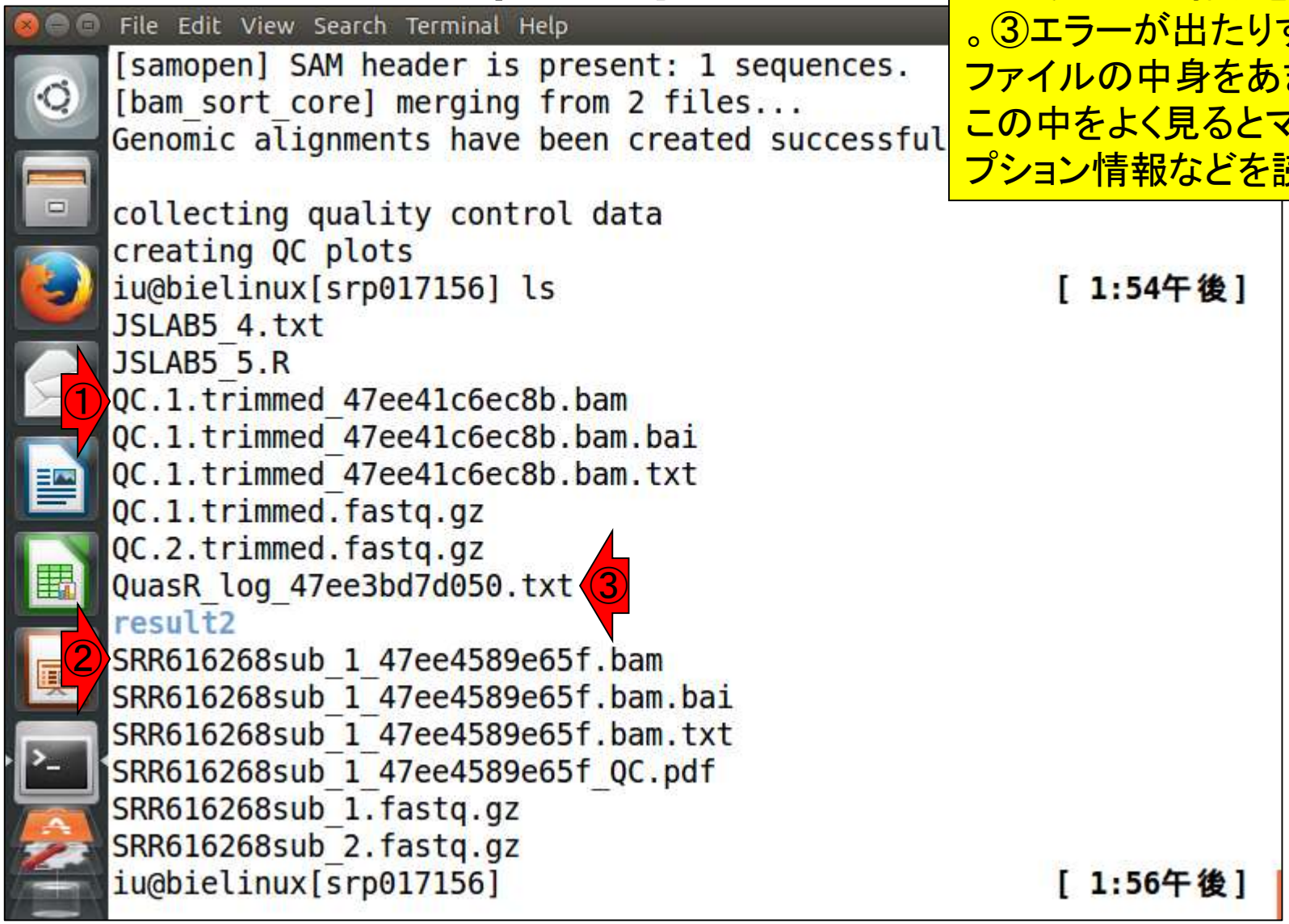
W15-1：結果の解説

①lsした結果。②多数のファイル（計8ファイル）が生成されていることがわかる。マッピング前(W14-5)と比べてみるとよい



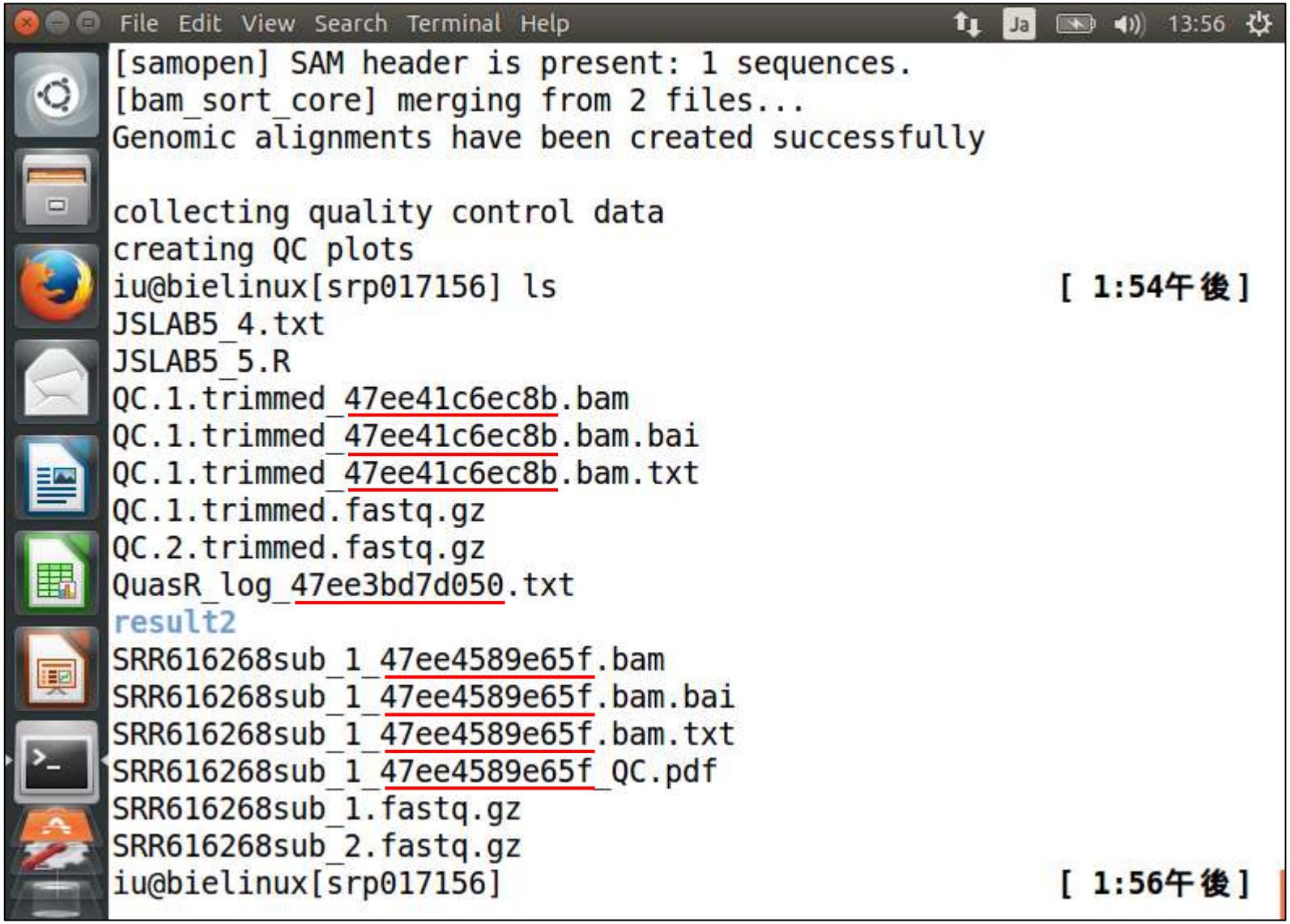
W15-1：結果の解説

マッピング結果ファイルのメインは①と②で示した.bam。この形式のファイルを入力としてその後の解析を行うプログラムは多い。③エラーが出たりすることがなければlogファイルの中身をあまり見ることはないが、この中をよく見るとマッピング時に用いたオプション情報などを読み取ることができる



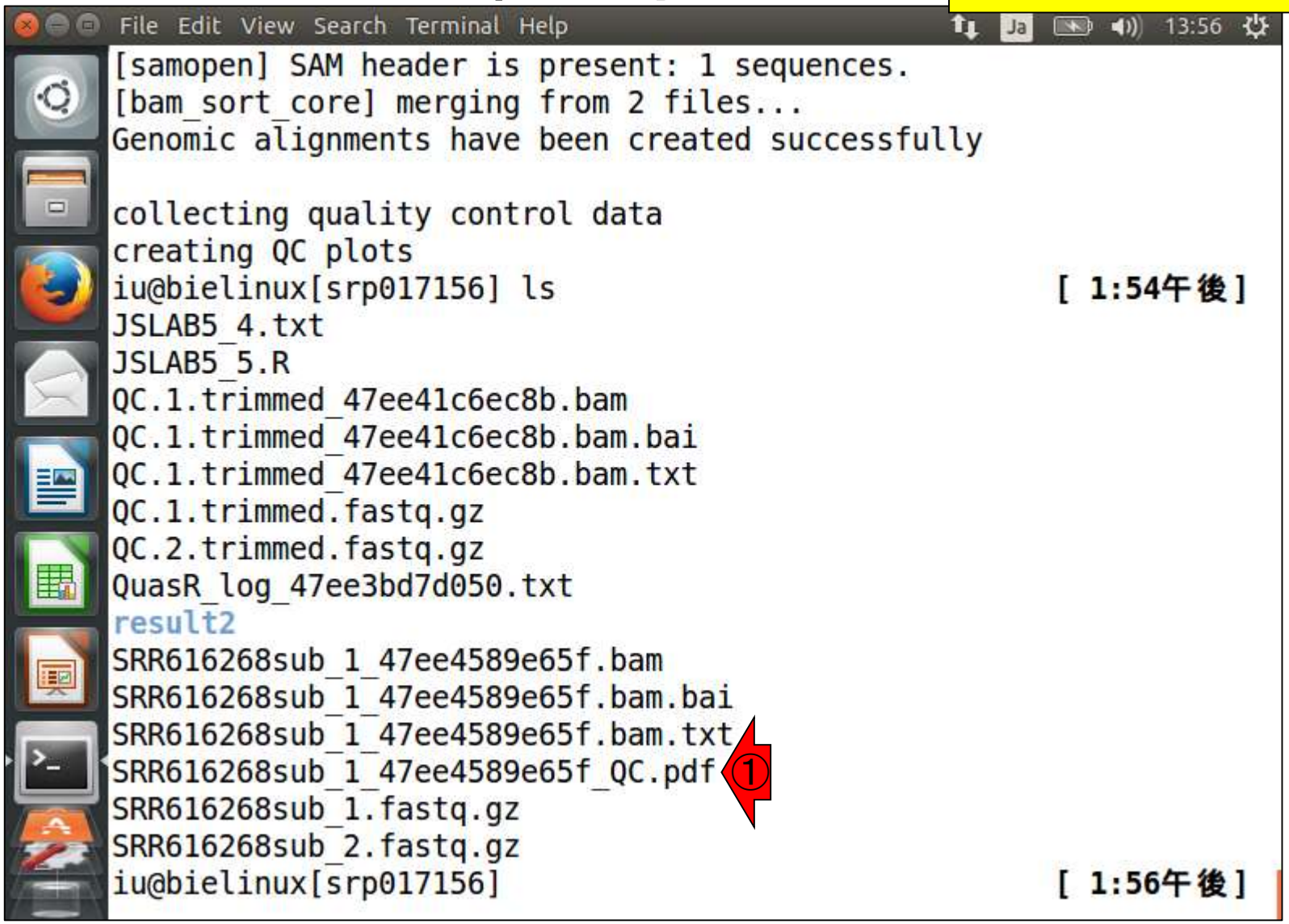
W15-1：結果の解説

赤下線部分の文字列はランダムに発生させているので、ヒトによって異なる



W15-1：結果の解説

①このPDFファイル中には、入力ファイル(paired-end RNA-seqリード)のQC情報や、どれだけマップされたかなどの結果がある



W15-1：結果の解説

①のファイルは、②の部分を実行したから生成された。わざわざ生成させたのは、Rockhopper2でアセンブルがうまくいかなかった理由が、このQCレポートファイルを眺めることでわかるから

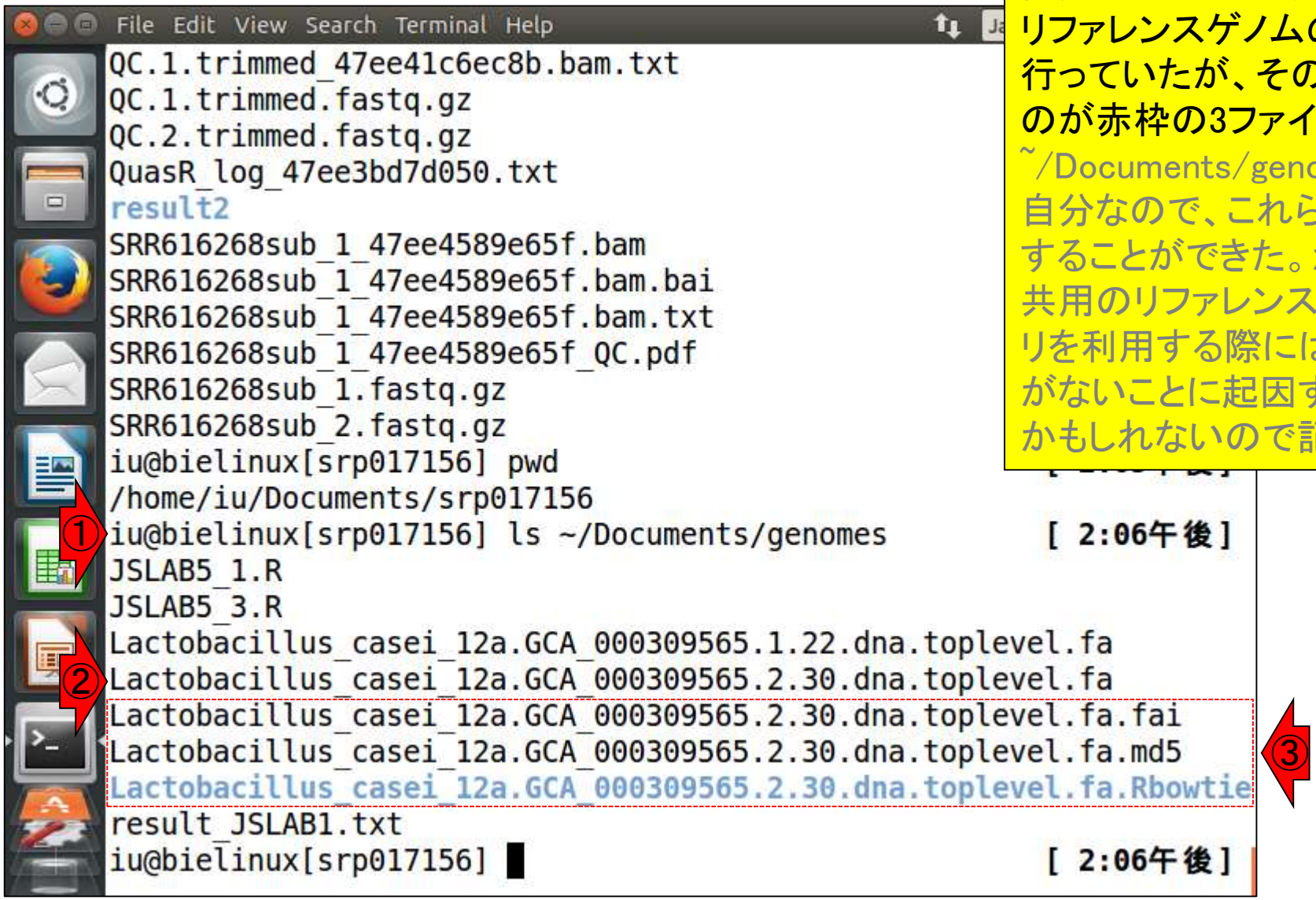
File Edit View Search Terminal Help

```
[samopen] SAM header is present: 1 sequences.
[bam_sort_core] merging from 2 files...
Genomic alignments have been created successfully

collecting quality
creating QC plots
iu@bielinux[srp017
JSLAB5_4.txt
JSLAB5_5.R
QC.1.trimmed_47ee4
QC.1.trimmed_47ee4
QC.1.trimmed_47ee4
QC.1.trimmed.fastq
QC.2.trimmed.fastq
QuasR_log_47ee3bd7
result2
SRR616268sub_1_47ee4589e65f.bam
SRR616268sub_1_47ee4589e65f.bam.bai
SRR616268sub_1_47ee4589e65f.bam.txt
SRR616268sub_1_47ee4589e65f_QC.pdf
SRR616268sub_1.fastq.gz
SRR616268sub_2.fastq.gz
iu@bielinux[srp017156]                                    [ 1:56午後 ]
```

```
in_f1 <- "JSLAB5_4.txt"                    #入力ファイル名を指定してin_f1に格納(RNA-seqリ
in_f2 <- "/home/iu/Documents/genomes/Lactobacillus_casei_12a.GCA_000309565.2.30.dna

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

#本番(マッピング)
out <- qAlign(in_f1, in_f2)                #マッピングを行うqAlign関数を実行した結果をoutに

#ファイルに保存(QCレポート用のpdfファイル作成)
out_f <- sub(".bam", "_QC.pdf", out@alignments[,1])#Q ity Controlレポートのpdfファ
qQCReport(out, pdfFilename=out_f)          #QCレポート結果 ファイルに保存
```

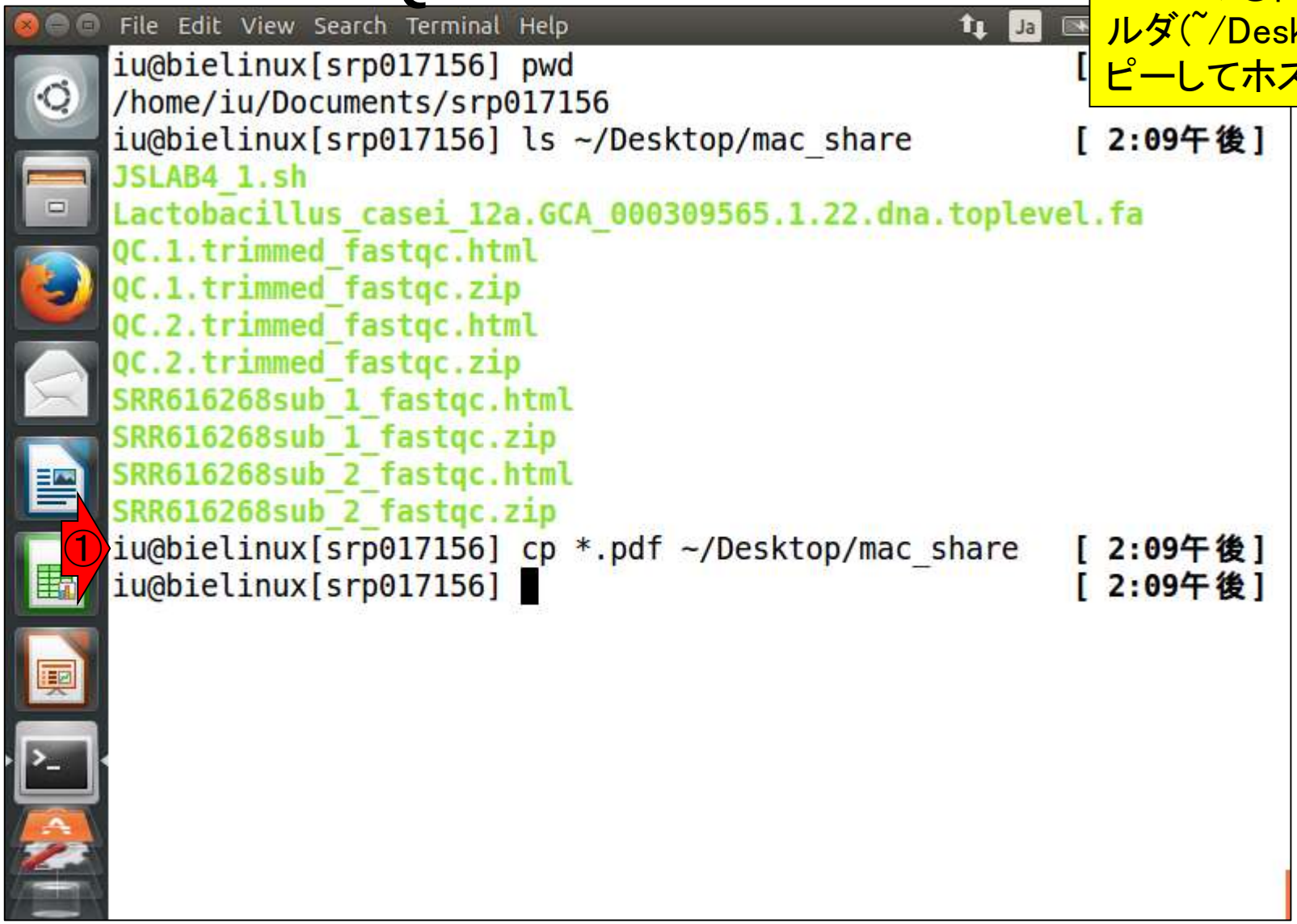
W15-2：リファレンスのほう

①リファレンスゲノムファイルがあるディレクトリをls。②リファレンスとして指定したファイル。③W14-5の最初でリファレンスゲノムのインデックス化を行っていたが、そのときに作成されたのが赤枠の3ファイル。「~/Documents/genomes」の所有者が自分なので、これらのファイルを作成することができた。が、スパコンなどで共用のリファレンスゲノムのディレクトリを利用する際には、書き込み権限がないことに起因するエラーが起こるかもしれないので記憶に留めておこう



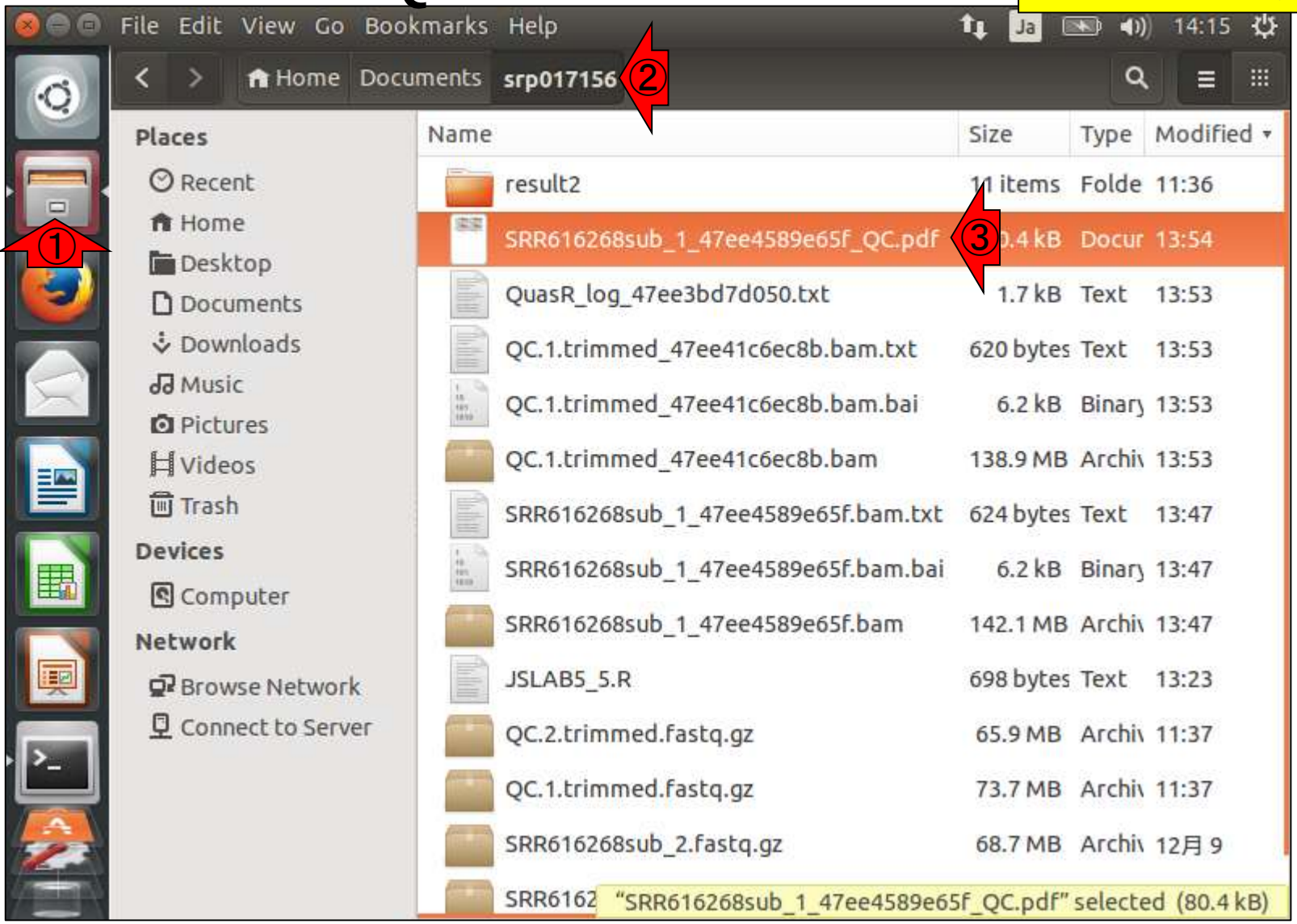
W15-3：QCレポート

QuasRでマッピングしたのは、QCレポートを眺めるのが主目的。ここでは、①pdfファイルを共有フォルダ(~/Desktop/mac_share)にコピーしてホストOS上で眺めるが…



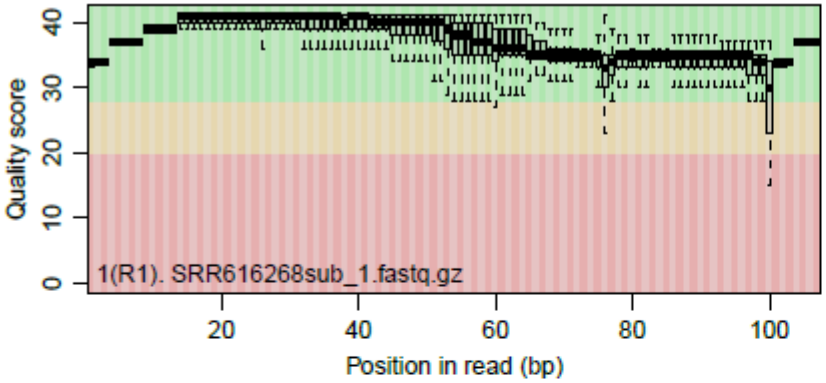
W15-3：QCレポート

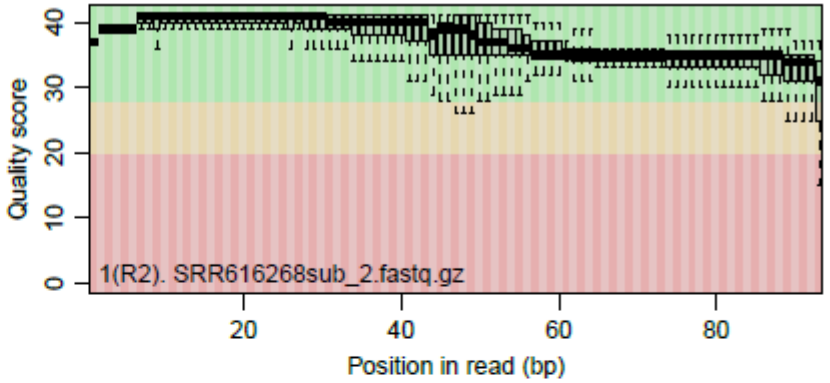
①引出しアイコンをクリックしていってゲストOS上で眺めてもよい。反応が遅いのでイラッとするが気長に待つべし



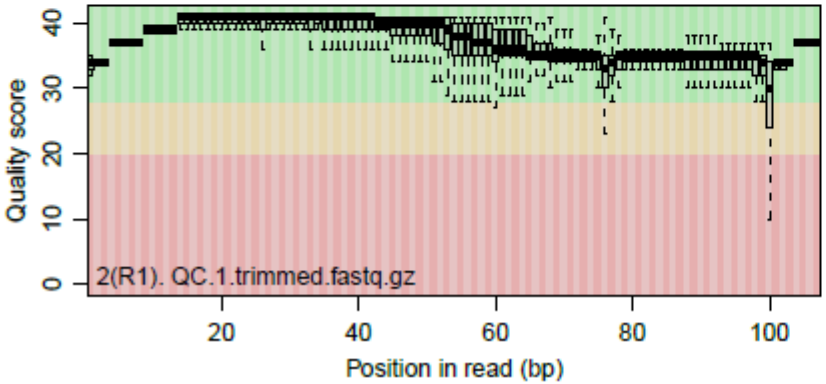
W15-4：PDF解説

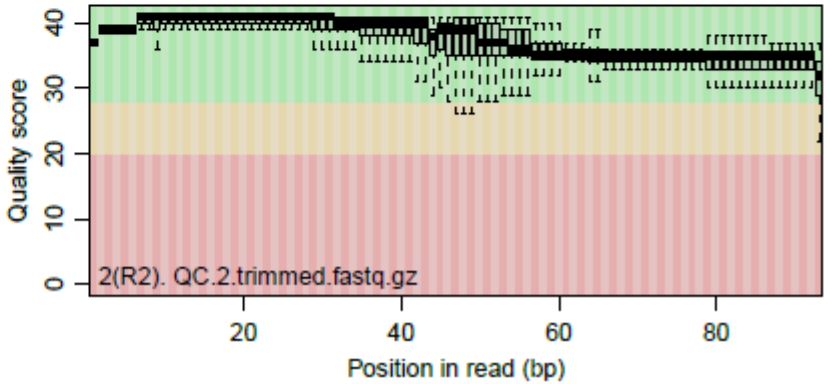
PDF1枚目。入力ファイルのQuality score分布。FastQC Report中の項目「Per base sequence quality」と同じ。①上段がFaQCs実行前(pre)、②下段が実行後(post)。③左がforward側、④右がreverse側。ここでの目的はFaQCs実行前後の比較ではなく、マップされなかったリードの割合や、数少ないマップされたリードの調査なので、劇的な違いはないが気にしない





① FaQCs 実行前 (pre)





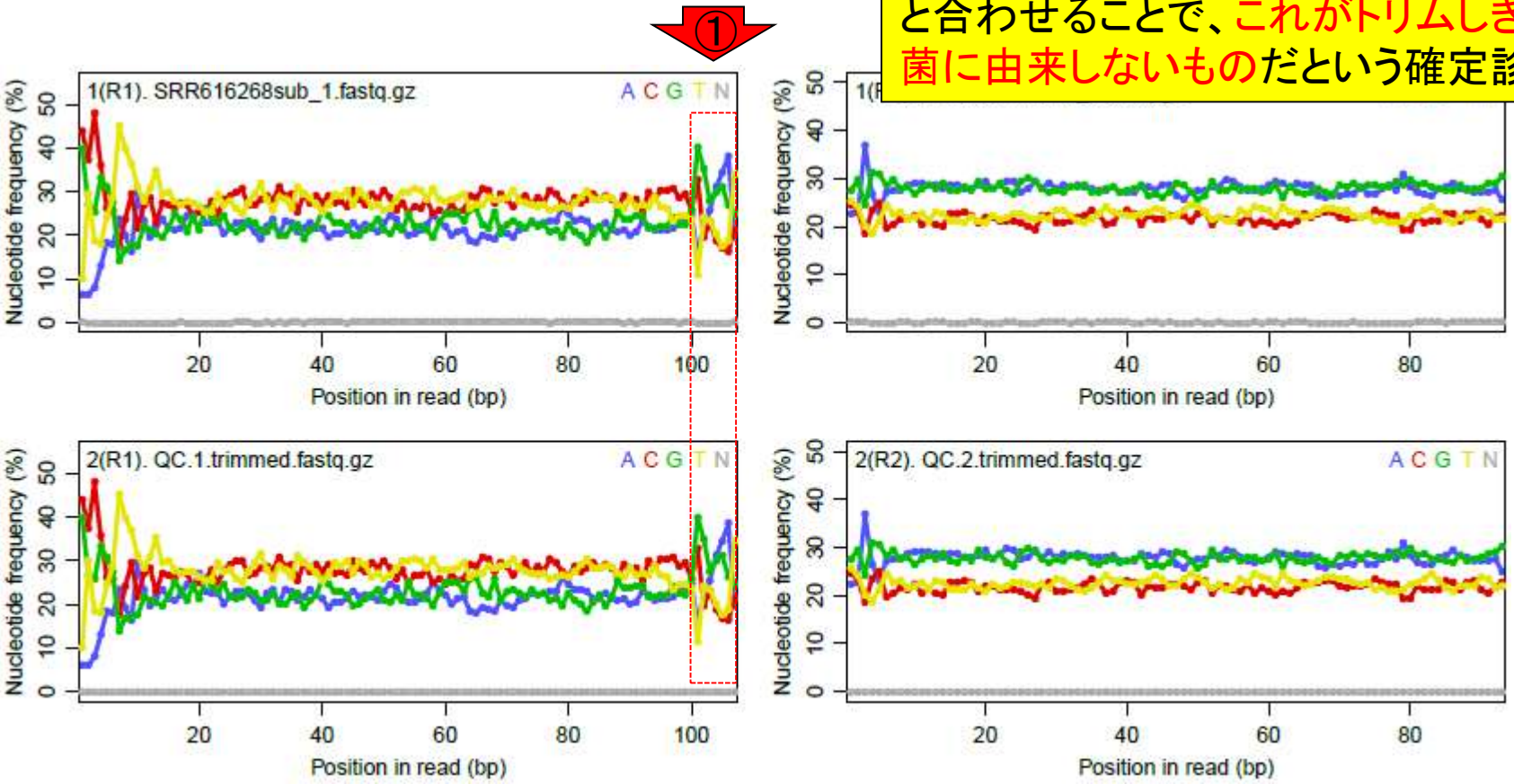
② FaQCs 実行後 (post)

③ forward側

④ reverse側

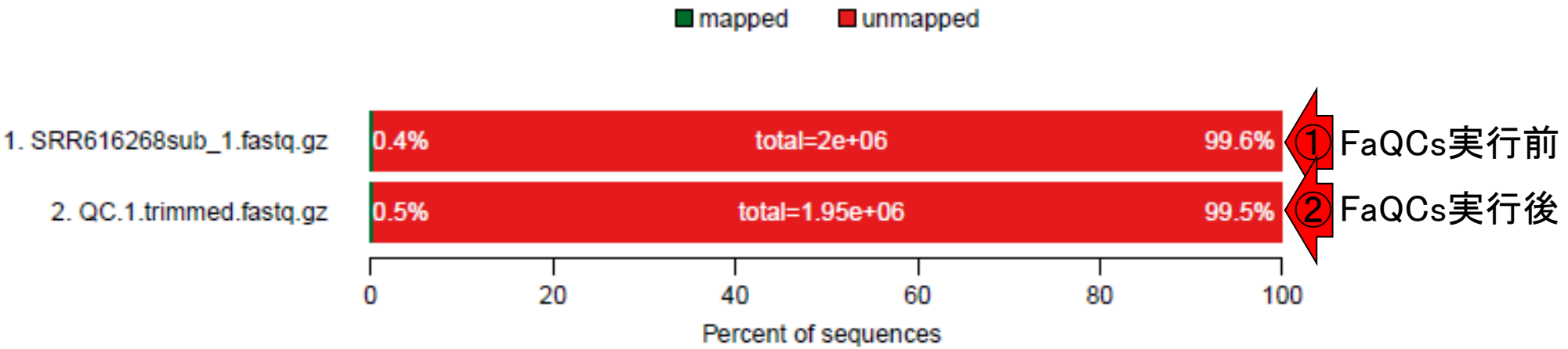
W15-5：PDF解説

PDF2枚目。ポジションごとの塩基の出現確率。FastQC Report中の項目「Per base sequence content」と同じ(但し色は異なる)。①forward側の100-107bp付近(赤枠部分)の分布が不自然。このような結果は、FastQCをデフォルトオプションで実行すると得られない。この結果と後のほうのPDFレポートと合わせることで、これがトリムしきれていない乳酸菌に由来しないものだという確定診断が下される



W15-6：PDF解説

PDF4枚目。全リード(forward, reverse合わせて約200万リード)のうち、マップされたリードの割合は①FaQCs実行前(pre)が0.4%、②実行後(post)が0.5%。ほとんどマップされなかったことを意味する

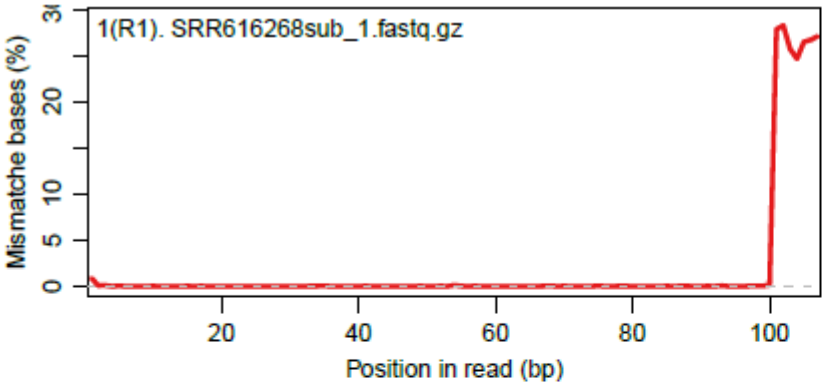


W15-7：PDF解説

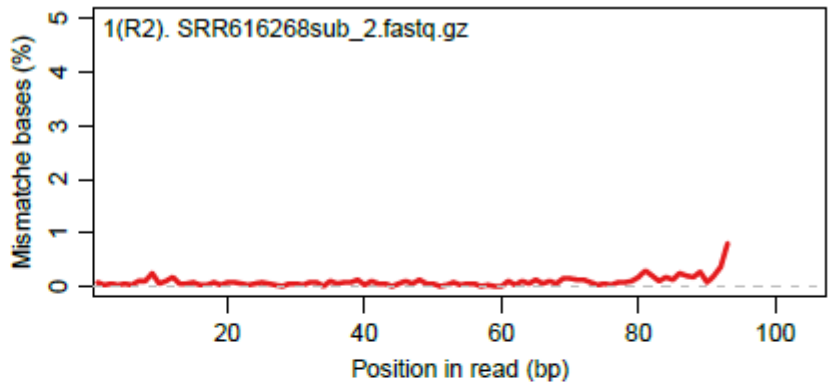
PDF6枚目。マニュアルを読んでもよくわからなかったが、おそらく縦軸がMismatche basesとなっているので、ミスマッチを許容してマップされたリードの中でどこにミスマッチがあったかを表示しているものと思われる。多少解釈が間違っていたとしても、①このプロット分布を見れば、「forward側の100-107 bp付近が犯人」という結論は不変

①

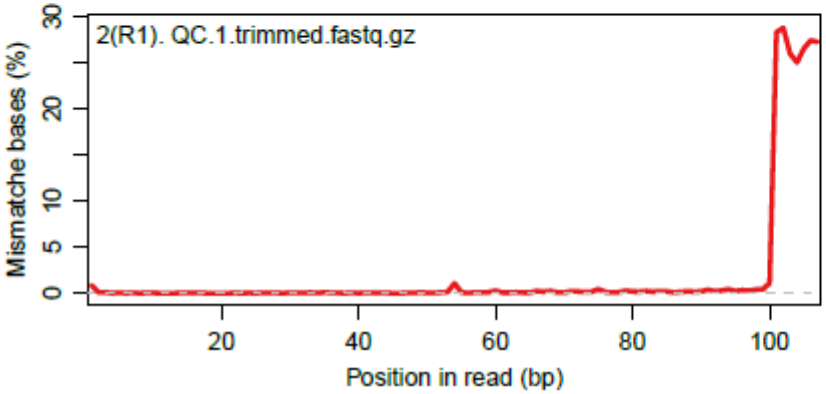
forward側

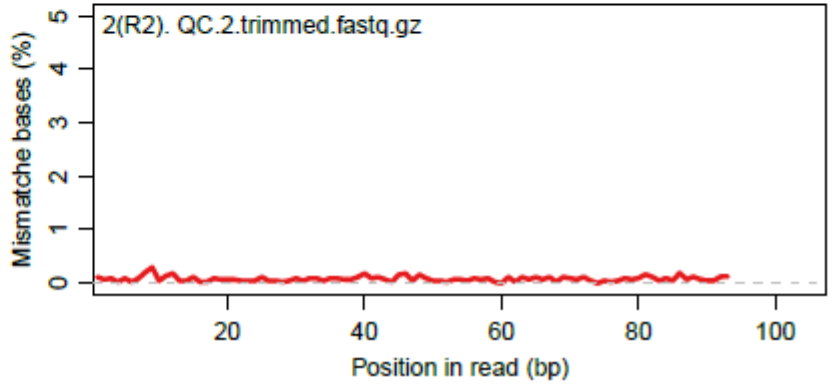


reverse側



FaQCs
実行前
（pre）



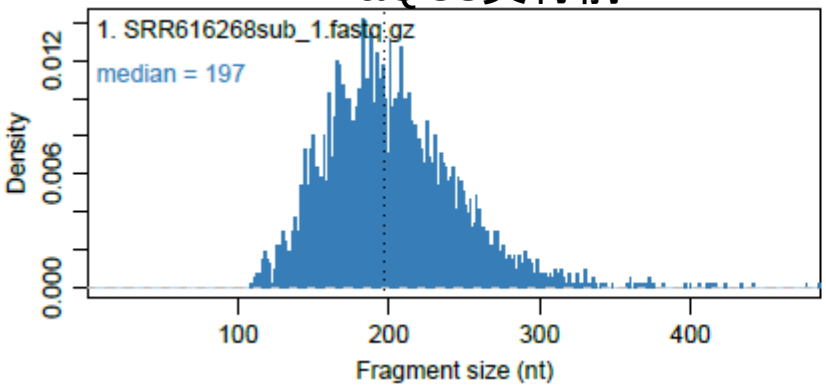


FaQCs
実行後
（post）

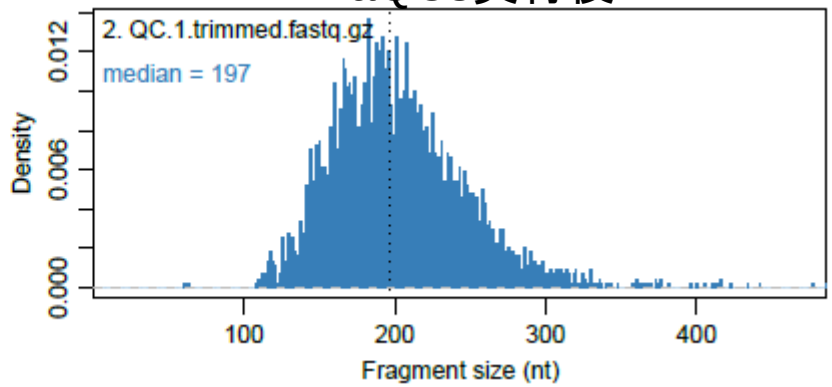
W15-8：PDF解説

PDF8枚目。入力はpaired-endなので、おそらくforward側とreverse側両方でマップされたリードのみを取り扱っている。ゲノム配列上でのforwardとreverse間の距離分布をプロットしているものと思われる。このあたりは、よほど変な分布になっていない限り、私は気にも留めない

FaQCs実行前



FaQCs実行後



W15-9：参考

参考

今回のRスクリプトファイル（JSLAB5_5.R）は、①のコードをテンプレートとして作成した。尚、第5回では述べないが、②カウント情報取得まで一気に行いたい場合のテンプレートなど、QuasRを用いたものは多数ある。スライドを見るだけ

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオイン
(last modified 2015/09/12, since 2011)

What's new?

- このウェブページはインストール||についての推奨手順 (Windows2015.04.04版とMacintosh2015.04.03版)に従ってフリーソフトRと必要なパッケージをインストール済みであるという前提で記述しています。初心者の方は基本的な利用法(Windows2015.04.03版と Macintosh2015.04.03版)で自習してください。本ウェブページを体系的にまとめた書籍もあります。(2015/04/03)

- sample1.fastaのようなコンティグ数が1つしかない場合に、rowSums(x)の計算時にエラーが出ることがわかったので、該当箇所をapply(as.matrix(x), 1, sum)のような感じに変更しました。(2015/09/12) NEW
- NGSハンズオン講習会2015のアメリエフ様分(服部先生と山口先生)の講義資料を差し替えました。(2015/09/03) NEW
- NGSハンズオン講習会のフォローア

- マッピング | (ESTレベルの長さの)contig (last modified 2014/06/24)
- マッピング | 基礎 (last modified 2013/06/19)
- マッピング | single-end | ゲノム | basic aligner(基礎) | QuasR(Gaidatzis_2015) (last modified 2014/06/21)
- マッピング | single-end | ゲノム | basic aligner(応用) | QuasR(Gaidatzis_2015) (last modified 2015/06/28)
- マッピング | single-end | ゲノム | splice-aware aligner | QuasR(Gaidatzis_2015) (last modified 2014/06/21)
- マッピング | paired-end | ゲノム | basic aligner(応用) | QuasR(Gaidatzis_2015) ① modified 2015/07/02)
- マップ後 | |について (last modified 2013/06/19)
- マップ後 | 出力ファイル形式について (last modified 2013/11/05)
- マップ後 | 出力ファイルの読み込み | BAM形式 (last modified 2014/06/21)
- マップ後 | 出力ファイルの読み込み | Bowtie形式 (last modified 2013/06/18)
- マップ後 | 出力ファイルの読み込み | SOAP形式 (last modified 2013/06/19)
- マップ後 | 出力ファイルの読み込み | htSeqTools(Planet_2012) (last modified 2013/06/19)
- マップ後 | カウント情報取得 | |について (last modified 2014/12/17)
- マップ後 | カウント情報取得 | single-end | ゲノム | アノテーション有 | QuasR(Gaidatzis_2015) (last modified 2015/02/26)
- マップ後 | カウント情報取得 | single-end | ゲノム | アノテーション無 | QuasR(Gaidatzis_2015) (last modified 2014/06/22)
- マップ後 | カウント情報取得 | paired-end | ゲノム | アノテーション有 | QuasR(Gaidatzis_2015) ② modified 2015/07/03)
- マップ後 | カウント情報取得 | paired-end | ゲノム | アノテーション無 | QuasR(Gaidatzis_2015) (modified 2015/07/02)

Contents

- Illumina HiSeqデータ（トランスクリプトーム）の乳酸菌ゲノムへのマッピング
  - W14：QuasRパッケージを用いたマッピングの事前準備と本番
  - W15：結果の解説、forward側の100–107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16：問題のある領域（forward側の100–107bp）のトリミング
  - W17：トリム後のデータでアセンブリを再実行（Rockhopper2; クラスパス設定関連Tips含む）
  - W18：トリム後のデータでマッピングを再実行（QuasR）
  - W19：トリム前のデータでクオリティチェックを再実行（FastQC）
- Illumina MiSeqデータ（乳酸菌ゲノム）の特徴と前処理
  - W4：FastQC、W5：FaQCs、W6：再度FastQC
- *de novo*ゲノムアセンブリ
  - W7：Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8：k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9：Velvet (ver. 1.2.10)のインストール
  - W10：Velvet (ver. 1.2.10)の実行

第5回原稿PDFのp195

現在、原因（forward側の100–107 bp）が特定された状態。この部分をトリミング（除去）しましょう、というのが次の話。新しいディレクトリ「~/Documents/srp017156_trim1」上にトリム後のファイルなどを置いて作業する

として、FastQC によるクオリティチェックを行えばよい［W1-2］。著者らは、FastQC 実行結果ファイルの項目（Overrepresented sequences）を眺めて、トリム前に見えていた既知のアダプターやプライマー配列が、トリム後に正しく見えなくなっていることを確認して安心している［W1-3］。

このデータに関して結論からいえば、forward 側の 107 bp のリードファイル（SRR616268sub_1.fastq.gz → QC.1.trimmed.fastq）のうち、100–107 塩基付近に乳酸菌に由来しないものがトリムしきれずに多く残っている。これは、アセンブルやマッピングがうまくできない、という実害を被ることでわかる。計算時間がかかるため、できるだけ QC 段階で問題解決するという方針もあろう。しかし、やってみてはじめてわかることもある。以降の内容は、著者らが実際に行ったことを問題解決に至る思考回路とともに述べる。大まかに述べると、Rockhopper2 [18] によるトランスクリプトームアセンブリ、QuasR [19] による乳酸菌ゲノムへのマッピング、そして QC 再実行である。

トランスクリプトームアセンブリ

ゲノムのアセンブリは、断片化されたゲノム配列由来リードをつなぎ合わせて、元のゲノム配列を再構築する作業である。この再構築に相当する英語がアセンブリ（assembly）であり、再構築を行うプログラムをアセンブラ（assembler）という。デノボ（*de novo*）という言葉が同時に用いられることが多いが、これは「最初から」と

か「一から」という意味であ
入力として（つまり他の情報
ルする際には、*de novo* asse
トランスクリプトームアセンブリとは、アセンブル対象がゲノムではなく解析サンプル中で発現している全転写物（トランスクリプトーム）の場合を指す。RNA-seq データのみを入力として一からアセンブルする場合は、*de novo* transcriptome assembly などと呼ばれる。

M
スク
てい
数）
なり
ンブ
ロー
を小
的に
片化
（重
k の
るだ
眼を
複を
お
トー
トラ

このデータに関して結論からいえば、forward 側の 107 bp のリードファイル（SRR616268sub_1.fastq.gz → QC.1.trimmed.fastq）のうち、100–107 塩基付近に乳酸菌に由来しないものがトリムしきれずに多く残っている。これは、アセンブルやマッピングがうまくできない、という実害を被ることでわかる。計算時間がかかるため、できるだけ QC 段階で問題解決するという方針もあろう。しかし、やってみてはじめてわかることもある。以降の内容は、著者らが実際に行ったことを問題解決に至る思考回路とともに述べる。大まかに述べると、Rockhopper2 [18] によるトランスクリプトームアセンブリ、QuasR [19] による乳酸菌ゲノムへのマッピング、そして QC 再実行である。

①

W16-1:トリミング

目的は、forward側リードの100-107塩基付近の乳酸菌に由来しないものを除去。末端8塩基分を除去するためのRスクリプト(次スライドのJSLAB5_6.R)のテンプレートは、①の例題4をベースに作成。スライドを見るだけ

(Rで)塩基配列解析

~NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス~

(last modified 2015/09/12, since 2011)

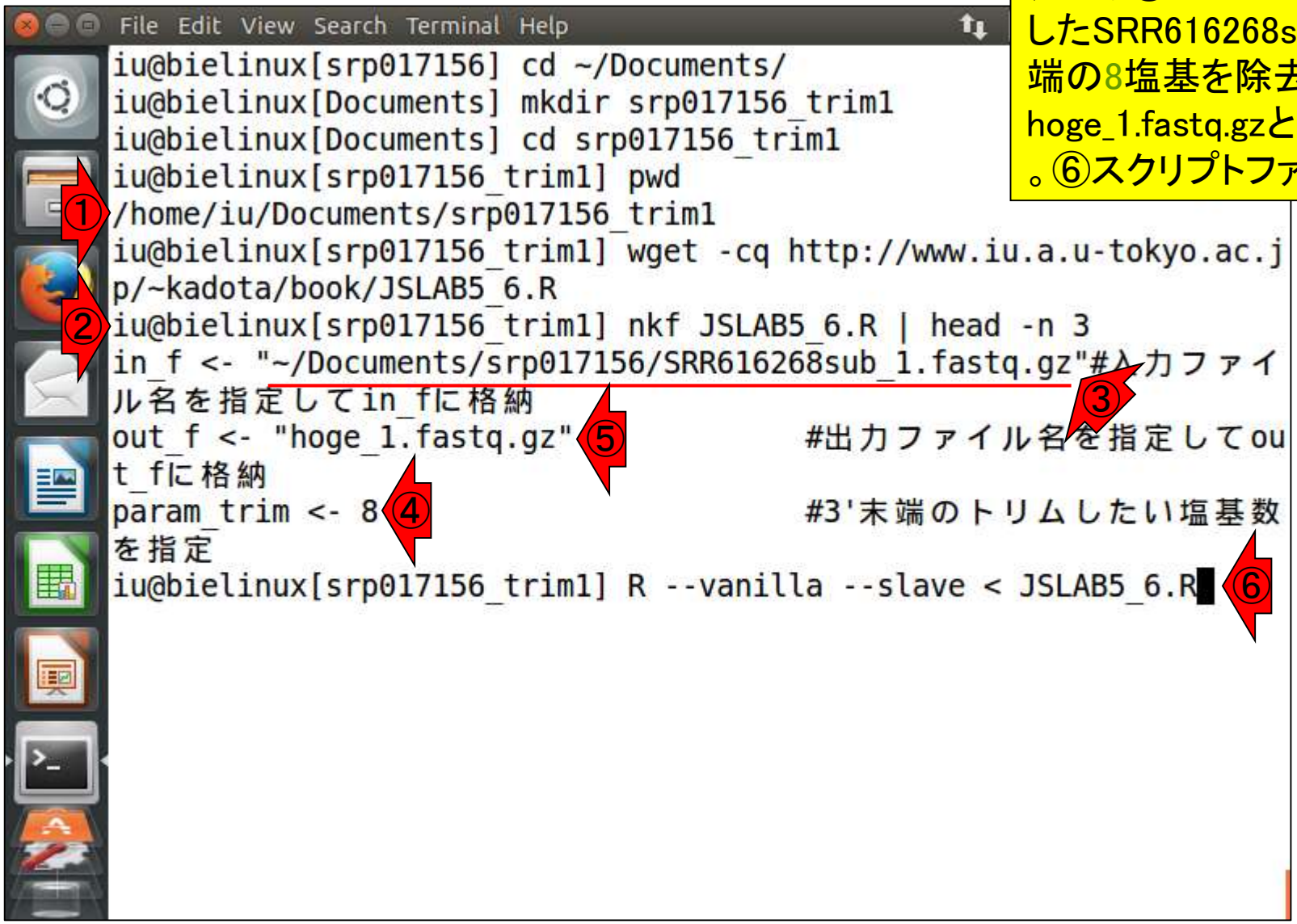
What's new?

- このウェブページはインストール||についての推奨手順 (Windows2015.04.04版とMacintosh2015.04.03版)に従ってフリーソフトRと必要なパッケージをインストール済みであるという前提で記述しています。初心者の方は基本的な利用法(Windows2015.04.03版と Macintosh2015.04.0
  あります。(2015/04/03)
- sample1.fastaのようなコンティグ数が1つしかない
  該当箇所をapply(as.matrix(x), 1, sum)のような感
- NGSハンズオン講習会2015のアメリエフ様分(服
  NEW
- NGSハンズオン講習会のフォローアップ勉強会

- 前処理|クオリティコントロール||について (last modified 2015/08/21) NEW
- 前処理|クオリティチェック|QuasR(Gaidatzis_2015) (last modified 2015/06/15)
- 前処理|クオリティチェック|qrqc (last modified 2014/07/17)
- 前処理|クオリティチェック|PHREDスコアに変換 (last modified 2013/06/18)
- 前処理|クオリティチェック|配列長分布を調べる (last modified 2015/06/22)
- 前処理|クオリティチェック|Overrepresented sequences|ShortRead(Morgan_2009) (last modified 2015/07/29)
- 前処理|トリミング|ポリA配列除去|ShortRead(Morgan_2009) (last modified 2014/06/11)
- 前処理|トリミング|アダプター配列除去(基礎)|QuasR(Gaidatzis_2015) (last modified 2015/06/26)推奨
- 前処理|トリミング|アダプター配列除去(基礎)|girafe(Toedling_2010) (last modified 2014/06/11)
- 前処理|トリミング|アダプター配列除去(基礎)|ShortRead(Morgan_2009) (last modified 2014/06/21)
- 前処理|トリミング|アダプター配列除去(応用)|QuasR(Gaidatzis_2015) (last modified 2015/06/26)推奨
- 前処理|トリミング|アダプター配列除去(応用)|ShortRead(Morgan_2009) (last modified 2015/09/12) NEW
- 前処理|トリミング|指定した末端塩基数だけ除去 ① modified 2015/06/29)
- 前処理|フィルタリング|PHREDスコアが低い塩基をNに置換 (last modified 2014/03/03)
- 前処理|フィルタリング|PHREDスコアが低い配列(リード)を除去 (last modified 2014/08/27)
- 前処理|フィルタリング|ACGTのみからなる配列を抽出 (last modified 2015/09/12) NEW
- 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)
- 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) NEW
- 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)

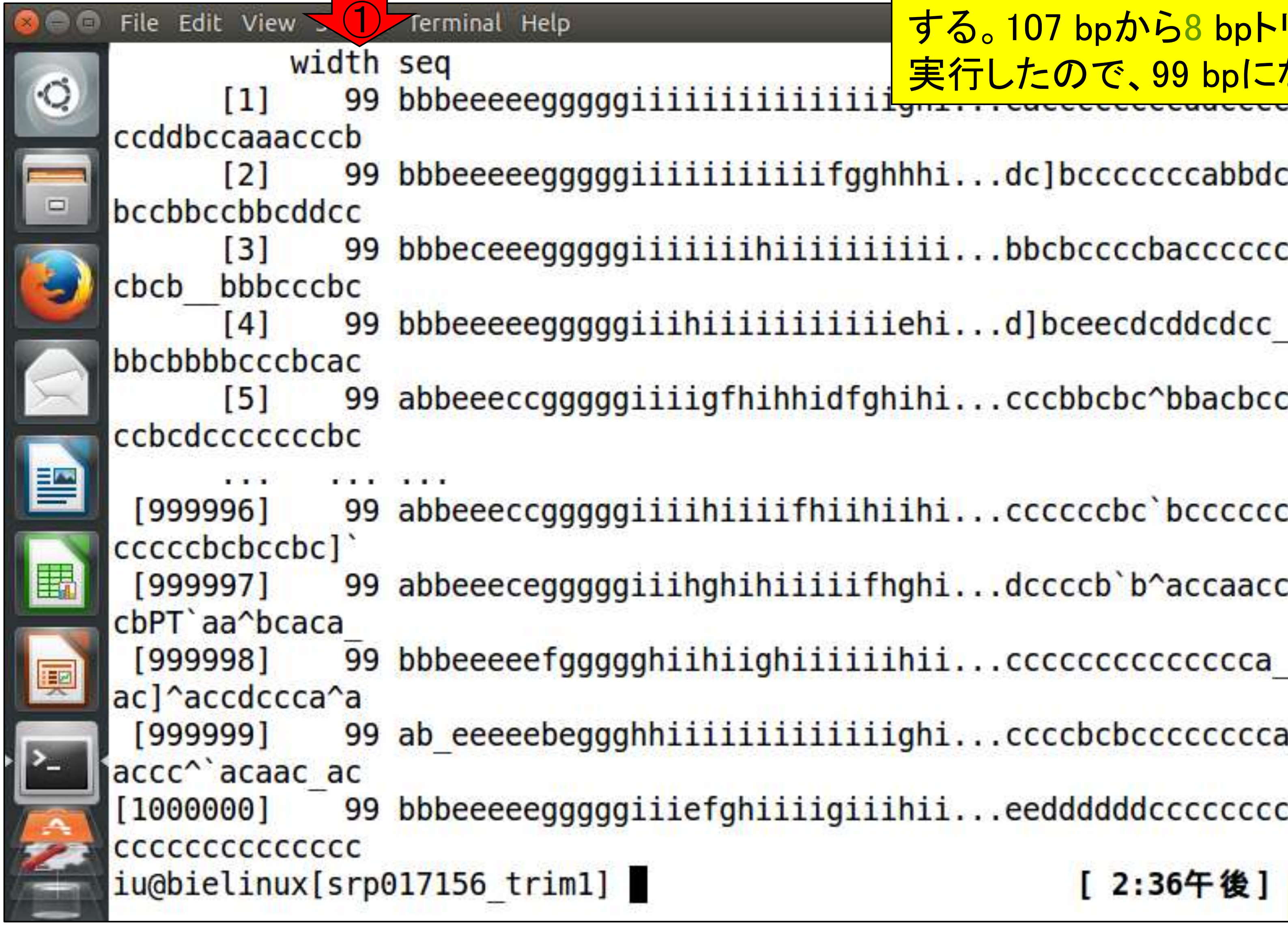
W16-1:トリミング

①作業ディレクトリはここ。②ダウンロードしたJSLAB5_6.Rの最初の3行分を表示。③入力ファイルは相対パスで示したSRR616268sub_1.fastq.gz。④3'末端の8塩基を除去した結果を⑤hoge_1.fastq.gzというファイル名で保存。⑥スクリプトファイルの実行。約10秒



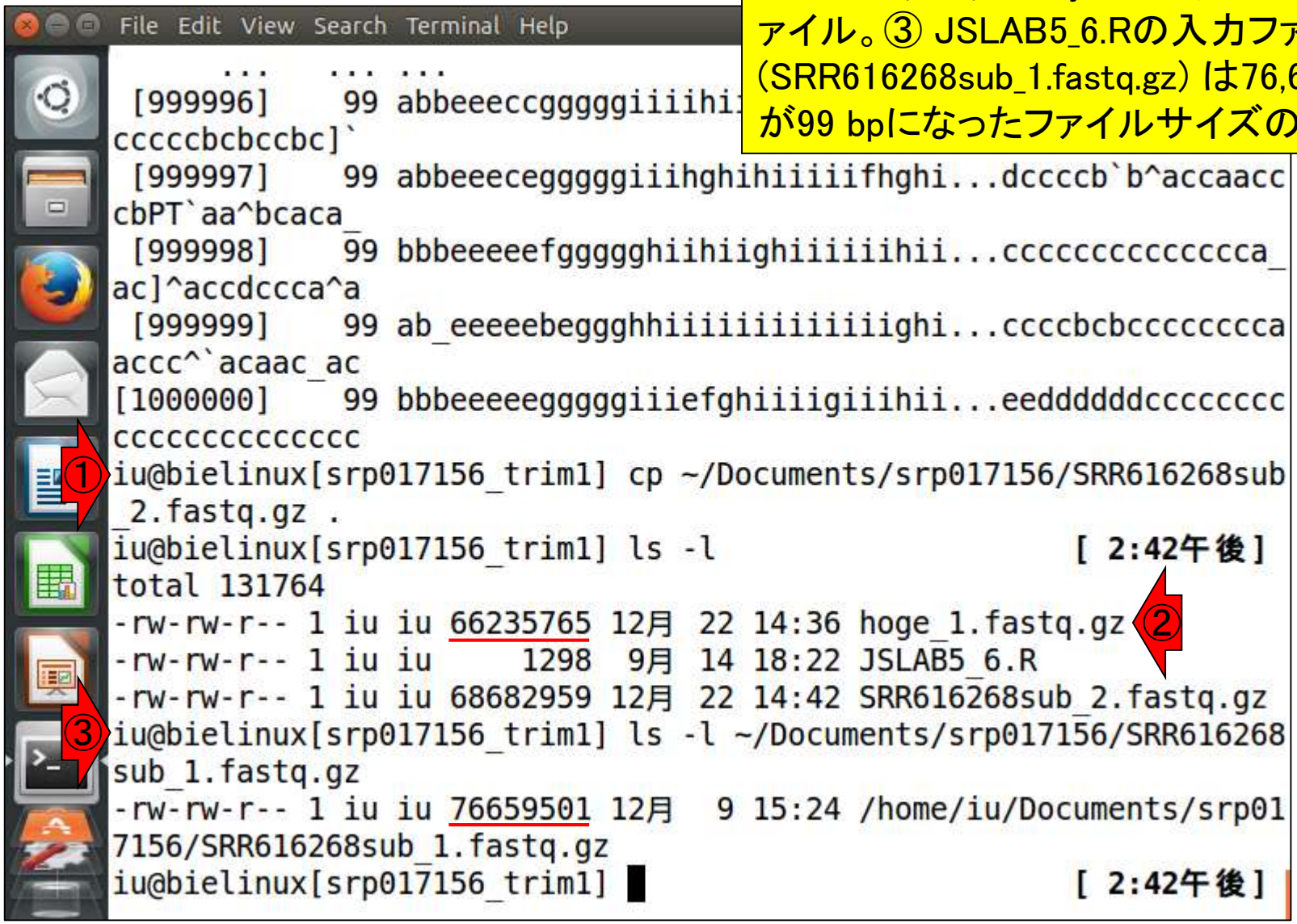
W16-1:トリミング

スクリプトファイル実行後の状態。①width列の数値が99になっているのがわかる。これは、トリミング後のリード長が99 bpであることを意味する。107 bpから8 bpトリムするプログラムを実行したので、99 bpになっているのは妥当



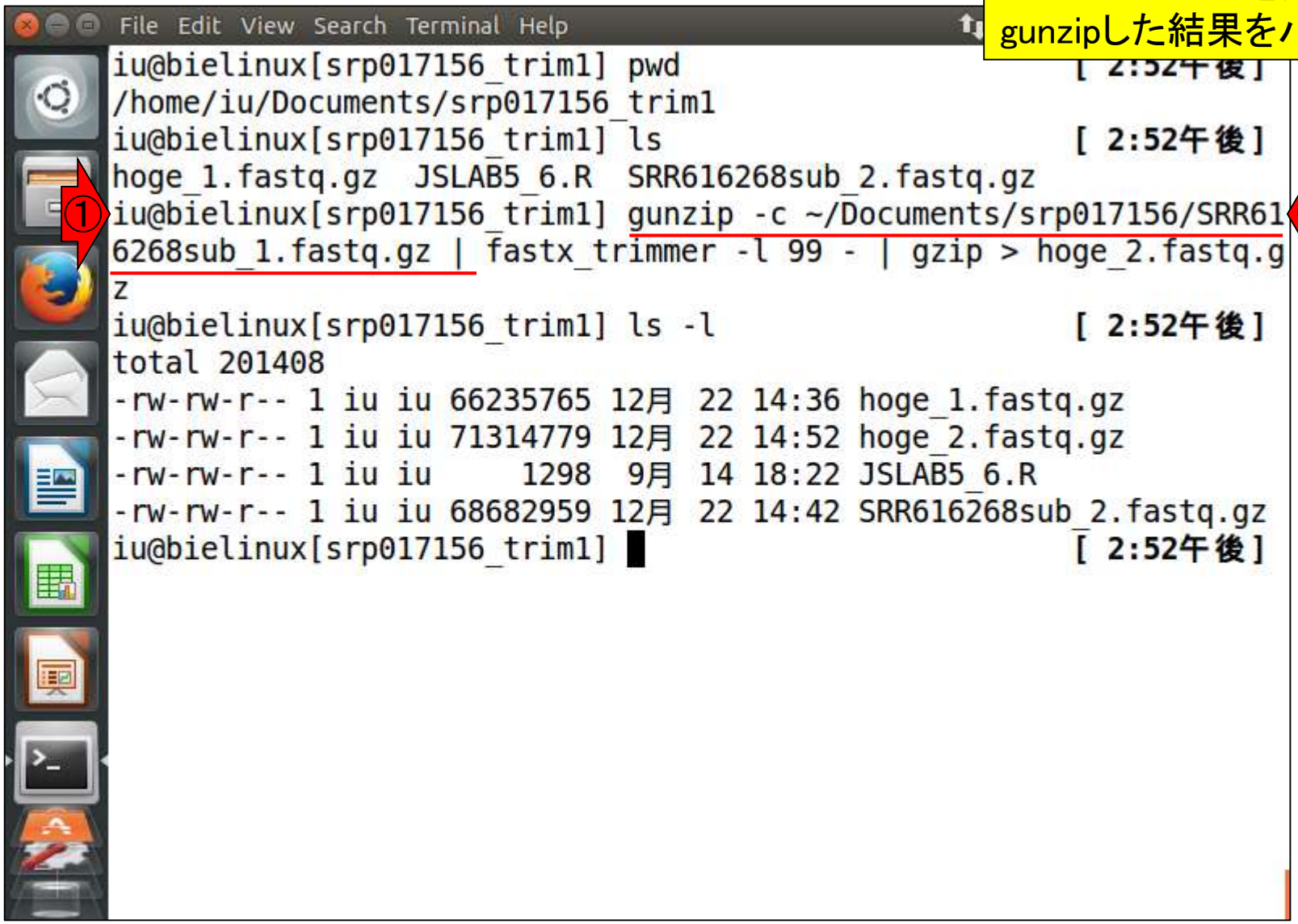
W16-1:トリミング

①reverse側のファイルとしてSRR616268sub_2.fastq.gzを作業ディレクトリにコピー。②hoge_1.fastq.gz (ファイルサイズ66,235,765 bytes)は、JSLAB5_6.Rの実行結果ファイル。③ JSLAB5_6.Rの入力ファイル(SRR616268sub_1.fastq.gz) は76,659,501 bytes。107 bpが99 bpになったファイルサイズの減少度合い的に妥当



W16-2:トリミング

①FASTX-Toolkitのfastx_trimmerを利用するやり方。②fastx_trimmerはgzip圧縮ファイルの入力を受け付けないので、gunzipした結果をパイプで渡している

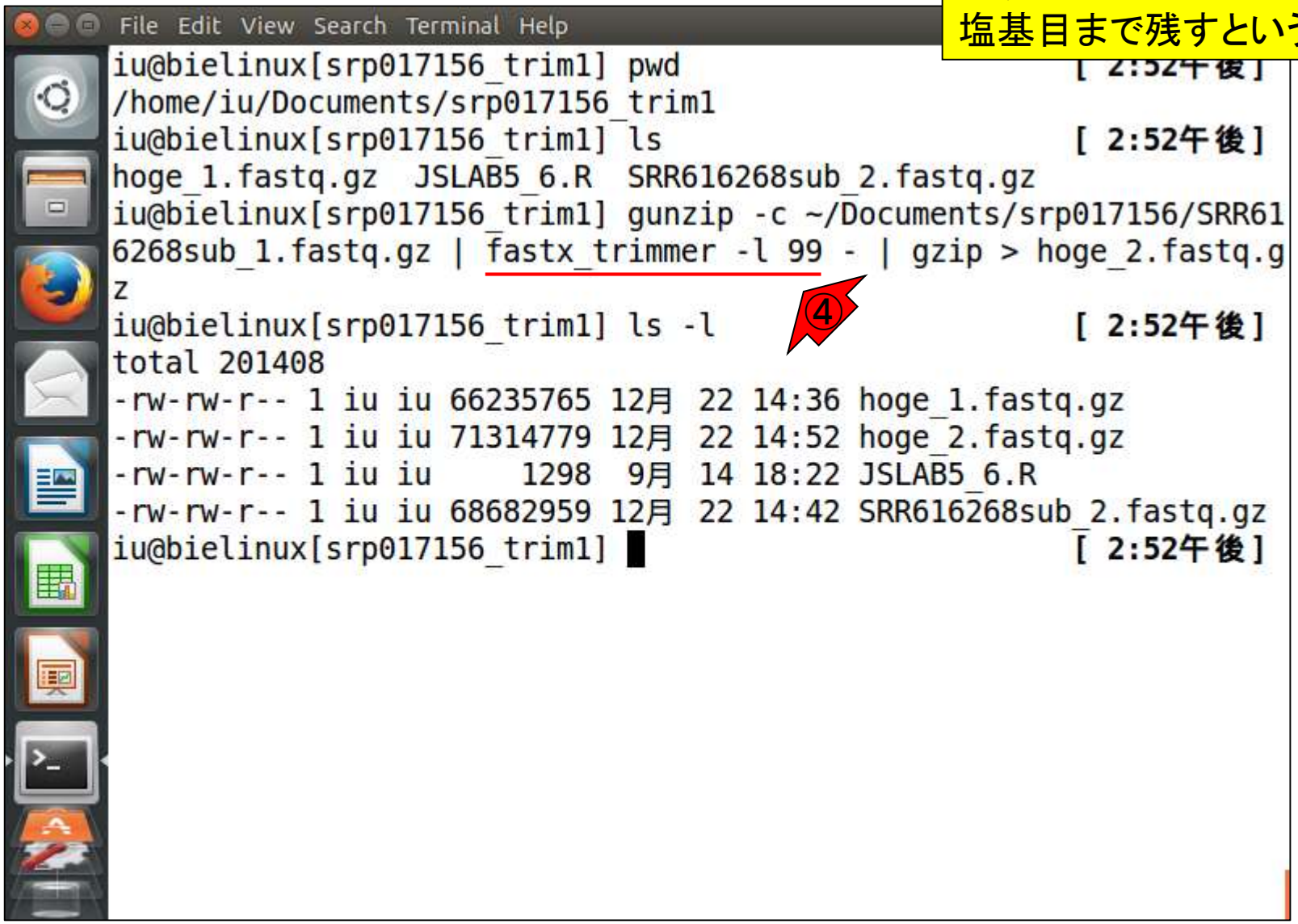


W16-2:トリミング

③この「- (ハイフン)」は、パイプで渡したものを入力として受け付けるという明示的な意思表示。省略することができるコマンド(or プログラム)もあるが、fastx_trimmerは明示しないと怒られるのでつけている

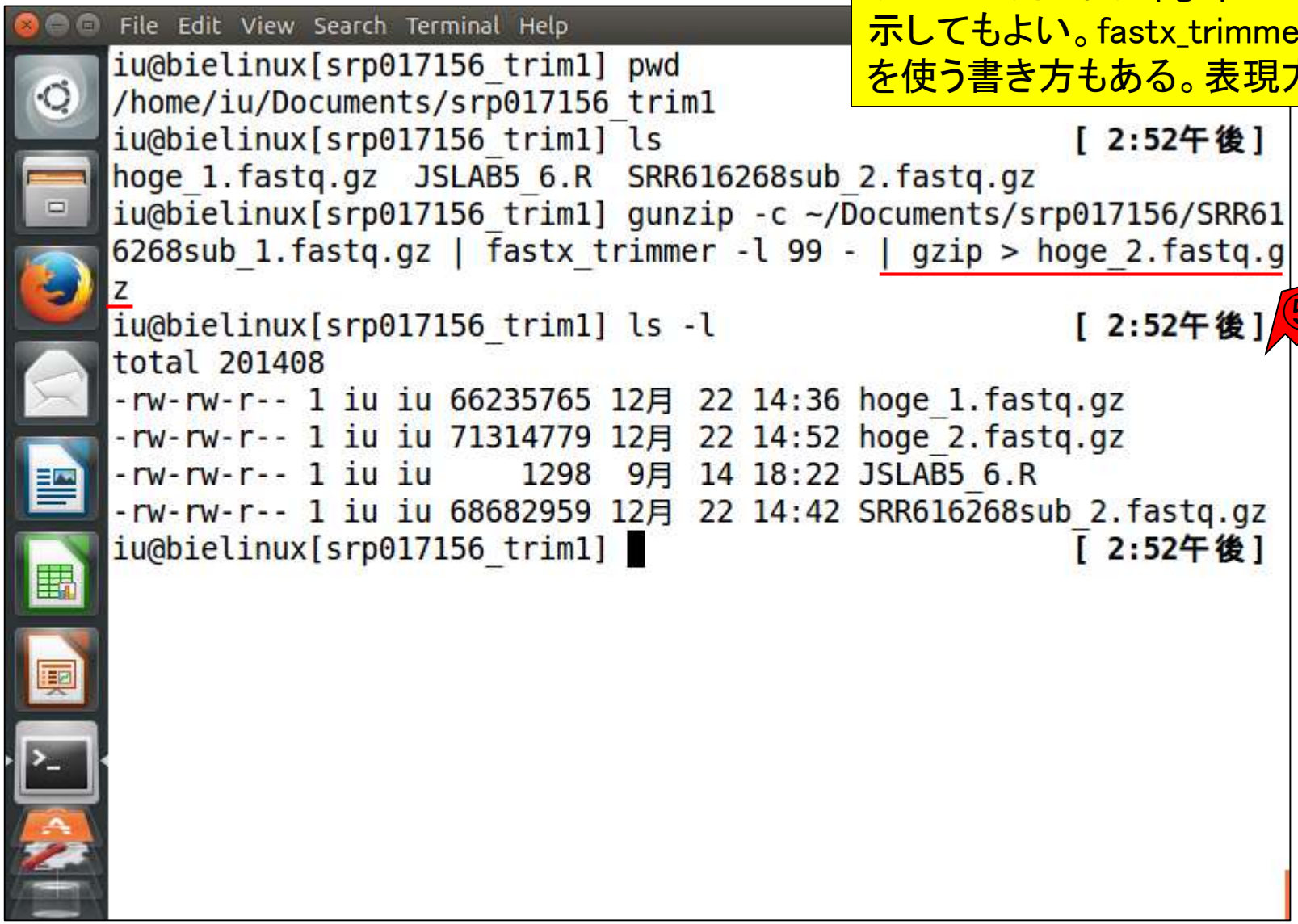
W16-2:トリミング

④fastx_trimmer実行の本体部分。-lでリードの何塩基目までを残すかを指定。ここでは、(100塩基目以降をトリムしたいので)99塩基目まで残すという指定を行っている



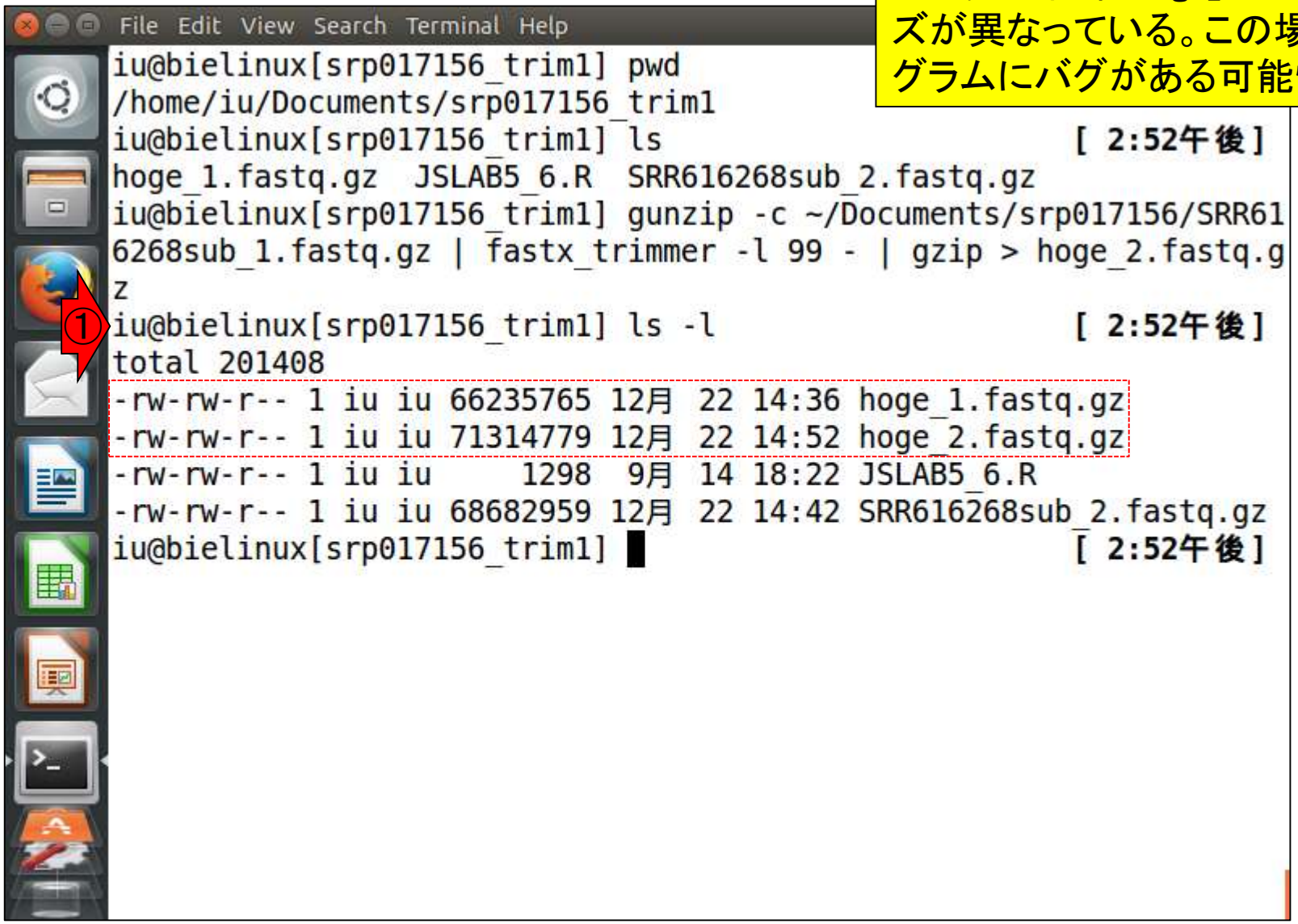
W16-2：トリミング

⑤ここでは、トリムした結果をパイプで流してgzip圧縮してhoge_2.fastq.gzというファイル名で保存するという指令。「| gzip － > …」とハイフン(－)を明示してもよい。fastx_trimmerの−zや−oオプションを使う書き方もある。表現方法はいろいろ



W16-3：確認

①ls -lでファイルサイズを確認。RのBiostringsでの実行結果(hoge_1.fastq.gz)とfastx_trimmerでの実行結果(hoge_2.fastq.gz)のファイルサイズが異なっている。この場合、どちらかのプログラムにバグがある可能性を考えるのが自然

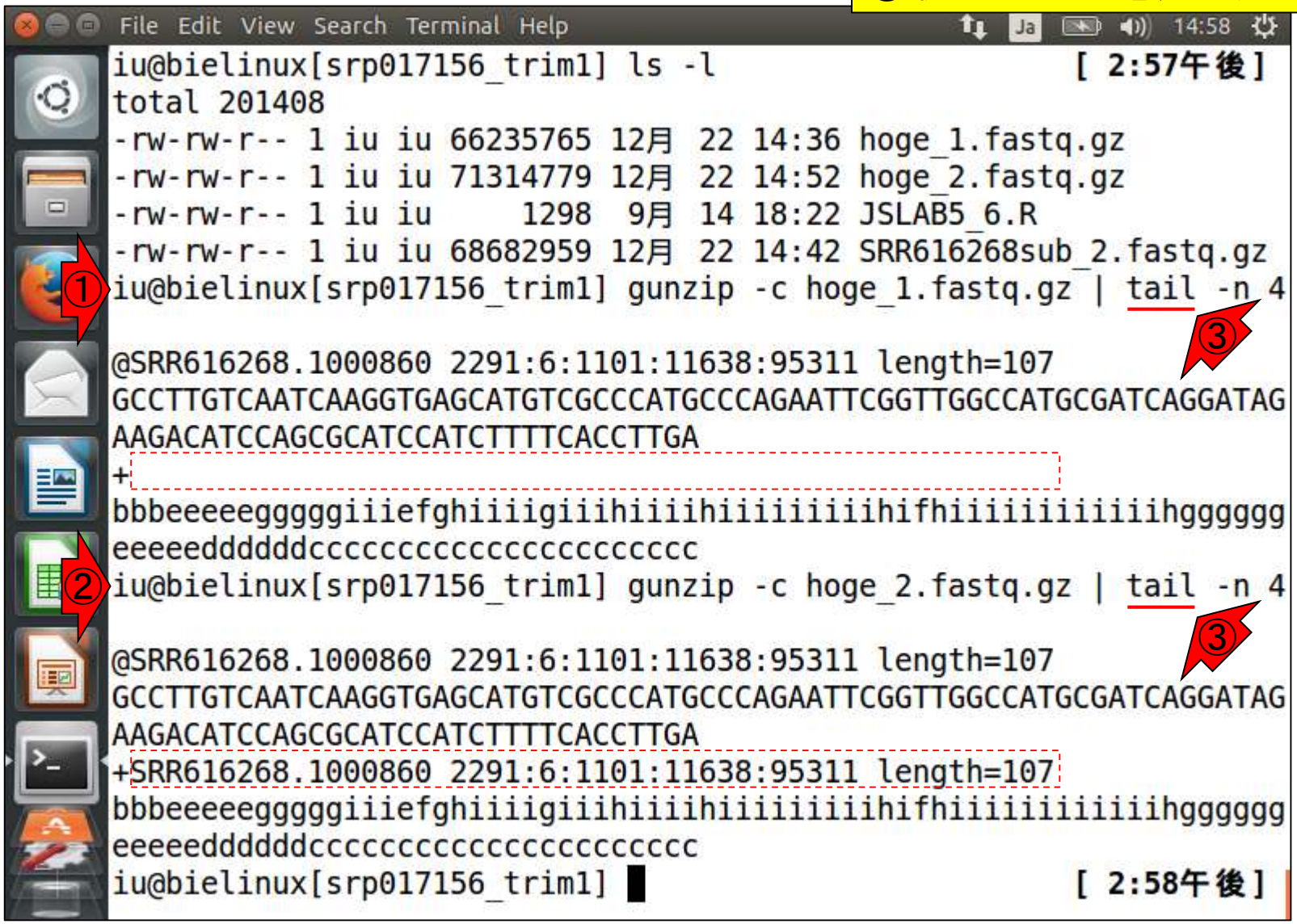


W16-3：確認

①RのBiostringsでの実行結果(hoge_1.fastq.gz)と②fastx_trimmerでの実行結果(hoge_2.fastq.gz)の③最初の4行分を表示。両者の違いは赤枠のdescription情報の有無だけのようだ。バグではなさそう

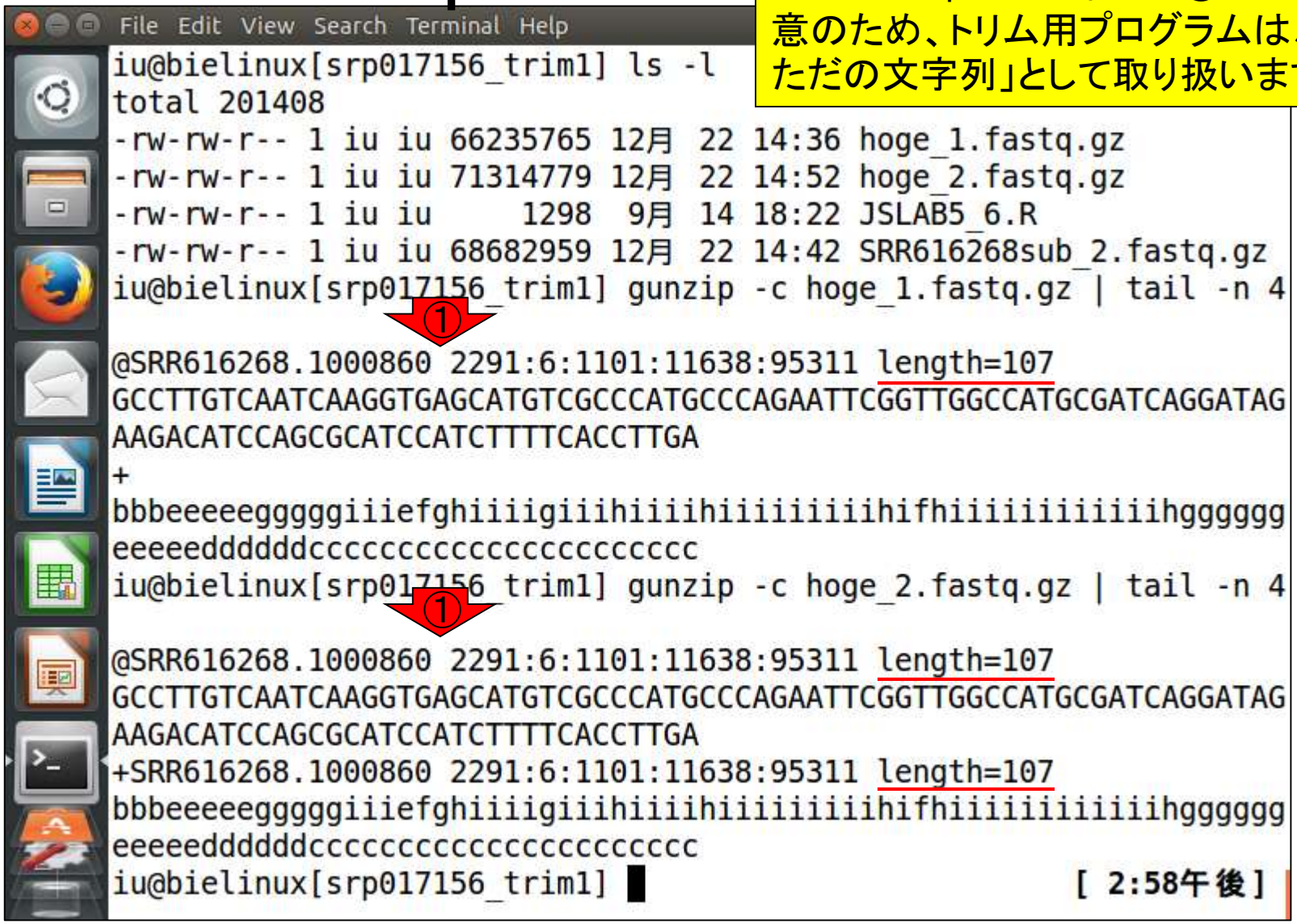
W16-3：確認

①RのBiostringsでの実行結果(hoge_1.fastq.gz)と
②fastx_trimmerでの実行結果(hoge_2.fastq.gz)の
③最後の4行分を表示。大丈夫そうだ



W16-4 : Tips

素朴な疑問として、よく赤下線部分の「description情報の記述が変わってないけど…」という質問が出ます。これはdescription行部分の①スペース以降の記述は任意のため、トリム用プログラムは、この赤下線部分は「ただの文字列」として取り扱います。そんなもんです

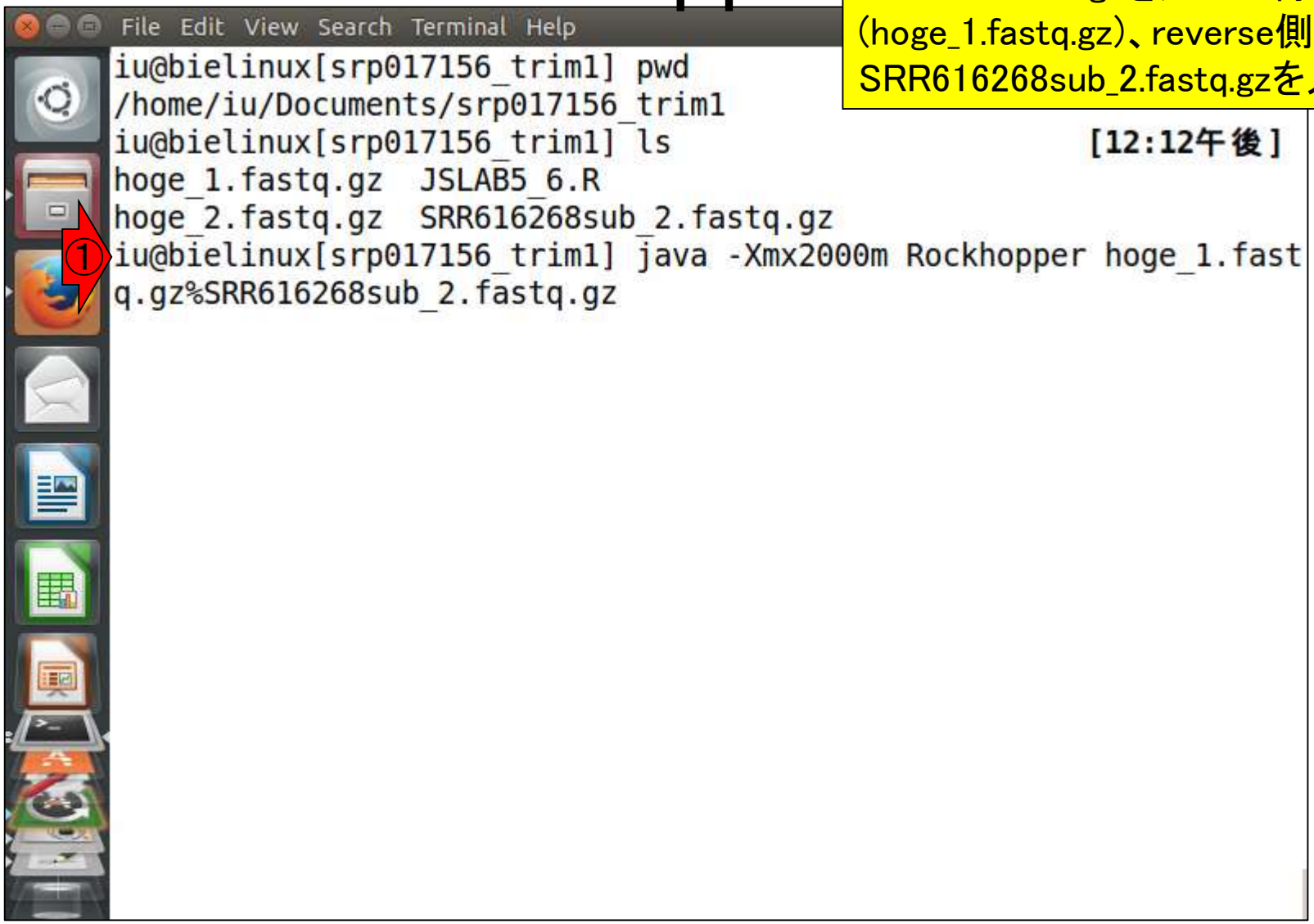


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

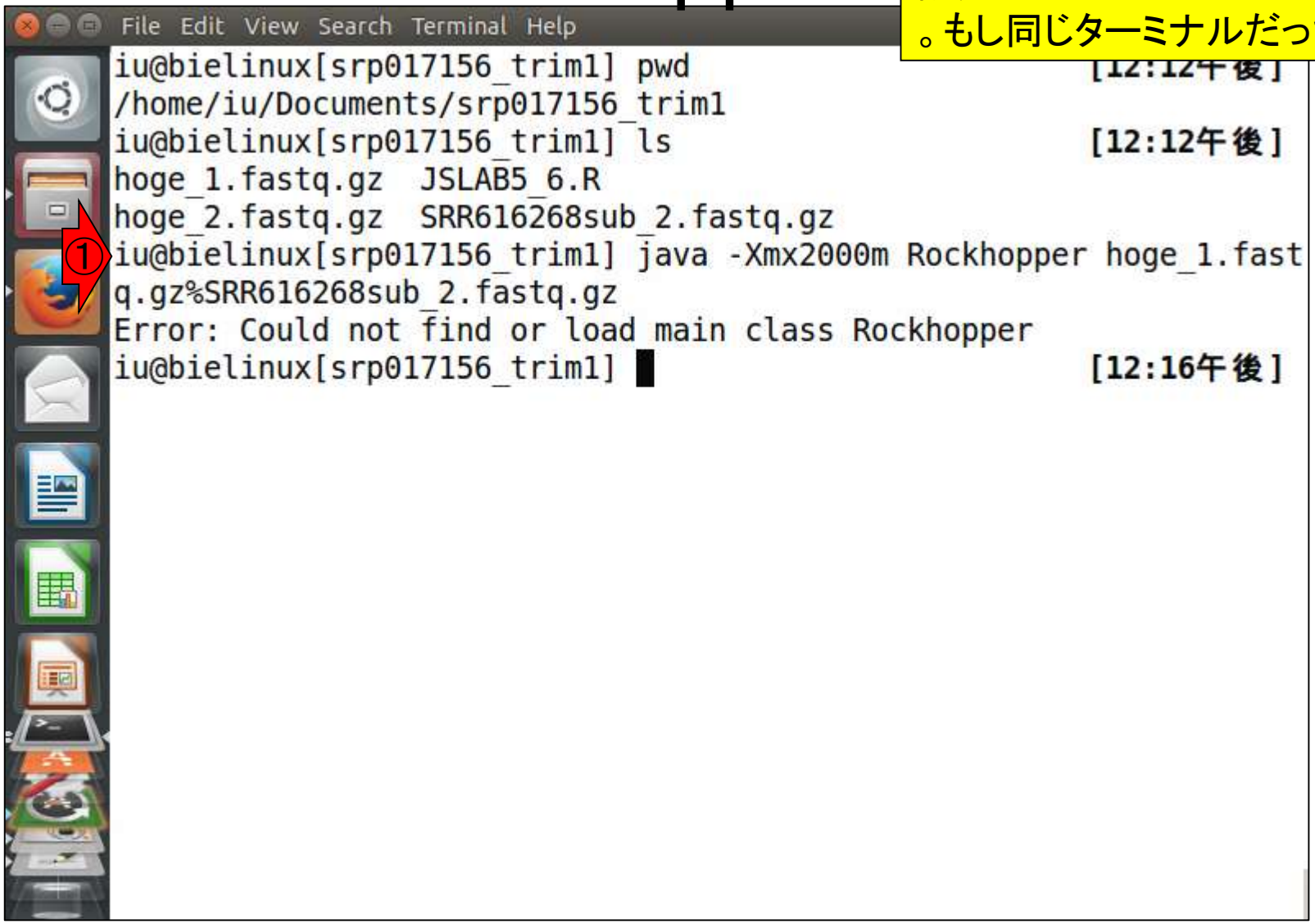
W17-1：Rockhopper

②Rockhopper2による*de novo* transcriptome assemblyをトリム後のデータで再実行。forward側はRのBiostringsを用いて得られたファイル(hoge_1.fastq.gz)、reverse側は特に何もしていないSRR616268sub_2.fastq.gzを入力として与えている



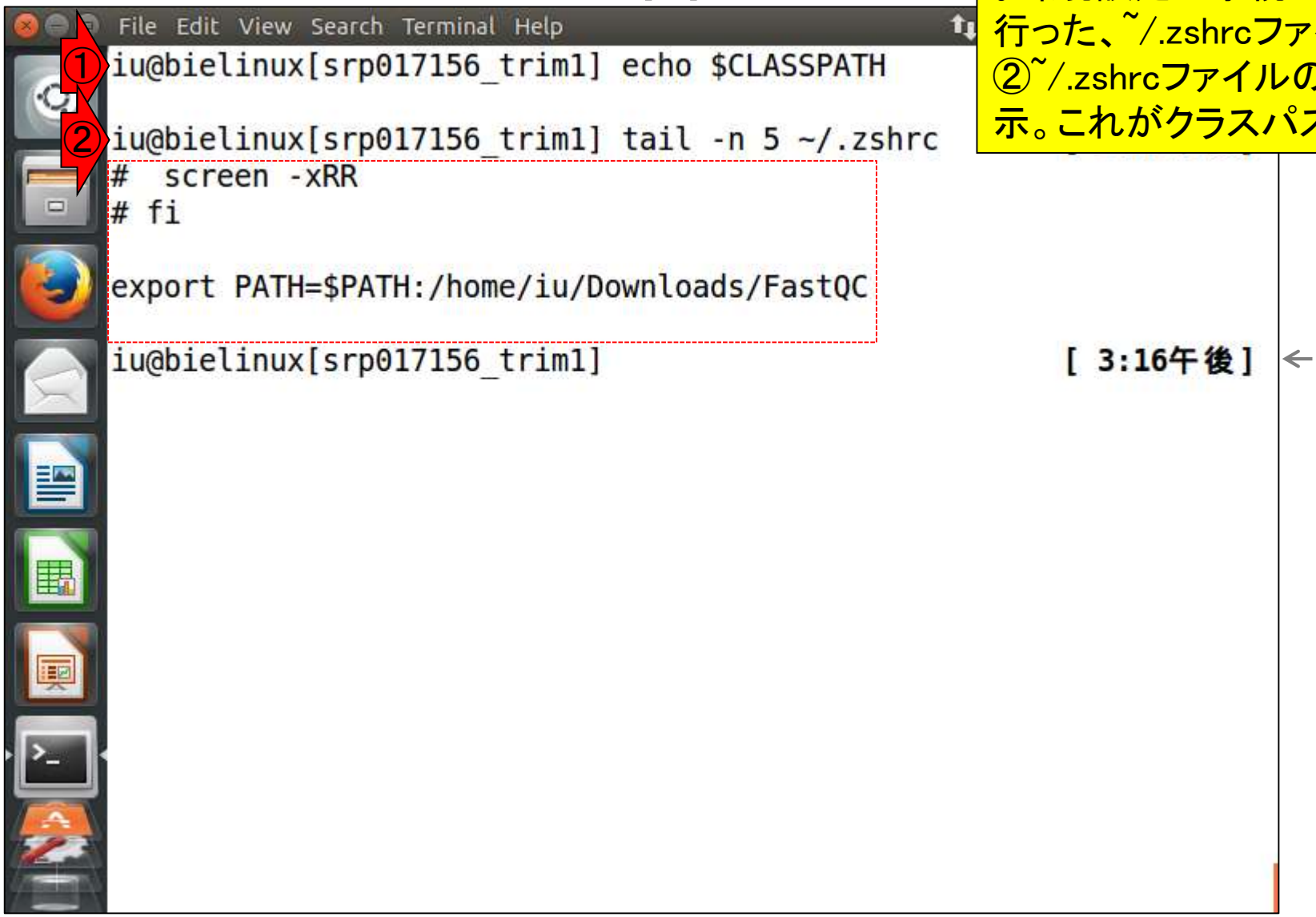
W17-1：Rockhopper

実行結果。エラーが出ていることがわかる。これは、今実行したターミナルは、クラスパスを設定したターミナル(W4)とは異なるものだから。もし同じターミナルだったら、エラーは出ない



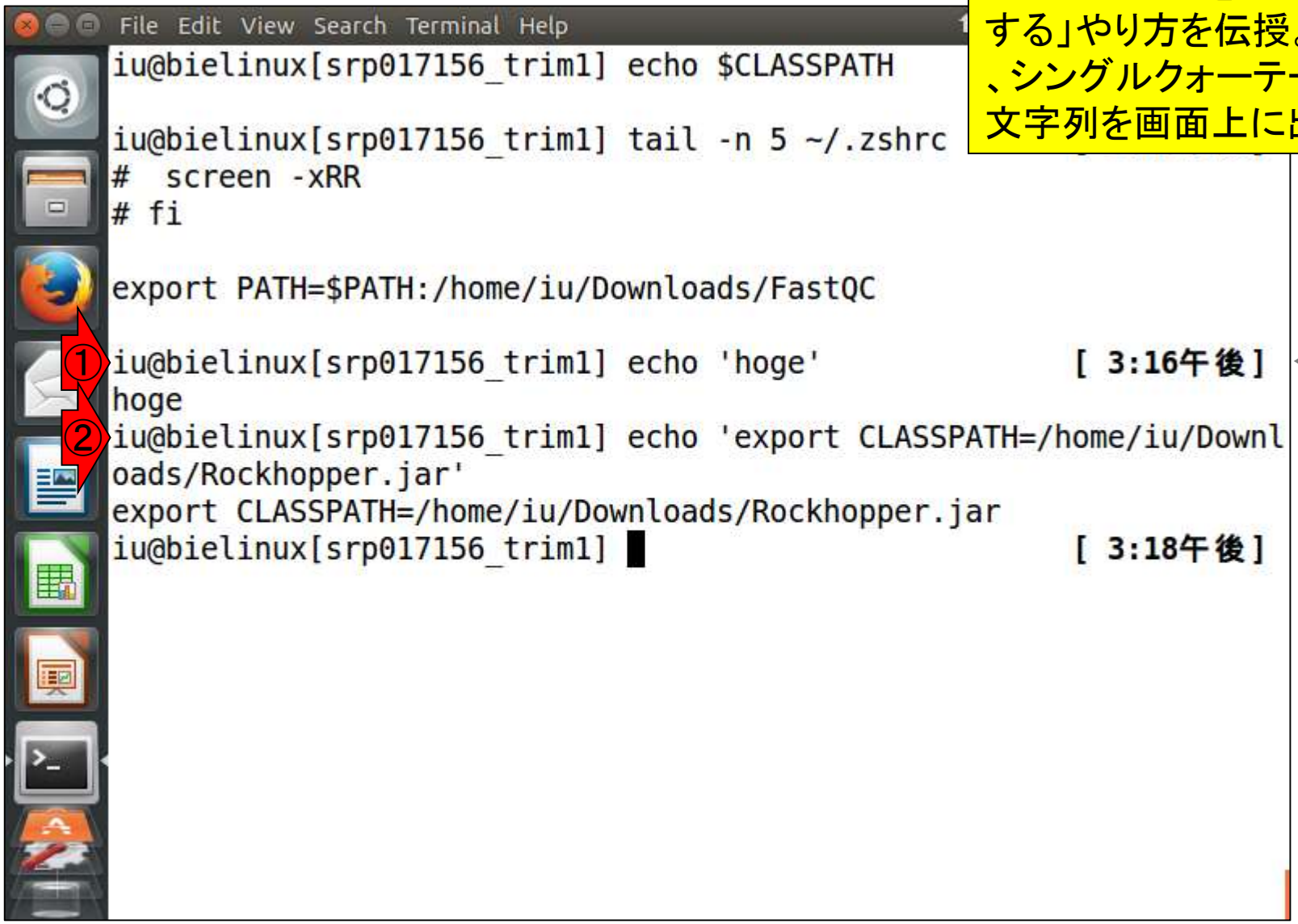
W17-2：echoで書き込み

①W4で設定したクラスパスが、このターミナルでは無効になっていることを確認。環境設定の永続化は、第4回W10-3で行った、~/.zshrcファイルへの書き込み。
②~/.zshrcファイルの最後の5行分を表示。これがクラスパス書き込み前の状態



W17-2：echoで書き込み

「gedit ~/.zshrc」で.zshrcファイルを編集してもよいが、せっかくなので「echoで表示させた文字列をファイルに追加書き込みする」やり方を伝授。①や②で示すように、シングルクォーテーション(‘)で囲まれた文字列を画面上に出力するのがecho

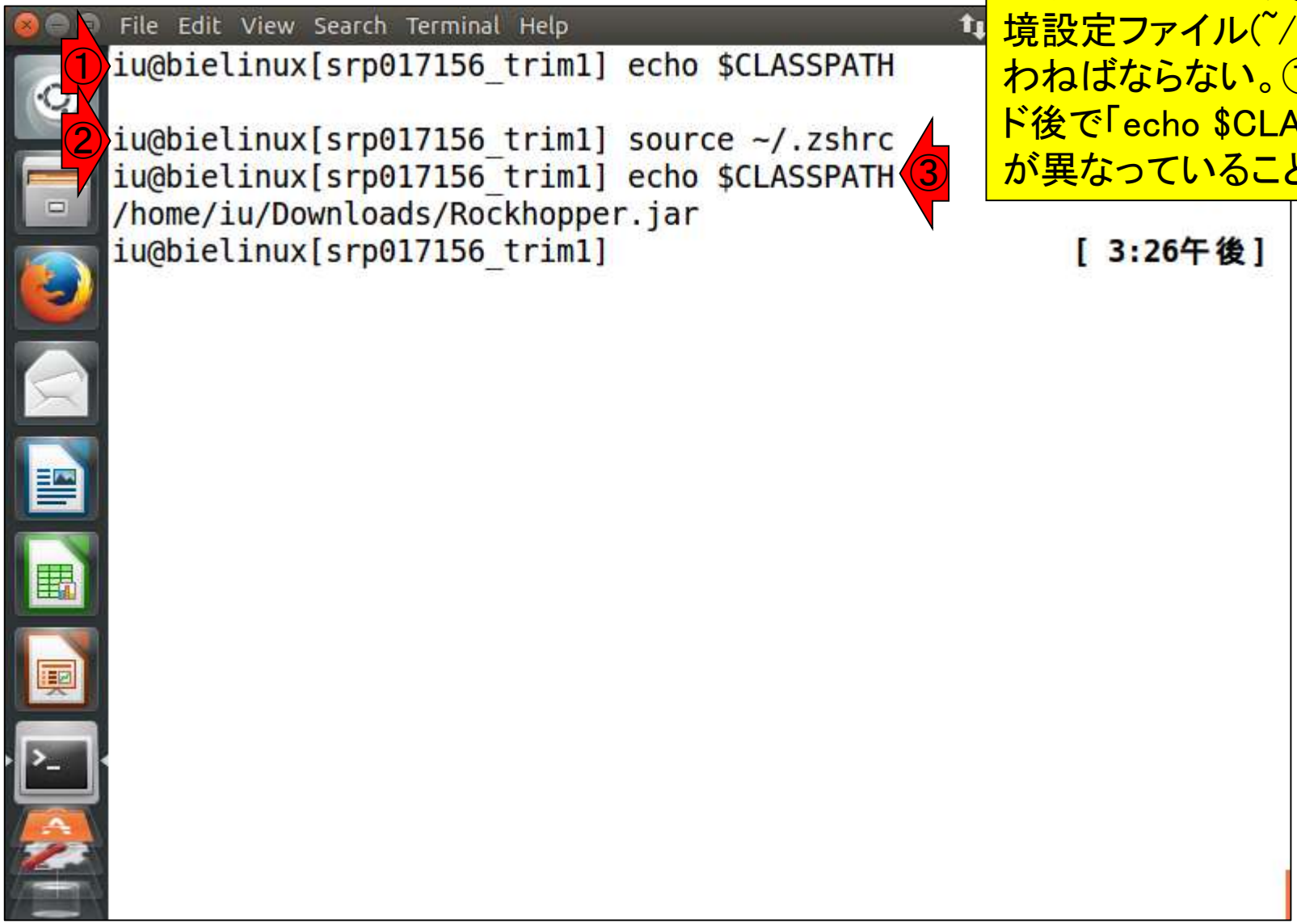


W17-2:>>で追加書き込み

```
File  Edit  View  Search  Terminal  Help                                  Ja
iu@bielinux[srp017156_trim1] echo $CLASSPATH

iu@bielinux[srp017156_trim1] tail -n 5 ~/.zshrc
#  screen -xRR
# fi

export PATH=$PATH:/home/iu/Downloads/FastQC

iu@bielinux[srp017156_trim1] echo 'hoge'                          [ 3:16午後 ]
hoge
iu@bielinux[srp017156_trim1] echo 'export CLASSPATH=/home/iu/Downl
oads/Rockhopper.jar'
export CLASSPATH=/home/iu/Downloads/Rockhopper.jar
①iu@bielinux[srp017156_trim1] echo 'export CLASSPATH=/home/iu/Downl
oads/Rockhopper.jar' >> ~/.zshrc
②iu@bielinux[srp017156_trim1] tail -n 5 ~/.zshrc                  [ 3:19午後 ]
# fi

export PATH=$PATH:/home/iu/Downloads/FastQC

export CLASSPATH=/home/iu/Downloads/Rockhopper.jar
iu@bielinux[srp017156_trim1]                                      [ 3:19午後 ]
```

①echoで表示させた、~/.zshrcファイルの最後に書き込みたい内容を「>>」で追加書き込み。「>」では追加ではなく上書きになってしまうので注意!「cp ~/.zshrc ~/.zshrc_org」などとしてバックアップファイルを作成しておくほうがいいかもしれない。②追加書き込み後にtailコマンドで最後の5行分を再表示。追加書き込みが正常終了

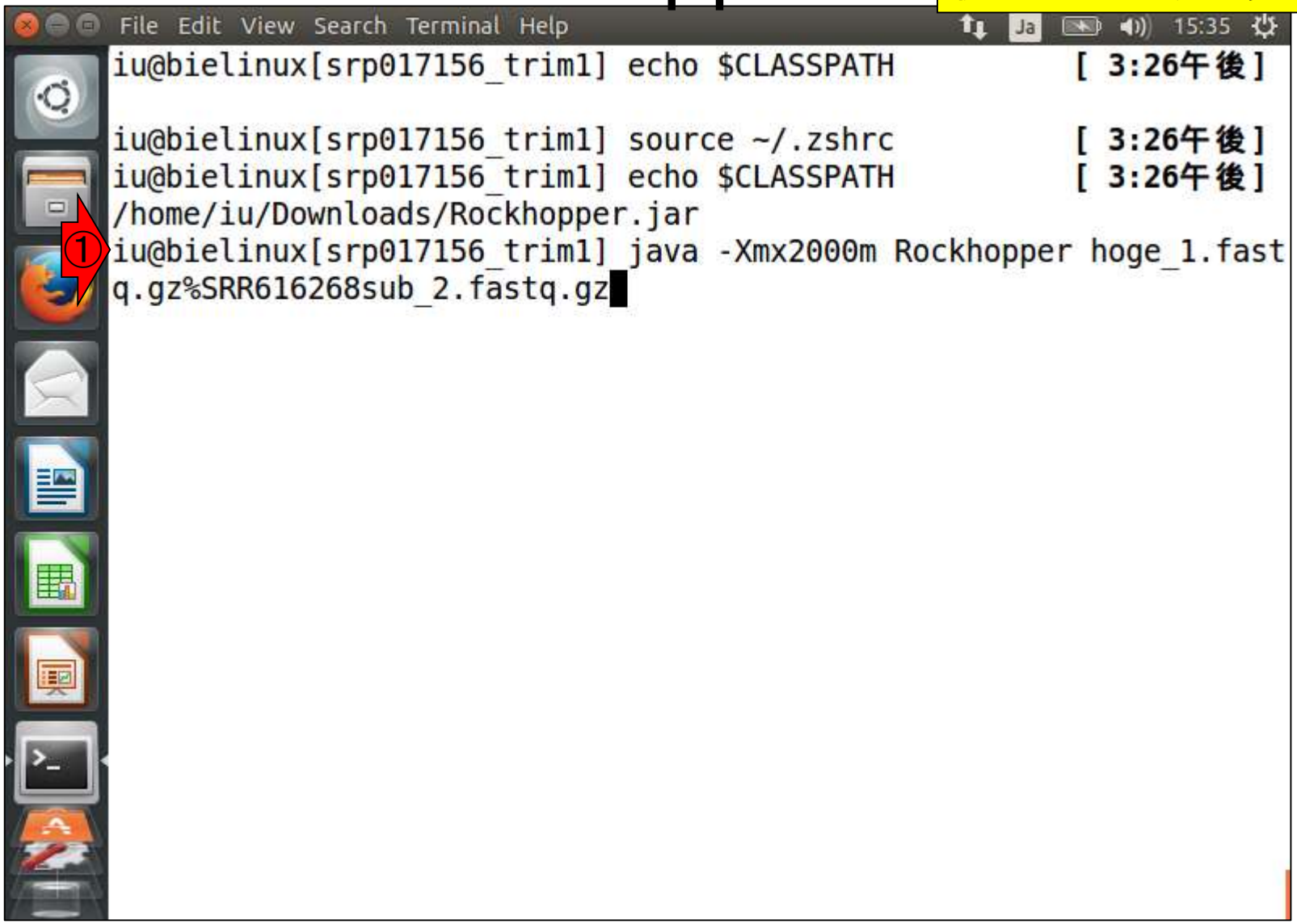
W17-3：sourceして確認



ただの復習(第4回のW10-4)。~/.zshrcにきちんと書き込みできたとしても、それを反映させるには、②sourceを実行して環境設定ファイル(~/.zshrc)のリロードを行わねばならない。①リロード前と③リロード後で「echo $CLASSPATH」実行結果が異なっていることがわかる

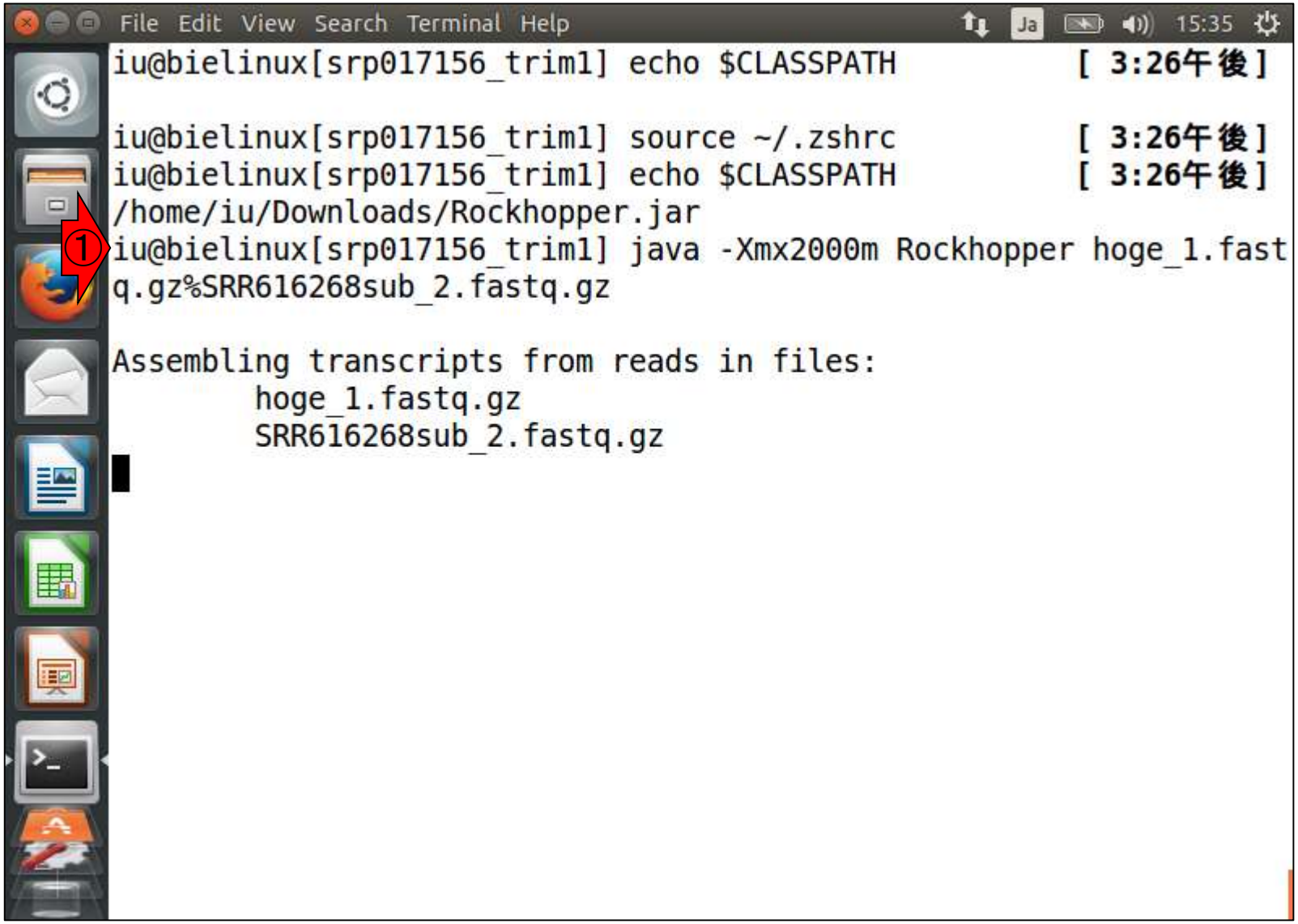
W17-4：Rockhopper

Rockhopper2を再々トライ。RのBiostrings を利用したファイルhoge_1.fastq.gzをforward側として入力する場合(paired-end)。約2分



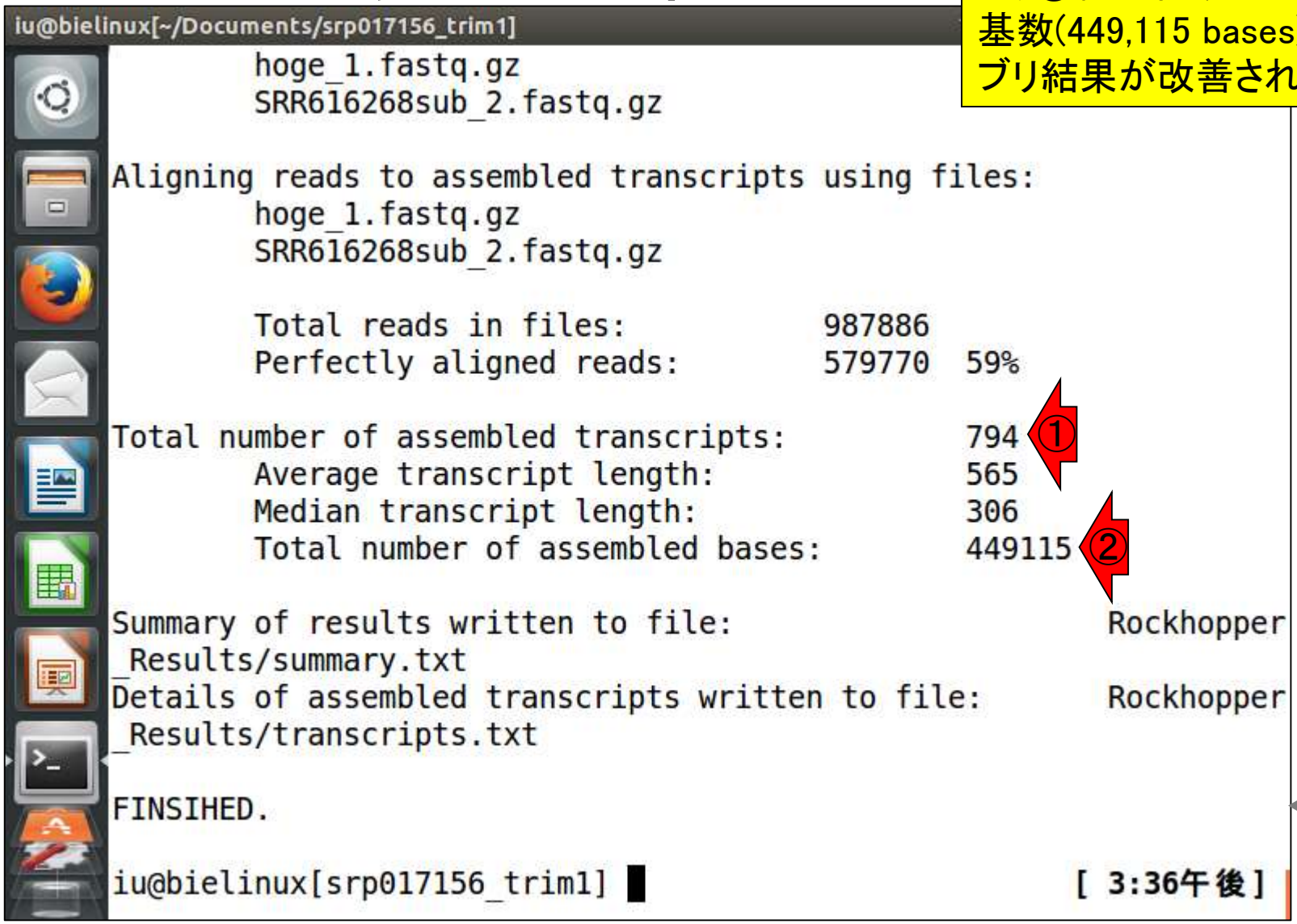
今度はうまく動いているようだ

W17-4：途中経過

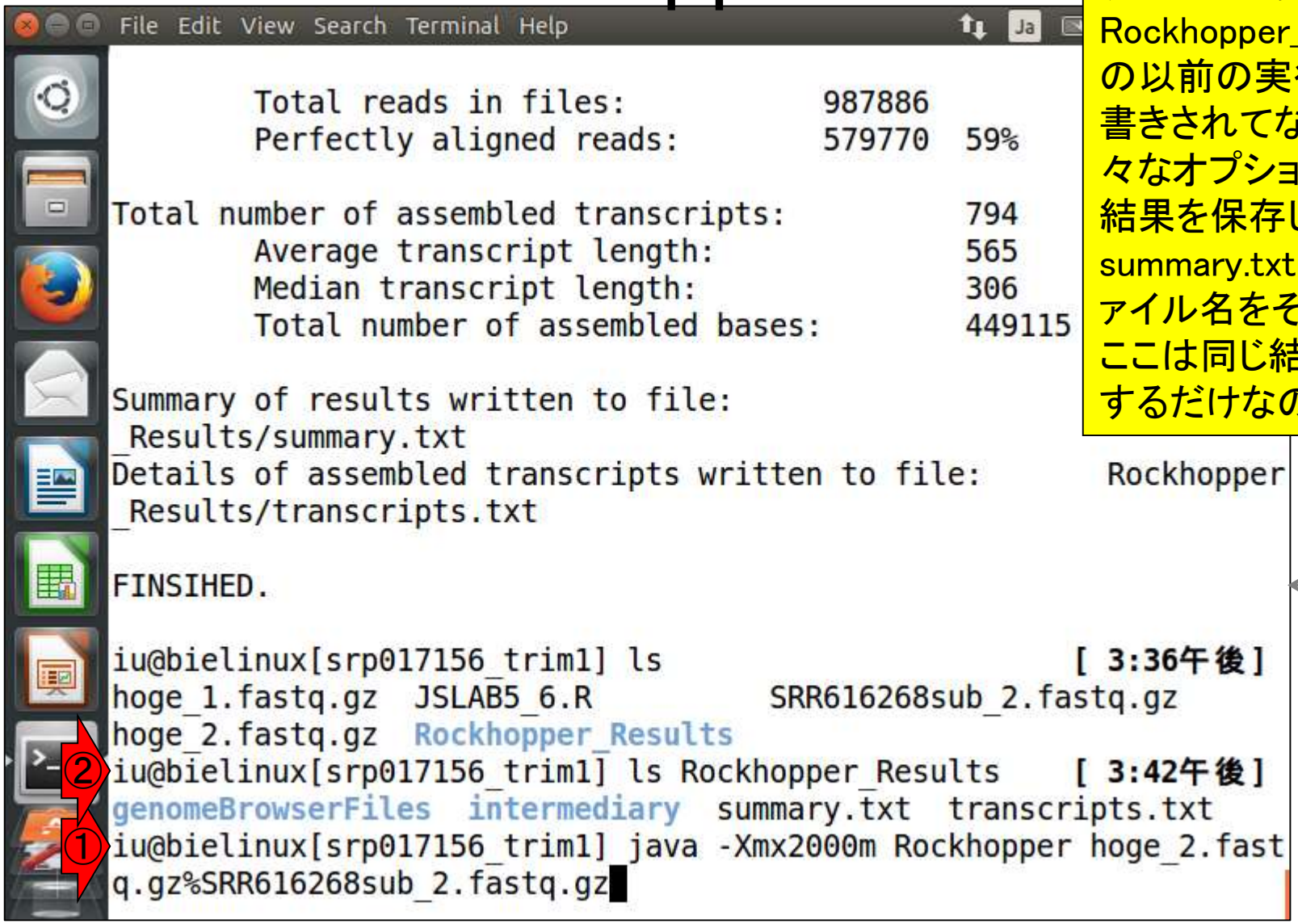


W17-4：実行結果

トリム前の無残な結果(W5-2)やreverse側のsingle-endのみの結果(W6-4)と比べても、①転写物数(794 transcripts)や②総塩基数(449,115 bases)の点で劇的にアセンブリ結果が改善されたことがわかる!



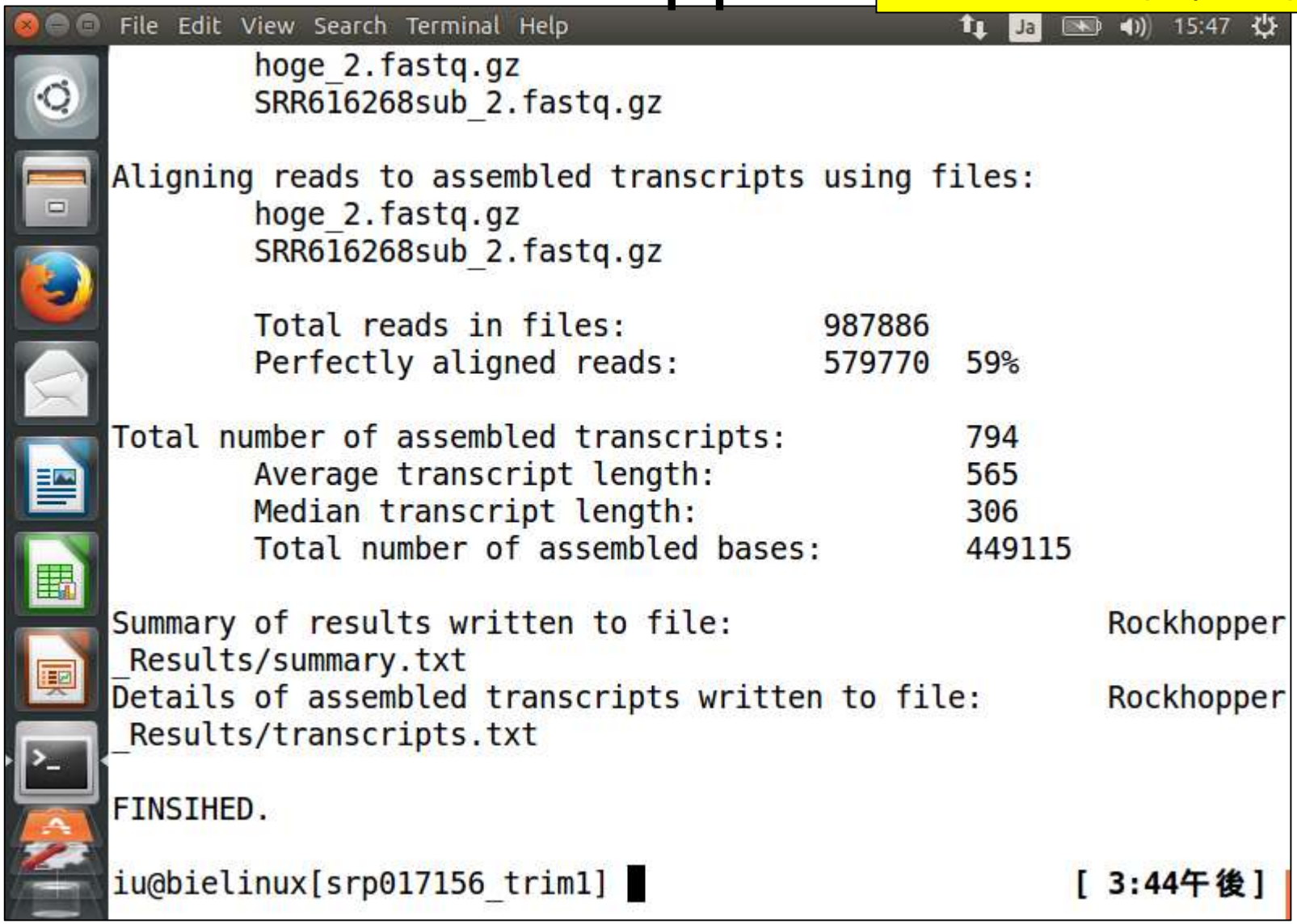
W17-5：Rockhopper



①fastx_trimmerでの実行結果ファイル(hoge_2.fastq.gz)を入力として、念のため実行。②Rockhopper_Resultsディレクトリ中の以前の実行結果ファイルは上書きされてなくなるので注意！様々なオプションや入力ファイルの結果を保存したい場合は、summary.txtやtranscripts.txtのファイル名をその都度変更しておく。ここは同じ結果になることを確認するだけなので気にしない

W17-5：Rockhopper

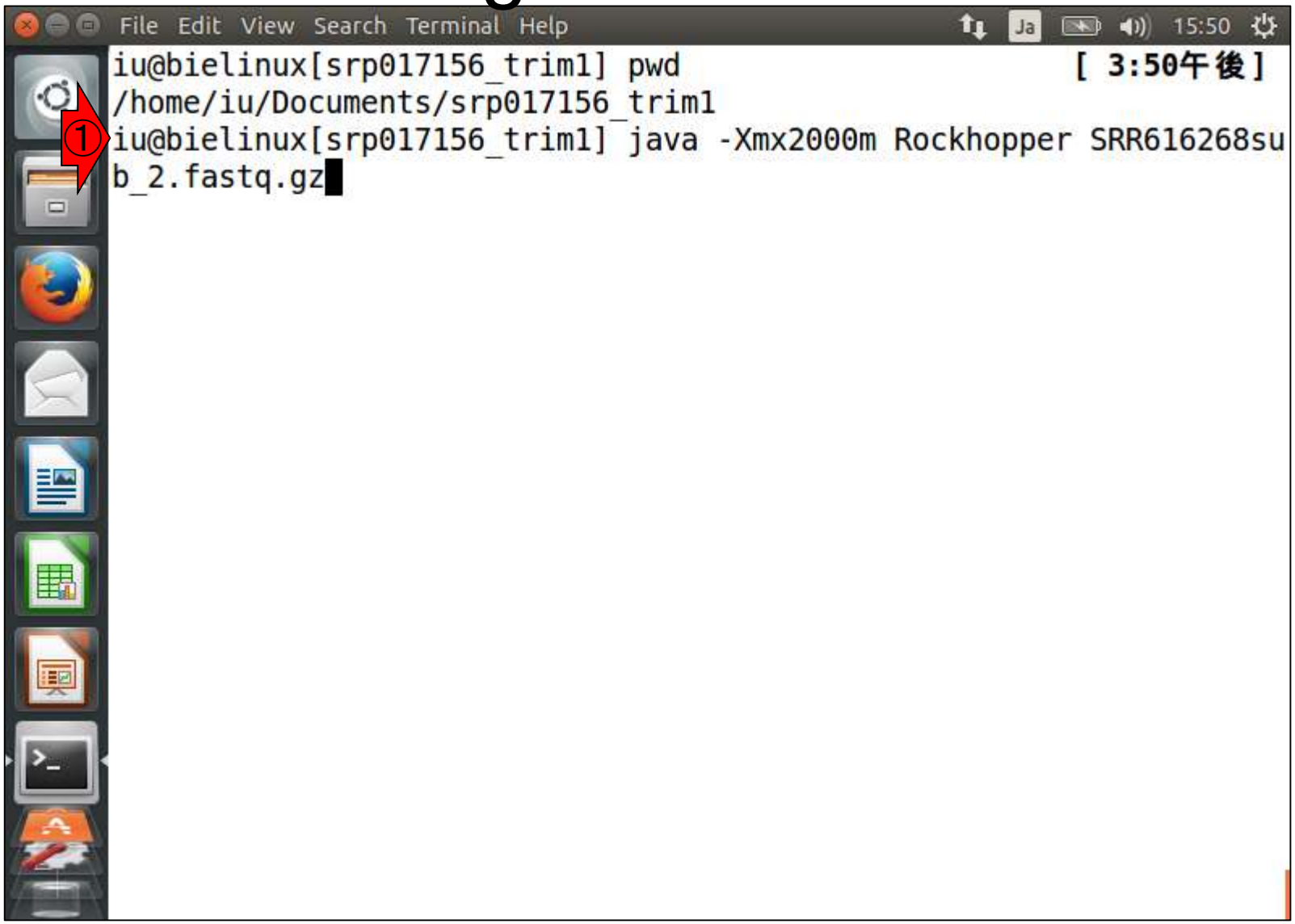
確かに同じ結果になった！2つのトリミングプログラムともに正しく動作していることも、ポジティブなアセンブル結果から証明されたといえる



W17-6：single-end

参考

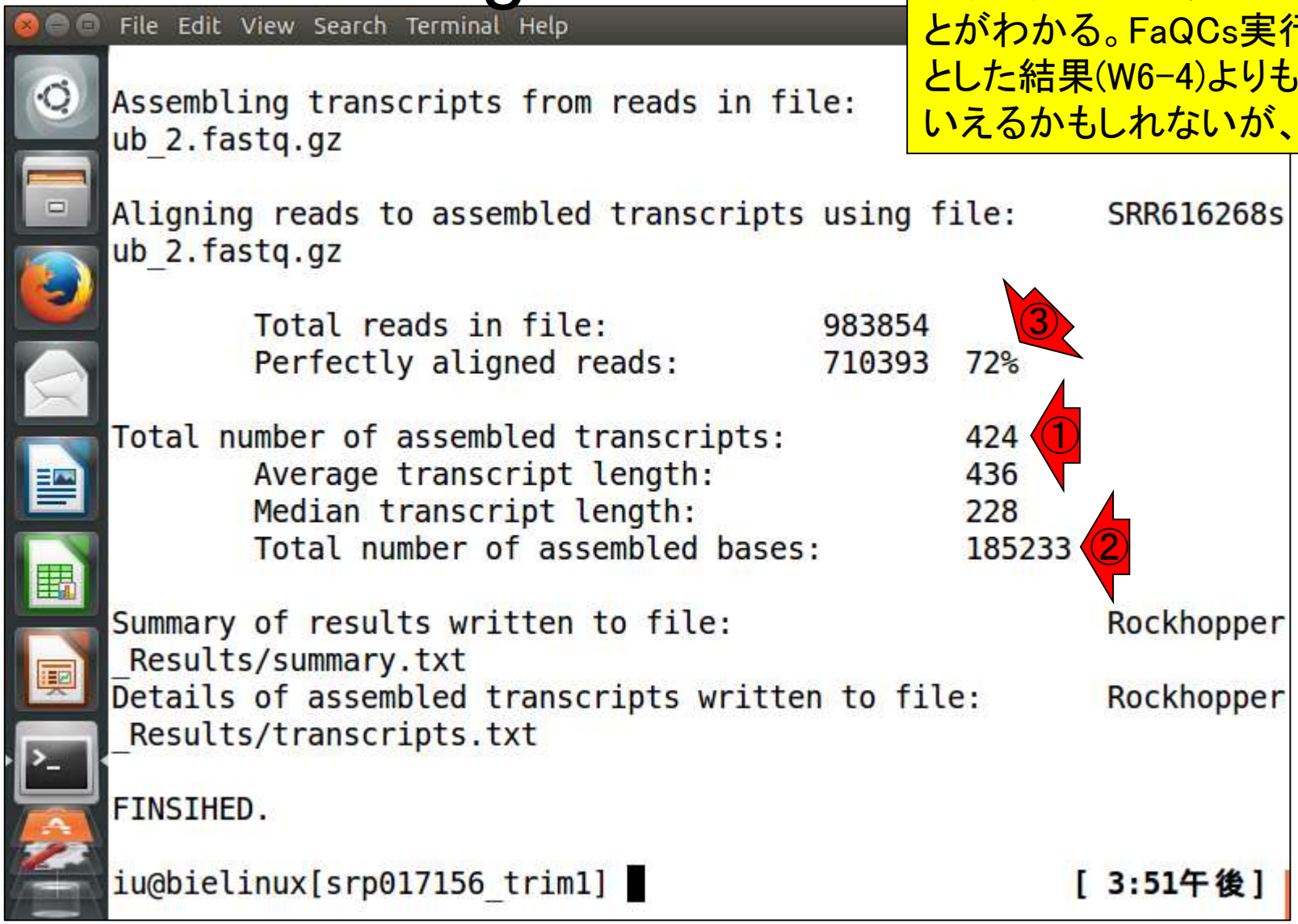
比較用に①何もしていないreverse側のsingle-endのみ(SRR616268sub_2.fastq.gz)で実行



W17-6：single-end

参考

①アセンブルされた転写物数は424個、②総塩基数は185,233。③入力リード数983,854個のうち、72%（710,393個）がマップされていることがわかる。FaQCs実行後のファイルを入力とした結果(W6-4)よりもわずかによい結果といえるかもしれないが、事実上誤差範囲



ここまでのまとめ

①が追加実行分。アセンブリ結果へのインパクトが大きいのは、forward側の100–107 bp部分のトリム（W6-2 → W17-4）

- オリジナル（SRR616268）
  - 乳酸菌paired-end RNA-seqデータで、最初の100万リードのみ抽出
  - forward側(SRR616268sub_1.fastq.gz)のリード長は107 bp
  - reverse側(SRR616268sub_2.fastq.gz)のリード長は93 bp

- FaQCs実行結果(W1-1)
  - 1,000,000リード → 977,202リード (W1-3)
  - forward側(QC.1.trimmed.fastq)
  - reverse側(QC.2.trimmed.fastq)
  - リード長はバラバラ。FastQC上で見られるIllumina adapterは消滅状態
- *de novo*トランスクリプトームアセンブリ(Rockhopper2)実行結果
  - paired-end（QC.1.trimmed.fastqとQC.2.trimmed.fastq）：0 transcript or contig (W5-2)
  - single-end (forward側のみ; QC.1.trimmed.fastq)：1 transcript (W6-2)
  - single-end (reverse側のみ; QC.2.trimmed.fastq)：423 transcripts (W6-4)
  - paired-end (hoge_1.fastq.gzとSRR616268sub_2.fastq.gz)：794 transcripts (W17-4)
  - single-end (reverse側のみ; SRR616268sub_2.fastq.gz)：424 transcripts (W17-6)

①

処理の順番について

今回、トリミングは①を入力として②を作成した。なぜ③のFaQCs実行後のファイルを入力としなかったのか?について思考回路を解説

- オリジナル(SRR616268)
  - 乳酸菌paired-end RNA-seqデータで、最初の100万リードのみ抽出
  - forward側(SRR616268sub_1.fastq.gz)①のリード長は107 bp
  - reverse側(SRR616268sub_2.fastq.gz)のリード長は93 bp

- FaQCs実行結果(W1-1)
  - 1,000,000リード → 977,202リード (W1-3)
  - forward側(QC.1.trimmed.fastq) ③
  - reverse側(QC.2.trimmed.fastq)
  - リード長はバラバラ。FastQC上で見られるIllumina adapterは消滅状態
- *de novo*トランスクリプトームアセンブリ(Rockhopper2)実行結果
  - paired-end(QC.1.trimmed.fastqとQC.2.trimmed.fastq) : 0 transcript or contig (W5-2)
  - single-end (forward側のみ; QC.1.trimmed.fastq): 1 transcript (W6-2)
  - single-end (reverse側のみ; QC.2.trimmed.fastq): 423 transcripts (W6-4)
  - paired-end (hoge_1.fastq.gzと②SRR616268sub_2.fastq.gz): 794 transcripts (W17-4)
  - single-end (reverse側のみ; SRR616268sub_2.fastq.gz): 424 transcripts (W17-6)

処理の順番について

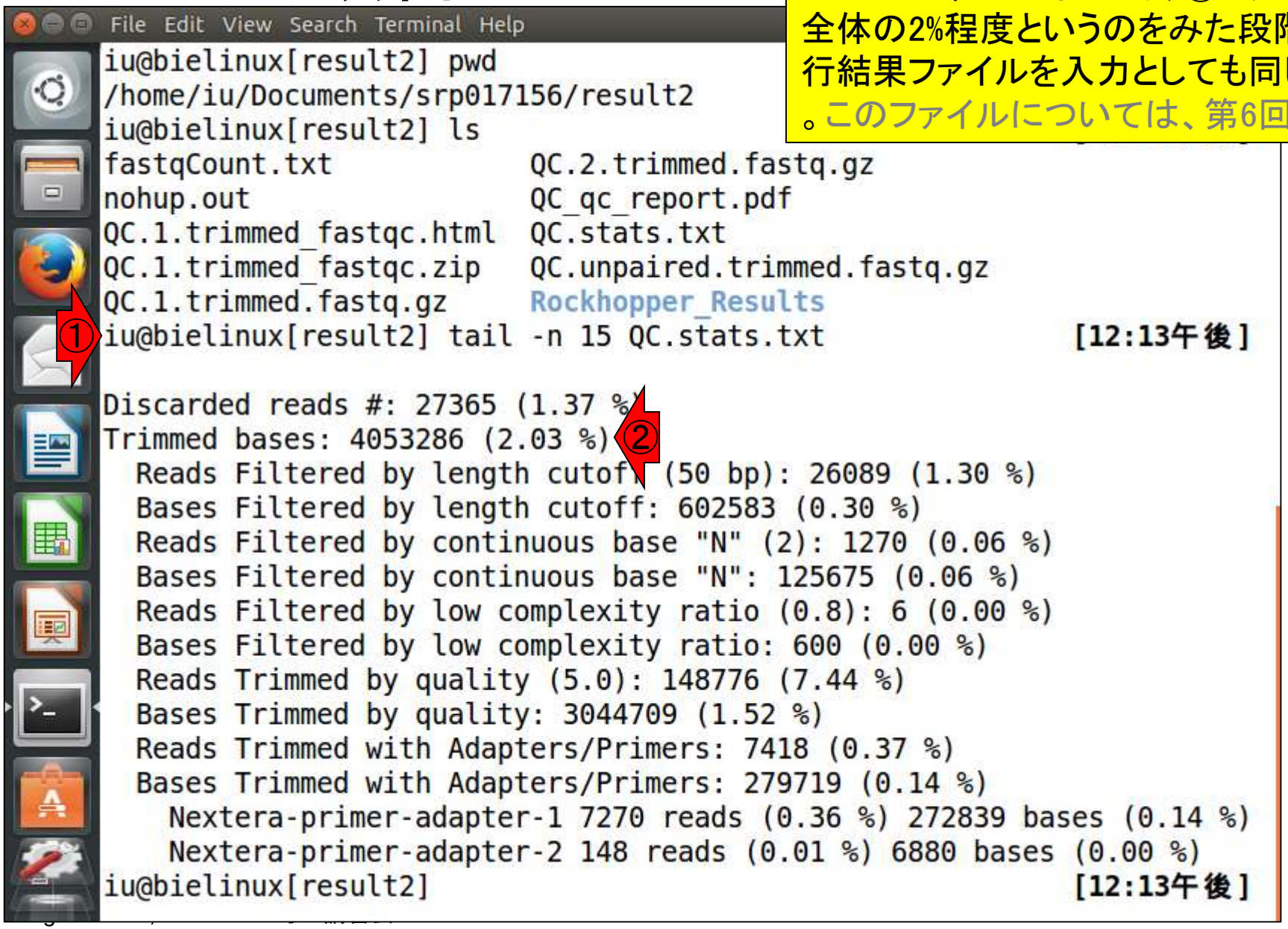
まず、結論としては、③を入力として3’側の8 bpをトリムしてもほとんど同じ結果となるだろうと予想した。理由は、④FaQCs実行で、リード数自体が3%程度しか減少していないから。また、FastQC実行結果（第4回W17-2）で見えていたアダプター配列の出現回数自体も1,296回と全体に与える影響度合い的には誤差範囲

- オリジナル（SRR616268）
  - 乳酸菌paired-end RNA-seqデータで、最初
  - forward側(SRR616268sub_1.fastq.gz) ①リー
  - reverse側(SRR616268sub_2.fastq.gz)のリード長は93 bp

- FaQCs実行結果(W1-1)
  - 1,000,000リード → 977,202リード (W1-3) ④
  - forward側(QC.1.trimmed.fastq) ③
  - reverse側(QC.2.trimmed.fastq)
  - リード長はバラバラ。FastQC上で見られるIllumina adapterは消滅状態
- *de novo*トランスクリプトームアセンブリ(Rockhopper 2)実行結果
  - paired-end（QC.1.trimmed.fastqとQC.2.trimmed.fastq）: 0 transcript or contig (W5-2)
  - single-end (forward側のみ; QC.1.trimmed.fastq): 1 transcript (W6-2)
  - single-end (reverse側のみ; QC.2.trimmed.fastq): 423 transcripts (W6-4)
  - paired-end (hoge_1.fastq.gz ②RR616268sub_2.fastq.gz): 794 transcripts (W17-4)
  - single-end (reverse側のみ; SRR616268sub_2.fastq.gz): 424 transcripts (W17-6)

処理の順番について

また、実は裏で①FaQCsのQC結果のサマリーファイル(QC.stats.txt)もざっと眺めている。細かいことはよくわからなくても、②トリムされた塩基が全体の2%程度というのをみた段階で、「FaQCs実行結果ファイルを入力としても同じだろう」と断定。このファイルについては、第6回W5-3でも解説



W1-3：FastQCで確認

さらに、①第5回W1-3（2016.08.01のスライド25）でFastQC実行結果を眺めているが、この時に②で配列長分布も見ている。つまり…

FaQCs(ver. 1.34)によるQ

①

- 連載第5回[W1-1]結果との比較用に、FaQCs実行前のデータでFastQC (ver. 0.11.3)を実行。
  連載第4回の[W7, W8, W9-7]あたり。

```
fastqc2 -v
fastqc2 -q SRR616268sub_1.fa
fastqc2 -q SRR616268sub_2.fa
```

連載第4回[W9-5]の手順通りに行った
です。実行結果ファイルSRR616268su
結果ですが)連載第4回W8-6に示すよ
と3」がリード中に含まれていることが
「Illumina Single End PCR Primer 1」が

- Skewer: Jiang et al., BMC Bioinformat
- FaQCs: Lo and Chain, BMC Bioinform

- 図1。第4回の[W17-3]と基本的に同じ。

```
cd ~/Documents/srp017156
rm -f hoge*
rm -f JS*
rm -rf result*
rm -f *.bz2

pwd
ls -lh
fastqc2 -v
FaQCs.pl -v
time FaQCs.pl -adapter -p SR
ls result2
```

```
pwd
ls -lh
fastqc2 -v
FaQCs.pl -v
time FaQCs.pl -adapter -p SRR616268sub_1.fastq.gz SRR616268sub_2.fastq.gz -d result2
ls result2
```

FaQCs ver. 1.34を実行し、結果をresult2ディレクトリに保存(約25分)。result2ディレクトリ上には、計6ファイルが生成されている。そのうちのQC結果のサマリーレポートは次の2つ：QC.stats.txtとQC_qc_report.pdf

- トリム後(FaQCs実行後)のデータを入力としてFastQC (ver. 0.11.3)をデフォルトで実行。[W1-2]

```
pwd
ls result2/*.fastq
fastqc2 -q result2/QC.1.trimmed.fastq --outdir=/home/iu/Desktop/mac_share
fastqc2 -q result2/QC.2.trimmed.fastq --outdir=/home/iu/Desktop/mac_share
date
ls -lh /home/iu/Desktop/mac_share/QC.*
```

②

出力結果を共有フォルダ(/home/iu/Desktop/mac_share)に直接保存。実行結果ファイル
QC.1.trimmed_fastqc.html(forward側)とQC.2.trimmed_fastqc.html(reverse側)ともに、Overrepresented sequences項目に見えていたアダプターやプライマー配列情報がなくなっているのがわかります。[W1-3]

①

- Rockhopper 2: Tjaden B, Genome Biol., 2015
- QuasR: Gaidatzis et al., Bioinformatics, 2015

トランスクリプトームアセンブリ

処理の順番について

①Basic Statisticsの、②Sequence lengthの50-107 bpという配列長の範囲情報や…

FastQC Report

Sun 20 Dec 2015
QC.1.trimmed.fastq

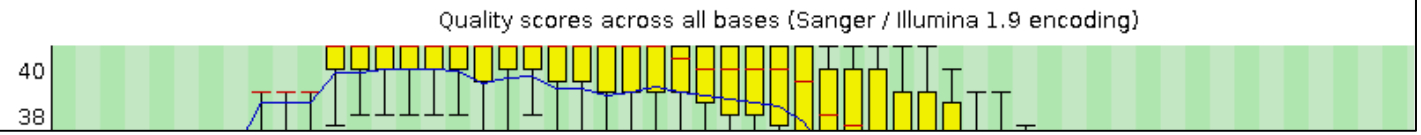
Summary

- Basic Statistics ①
- Per base sequence quality
- Per tile 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	QC.1.trimmed.fastq
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	977202
Sequences flagged as poor quality	0
Sequence length	50-107 ②
%GC	50

Per base sequence quality



処理の順番について

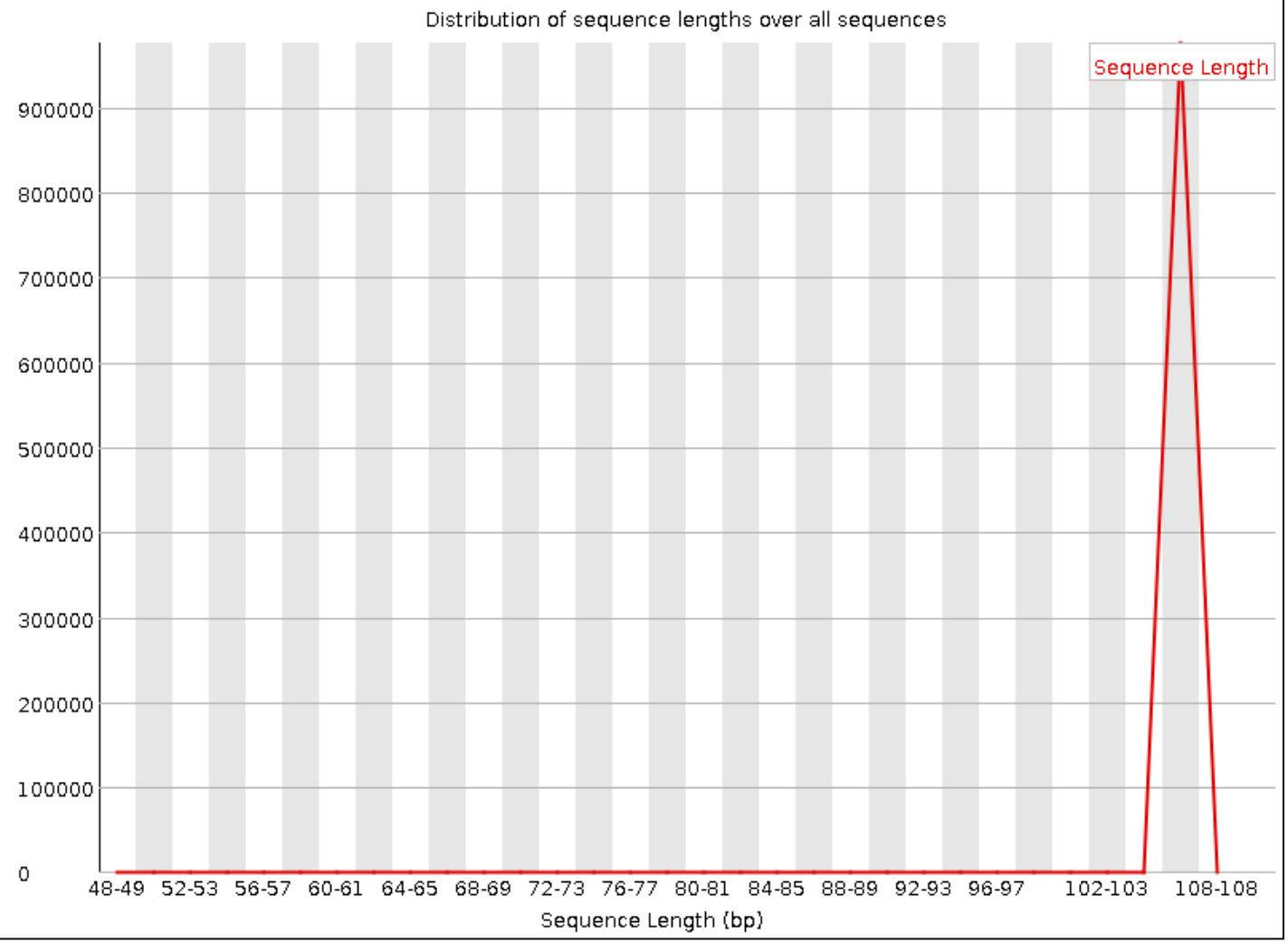
①Sequence Length Distributionのところを眺めて、リード数としては少ないが、最短で50 bpになっているリードもあることを認識する。そしてこれらに対して無条件で8 bpトリムするのは、心情的に忍びない、という思考回路。8 bpトリム後にFaQCsはアリ。むしろ推奨

FastQC Report

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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

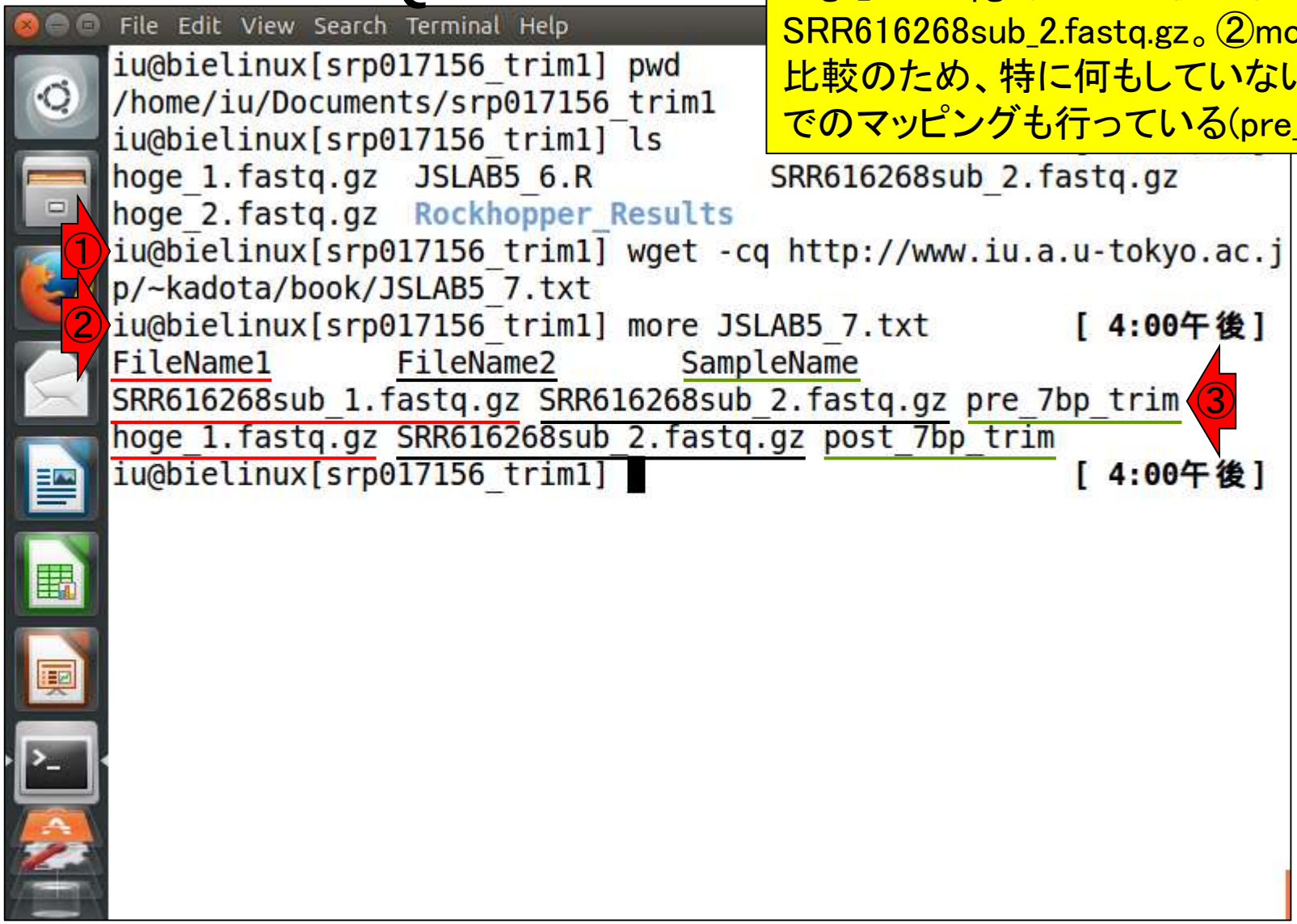


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

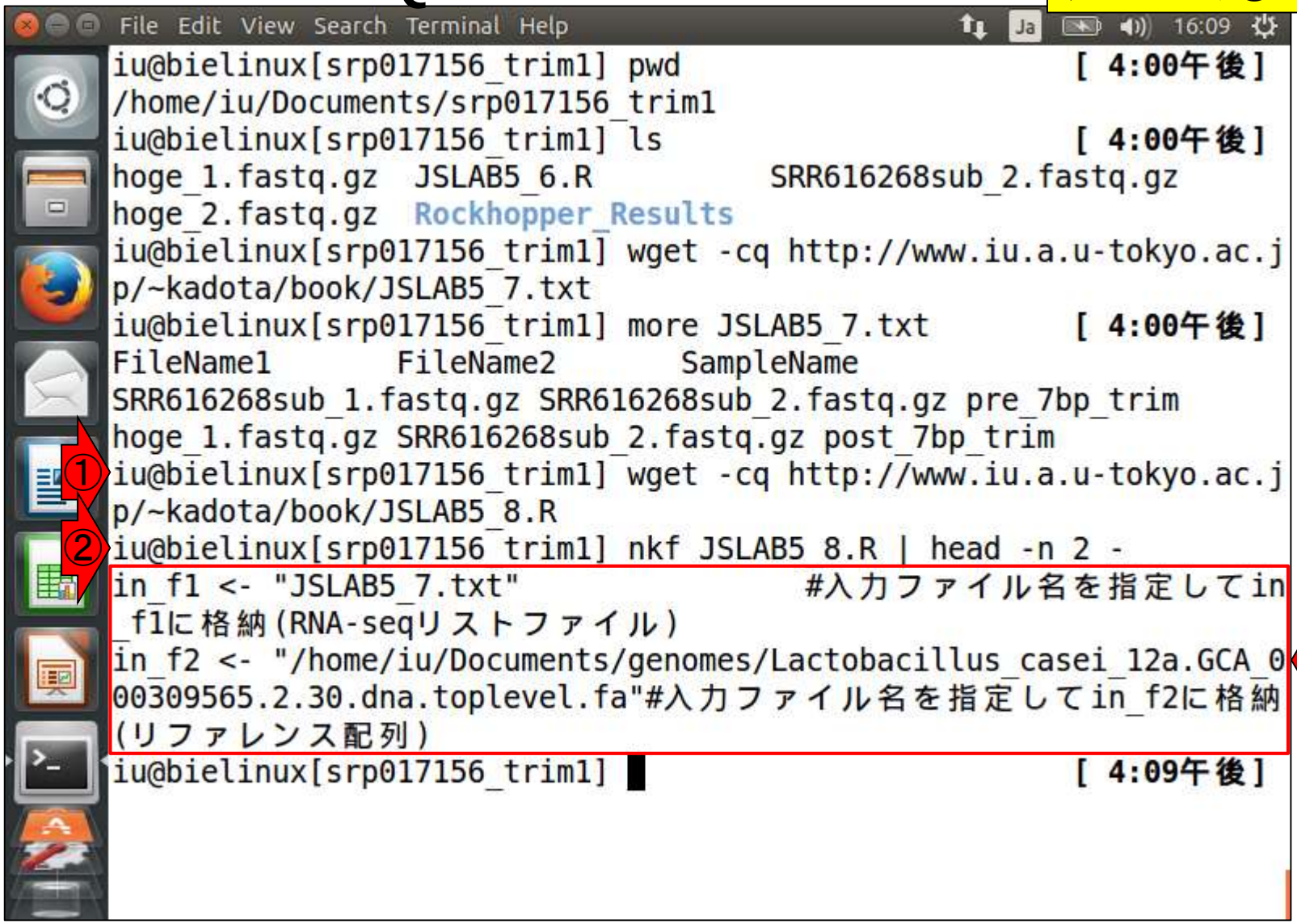
W18-1：QuasR

①トリム後のデータでマッピングを再実行すべく、リストファイル(JSLAB5_7.txt)をダウンロード。forward側はhoge_1.fastq.gz、reverse側は特に何もしていないSRR616268sub_2.fastq.gz。②moreで中身を表示。③比較のため、特に何もしていないforward側のファイルでのマッピングも行っている(pre_7bp_trimの行に相当)



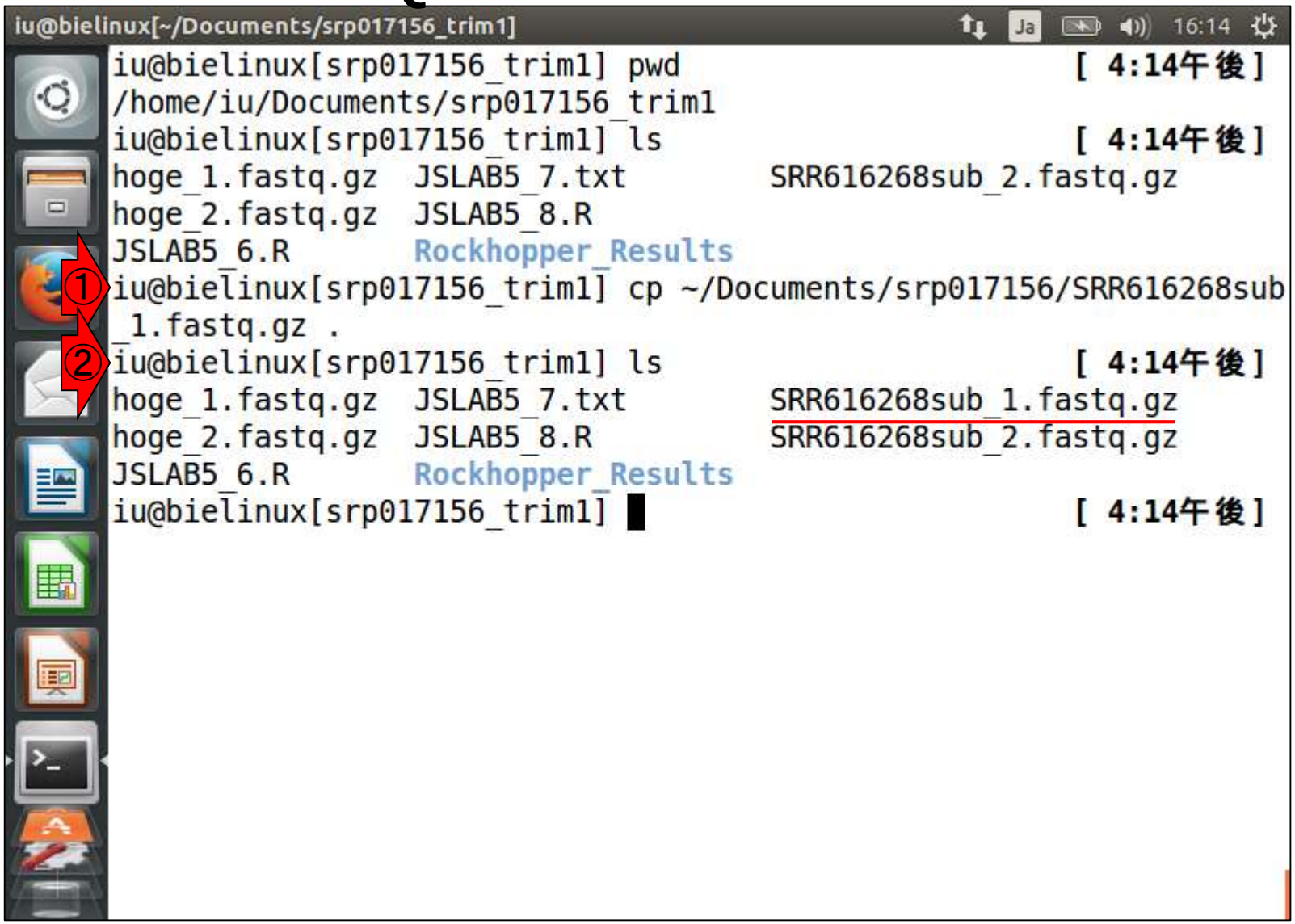
W18-2：QuasR

①Rスクリプトファイル(JSLAB5_8.R)をダウンロードし、②最初の2行分を表示させ、③入出力ファイルを確認



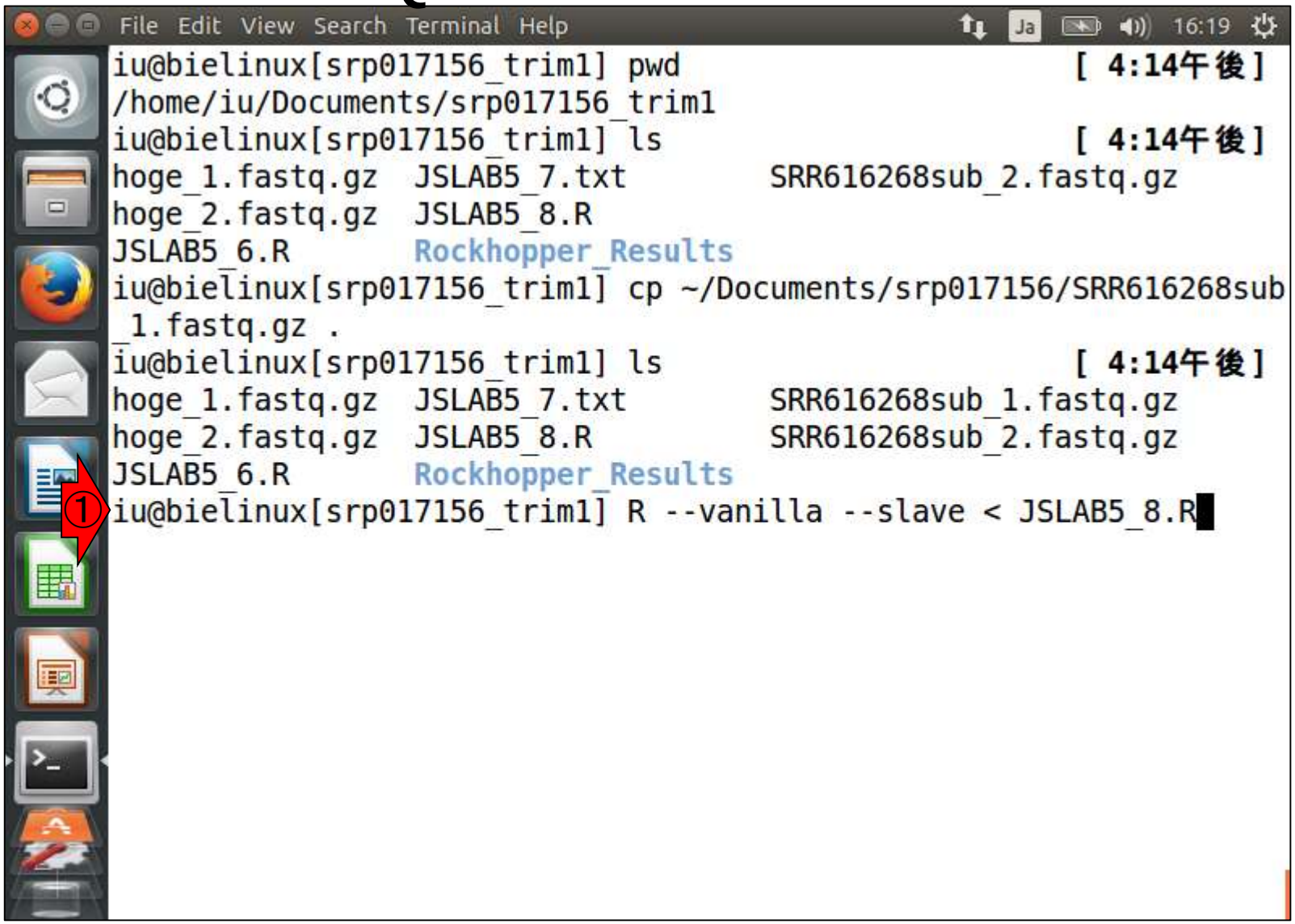
W18-3：QuasR

①比較用入力ファイル
(SRR616268sub_1.fastq.gz)のコピーと②確認



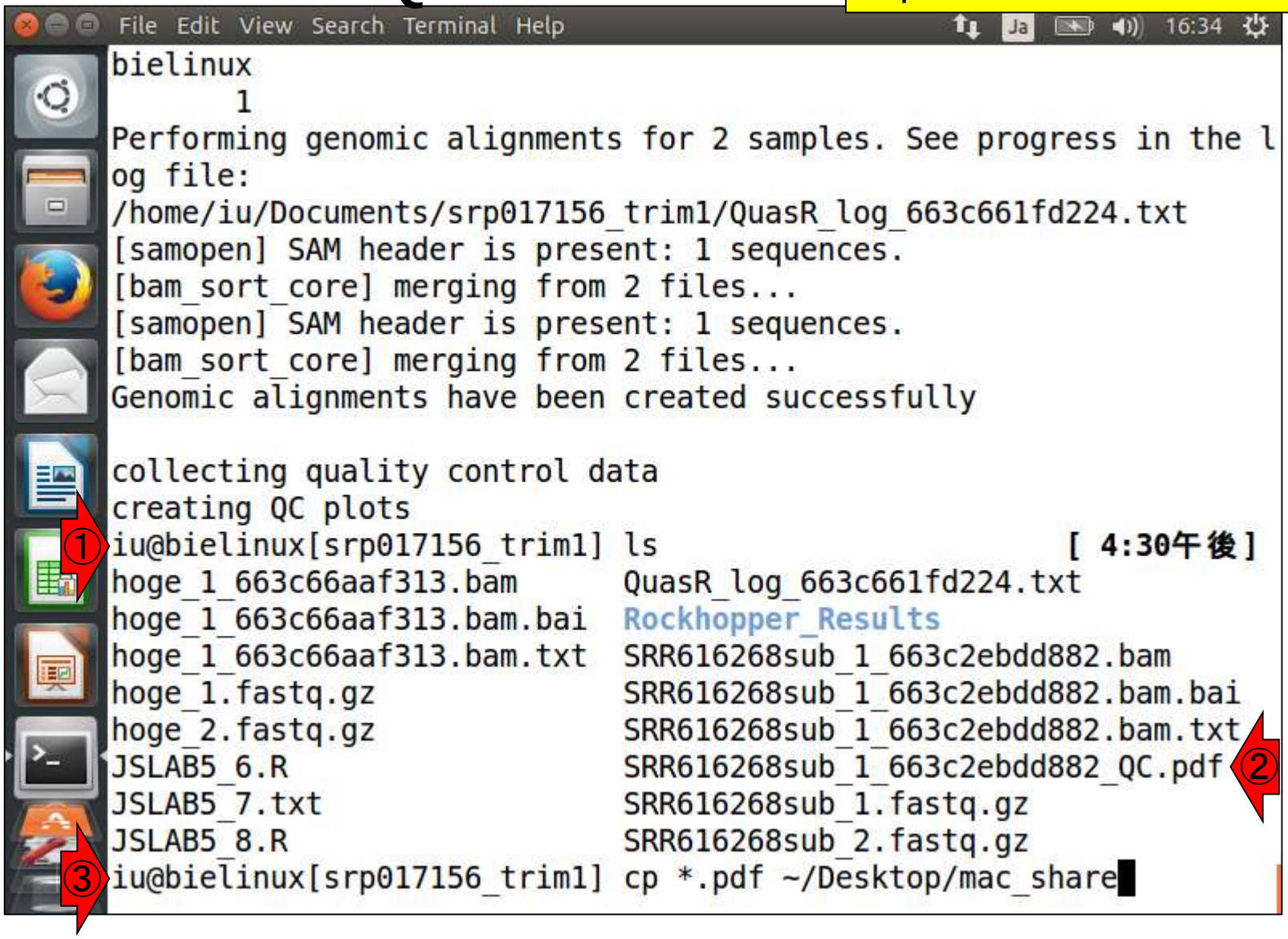
W18-4：QuasR

①Rスクリプトファイル
(JSLAB5_8.R)の実行。約13分



W18-4：QuasR

無事終了。①lsで確認。bamファイルや②QCレポートファイルが作成されていることがわかる。③pdfファイルを共有フォルダにコピーして眺める



W18-5：PDFはここにもあり

何らかの原因でファイルが作成されなかった場合は、①こちらをご覧になってください

- QuasR[W18-4]
  Rスクリプトファイル(JSLAB5_8.R)の実行。

```
R --vanilla --slave < JSLAB5_8.R
ls
cp *.pdf ~/Desktop/mac_share
```

- 得られたQCレポートPDFファイル(SRR616268sub_1_663c2ebdd882_QC.pdf)の解説[W18-5]

①

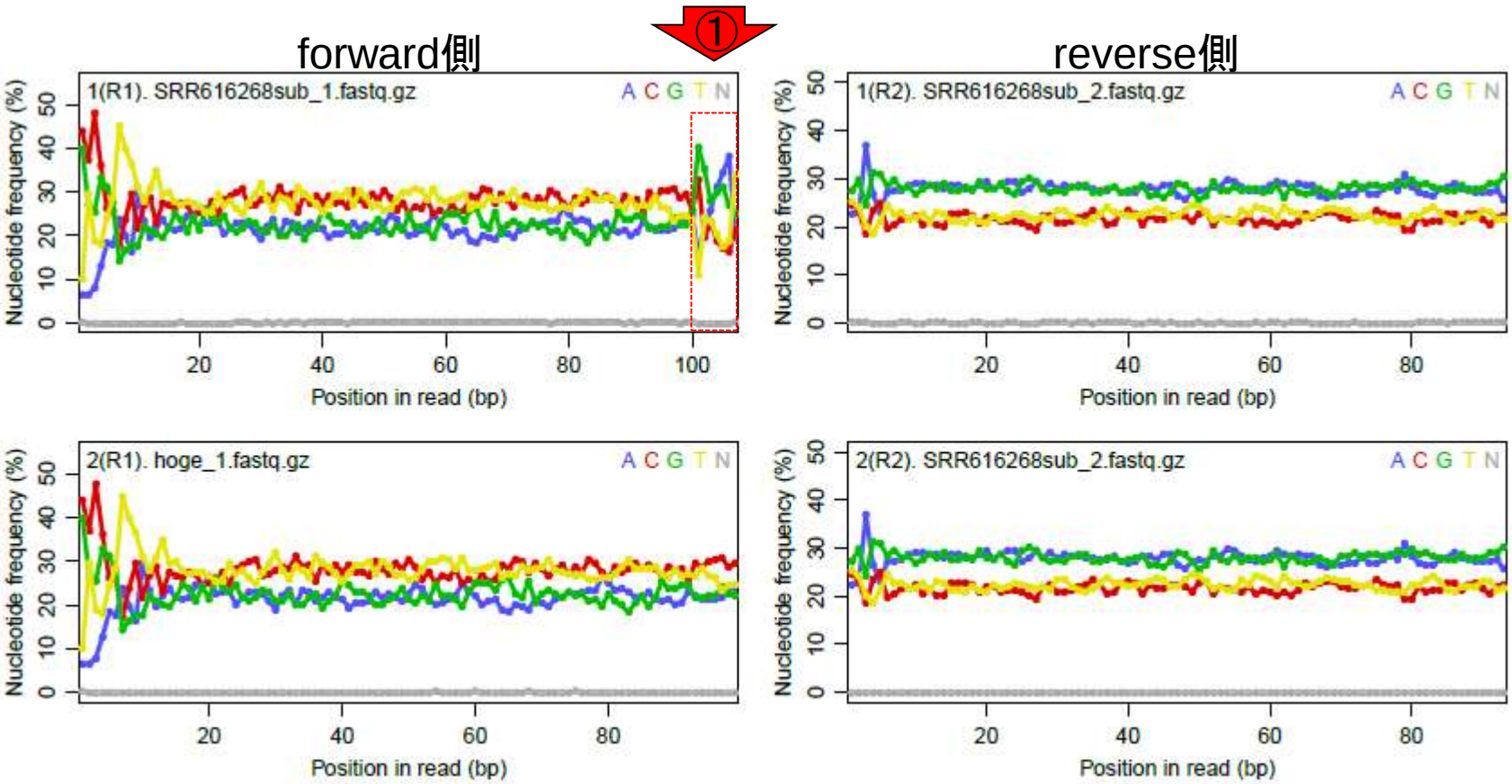
おわりに

- FastQC[W19-1]
  トリム前(FaQCs実行前)のforward側のリードを入力として--nogroupオプションをつけてFastQC (ver. 0.11.4)をデフォルトで実行。-qオプションは、途中経過を表示させないだけのオプションなので、結果自体には無関係。

```
pwd
ls SRR616268sub_1*
fastqc2 -v
fastqc2 -q --nogroup SRR616268sub_1.fastq.gz
ls SRR616268sub_1*
mv SRR616268sub_1_fastqc.html SRR616268sub_1_nogroup.html
cp SRR616268sub_1_nogroup.html ~/Desktop/mac_share
```

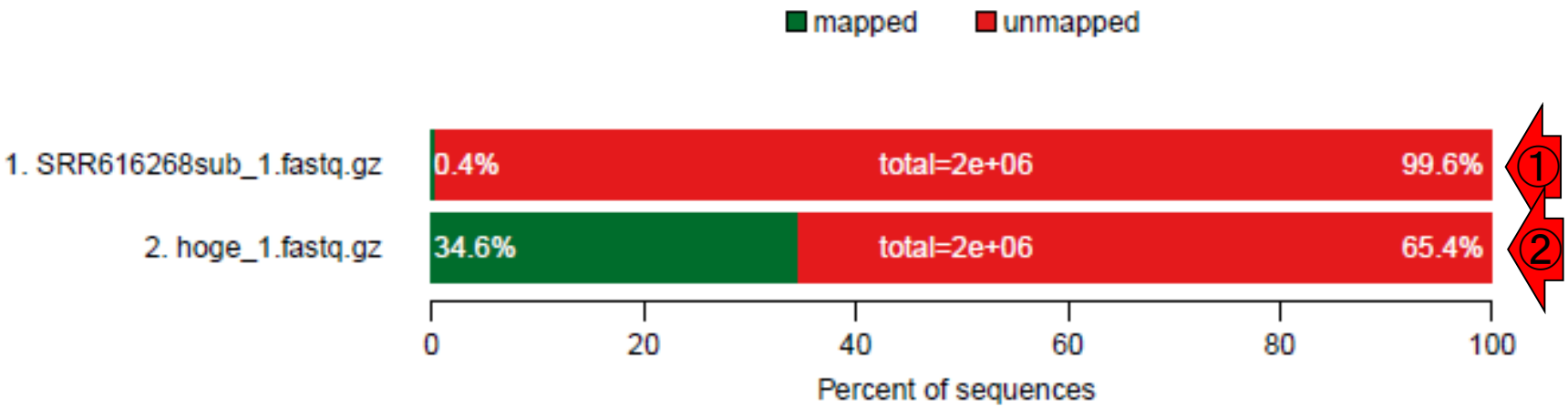
W18-5：PDF解説

PDF2枚目。ポジションごとの塩基の出現確率。FastQC Report中の項目「Per base sequence content」と同じ(但し色は異なる)。赤枠部分をトリムしたおかげで、アセンブルやマッピングが劇的に改善したことになる



W18-6：PDF解説

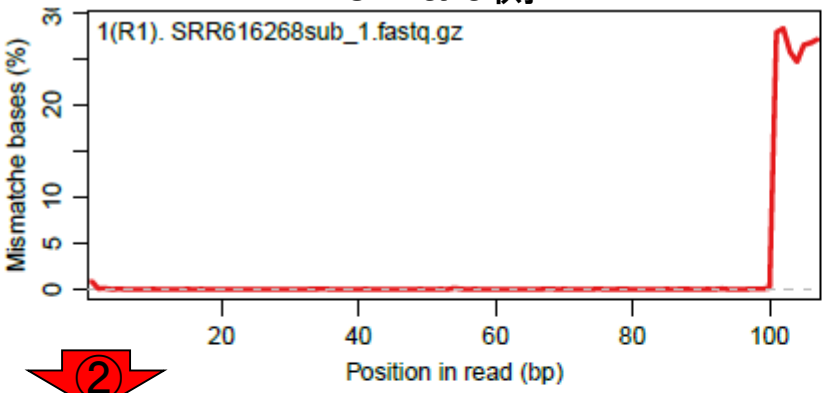
PDF4枚目。全リード(forward, reverse合わせて200万リード)のうち、マップされたリードの割合は①トリム実行前が0.4%、②実行後が34.6%。トリム後のマップ率が劇的に向上!



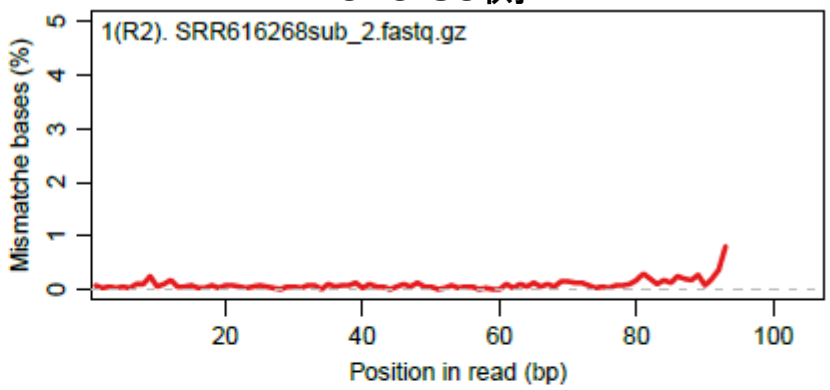
W18-7：PDF解説

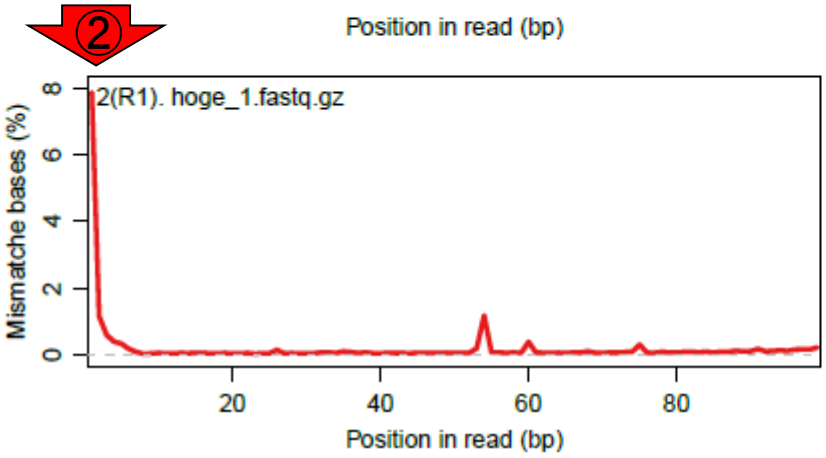
PDF6枚目。forward側の100–107 bpをトリムしたおかげで①のミスマッチ塩基の割合が劇的に低下していることがわかる。そのおかげで、相対的なインパクトが弱かった②forward側の1塩基目あたりもミスマッチ率が高かったことがわかる。
ここもトリムしておけばいいかも…と妄想する

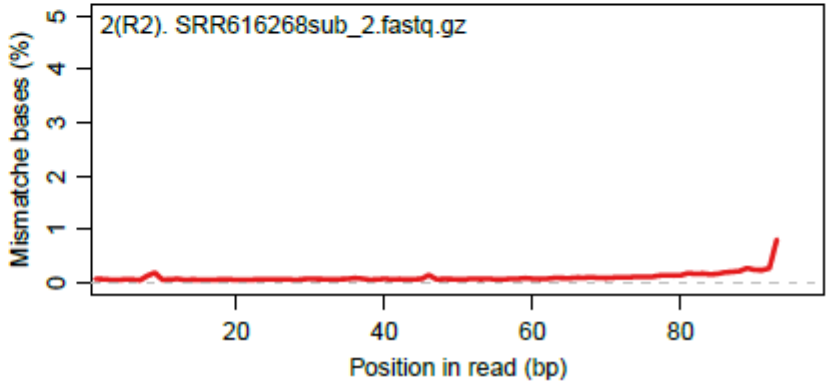
forward側



reverse側





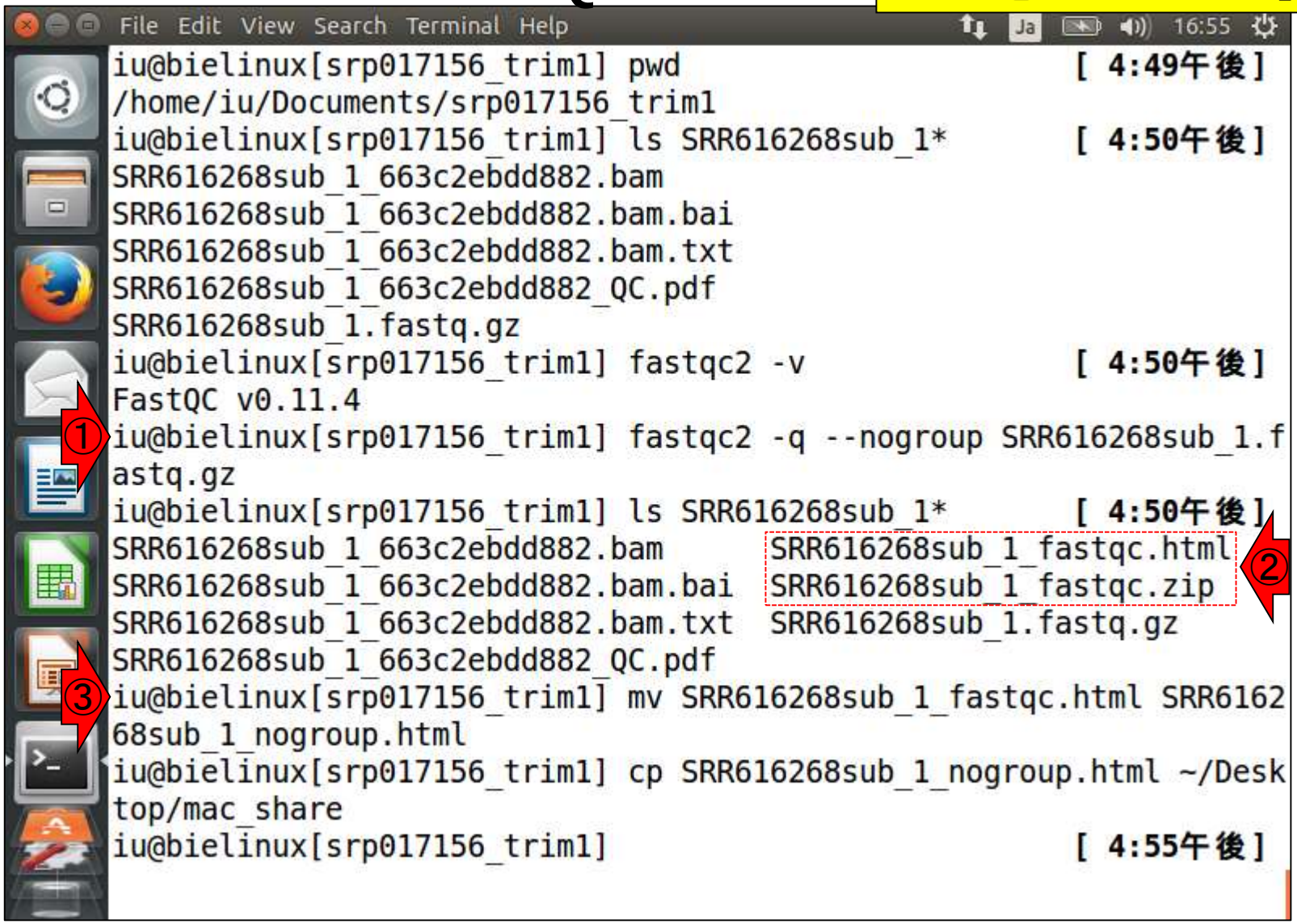


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

W19-1：FastQC

①--nogroupオプションをつけてFastQCを実行。②赤枠が出力ファイル。③htmlレポートのファイル名をSRR616268sub_1_nogroup.htmlに変更



W19-2：FastQC

①SRR616268sub_1_nogroup.htmlのKmer Content項目を表示。1-59塩基目には極端に多いk-merの上位6個は存在しないことがわかる

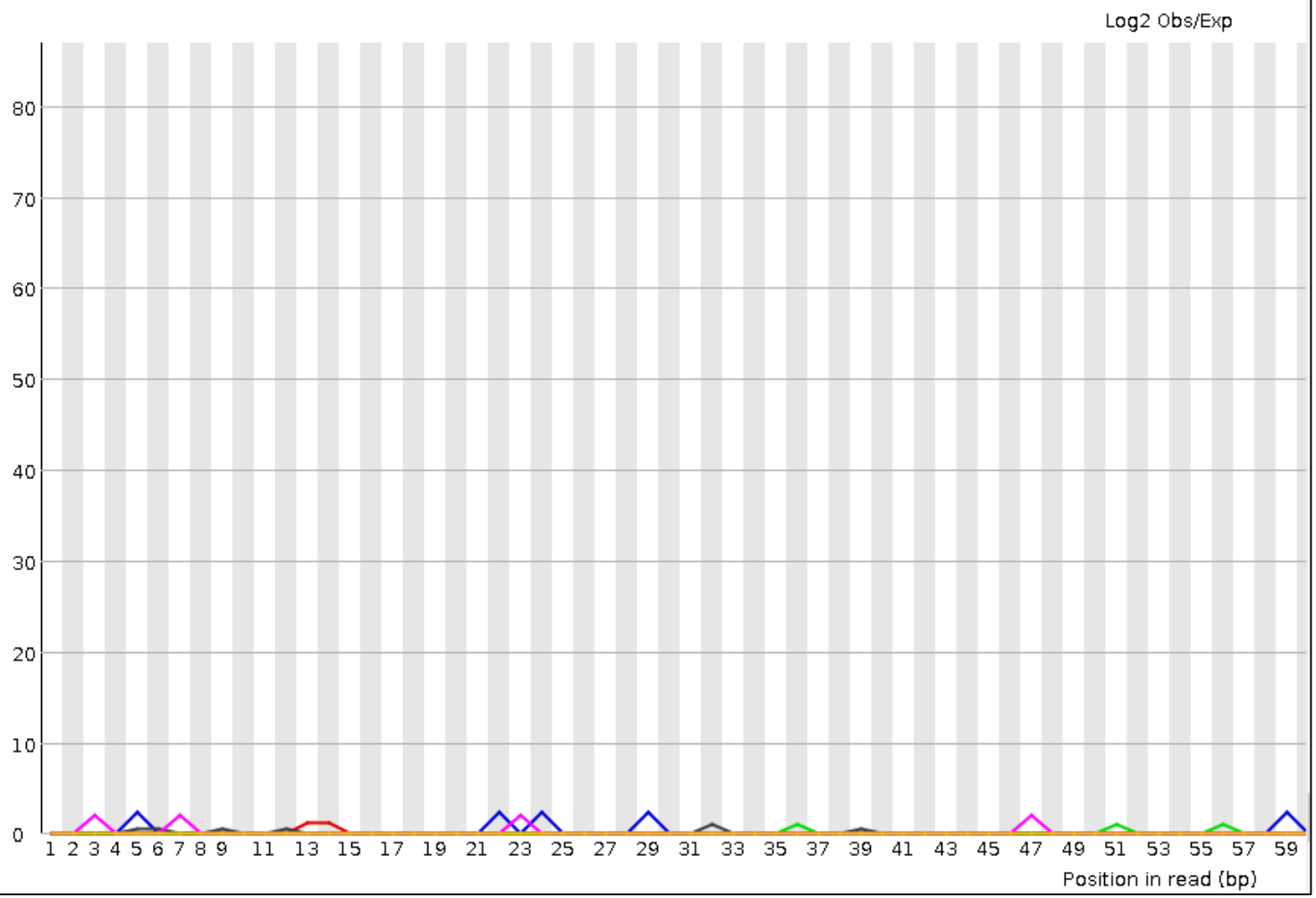
FastQC Report

Wed 16 Sep 2015
SRR616268sub_1.fastq.gz

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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 ①

Kmer Content



W19-2：FastQC

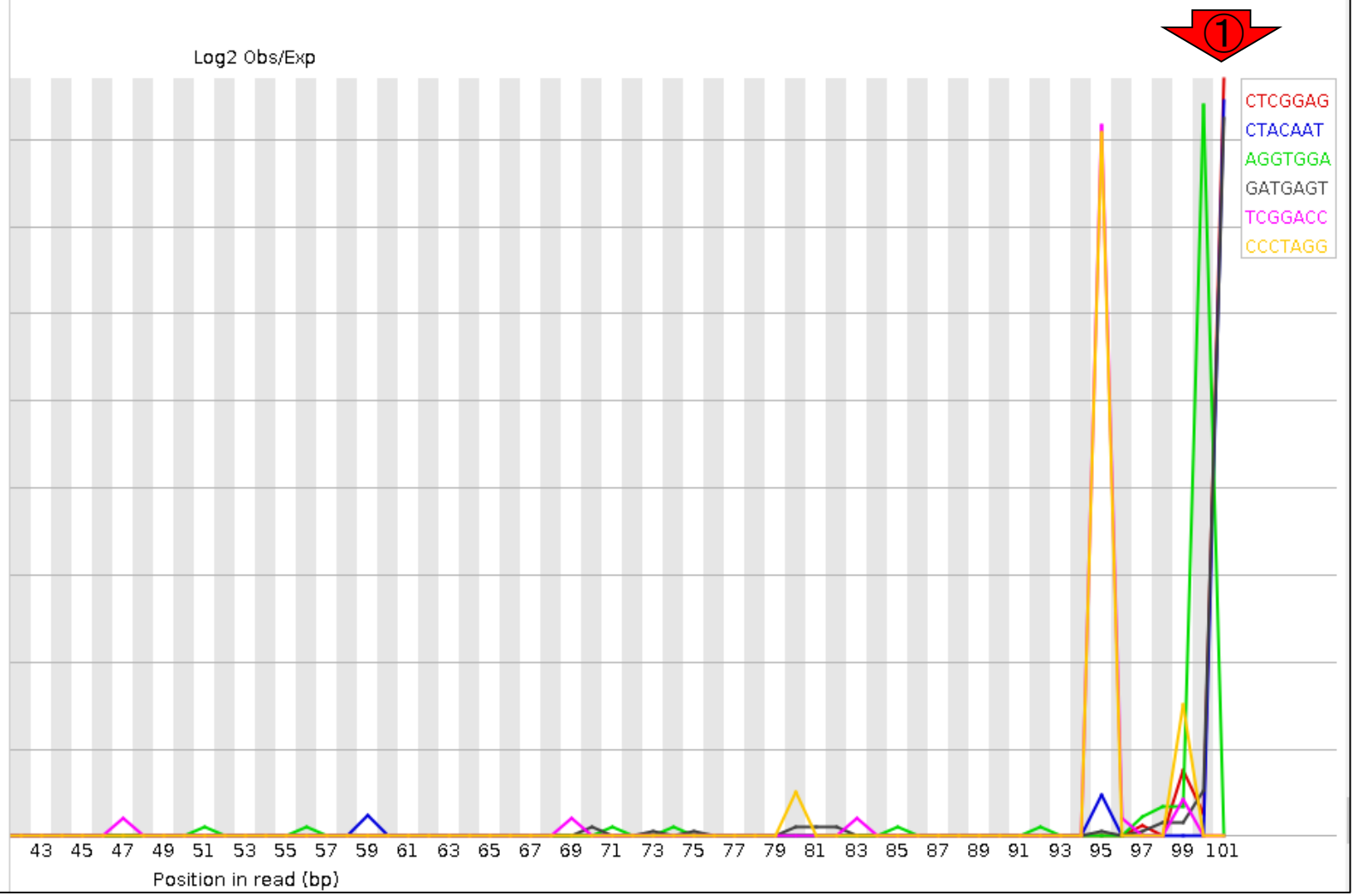
①リードの右側(3’側)を表示。極端に多いk-merの上位6個が右側に偏って存在することがわかる

FastQC Report

Wed 16 Sep 2015
SRR616268sub_1.fastq.gz

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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



W19-2：FastQC

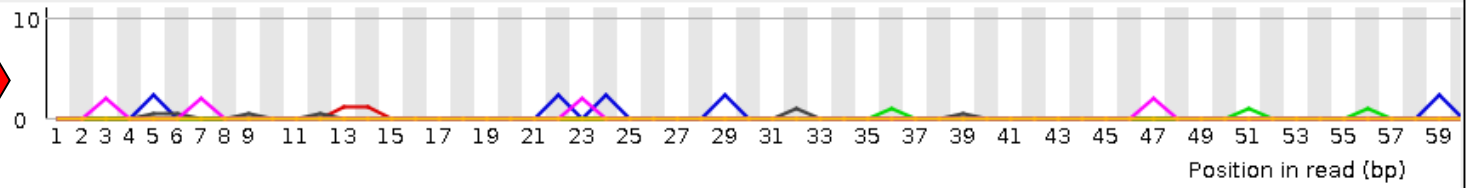
① Kmer Content項目のちょっと下のほうを表示。②上の折れ線グラフは、③赤枠で示す観測値/期待値が大きい上位6個をプロットしたもの

FastQC Report

Wed 16 Sep 2015
SRR616268sub_1.fastq.gz

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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 ①

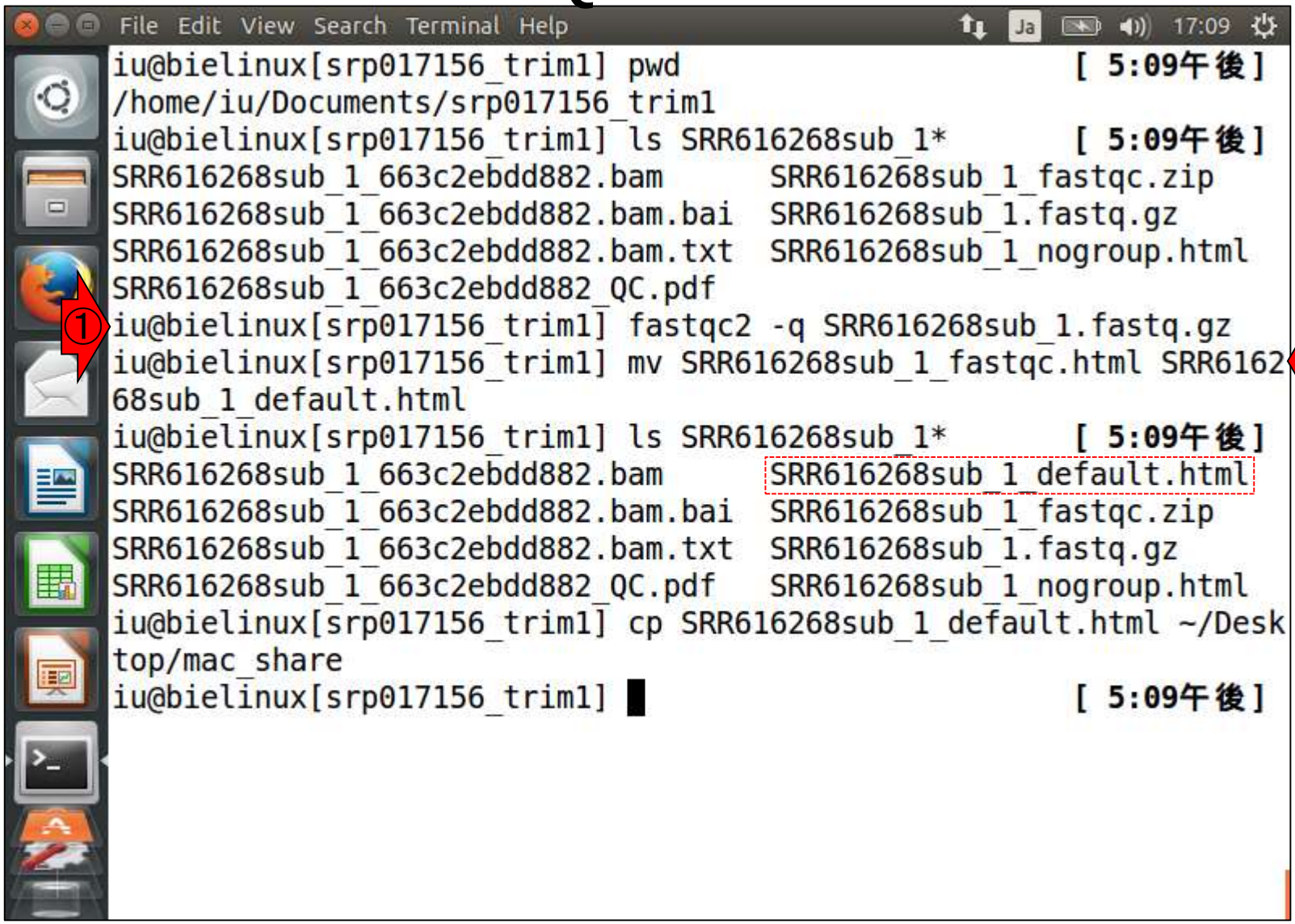
② 

Sequence	Count	PValue	Obs/Exp Max	Max Obs/Exp Position
CTCGGAG	395	0.0	86.890045	101
CTACAAT	215	0.0	84.51276	101
AGGTGGA	445	0.0	83.98286	100
GATGAGT	960	0.0	82.54421	101
TCGGACC	235	0.0	81.66475	95
CCCTAGG	100	0.0	80.80512	95
CCTAGGT	45	1.4897523E-9	78.56054	96
GTTGTGG	965	0.0	75.840096	101
GGCCCTG	180	0.0	75.75481	100
TACTACA	20	0.0016262057	75.75481	96
TAGGTGG	225	0.0	74.07136	99
GTTCTCT	75	0.0	74.02692	101
CGGGCCT	235	0.0	71.00822	1
TCCCTCG	50	3.430614E-9	70.70448	98
CCCACAC	440	0.0	70.0158	95
TACTTAC	405	0.0	68.5846	95

③

W19-3：FastQC

①デフォルトでFastQCを実行。②htmlレポートのファイル名をSRR616268sub_1_default.htmlに変更



W19-4：FastQC

①SRR616268sub_1_default.htmlのKmer Content項目を表示。極端に多いk-merの上位6個が左側（5’側）に偏って存在していることがわかる

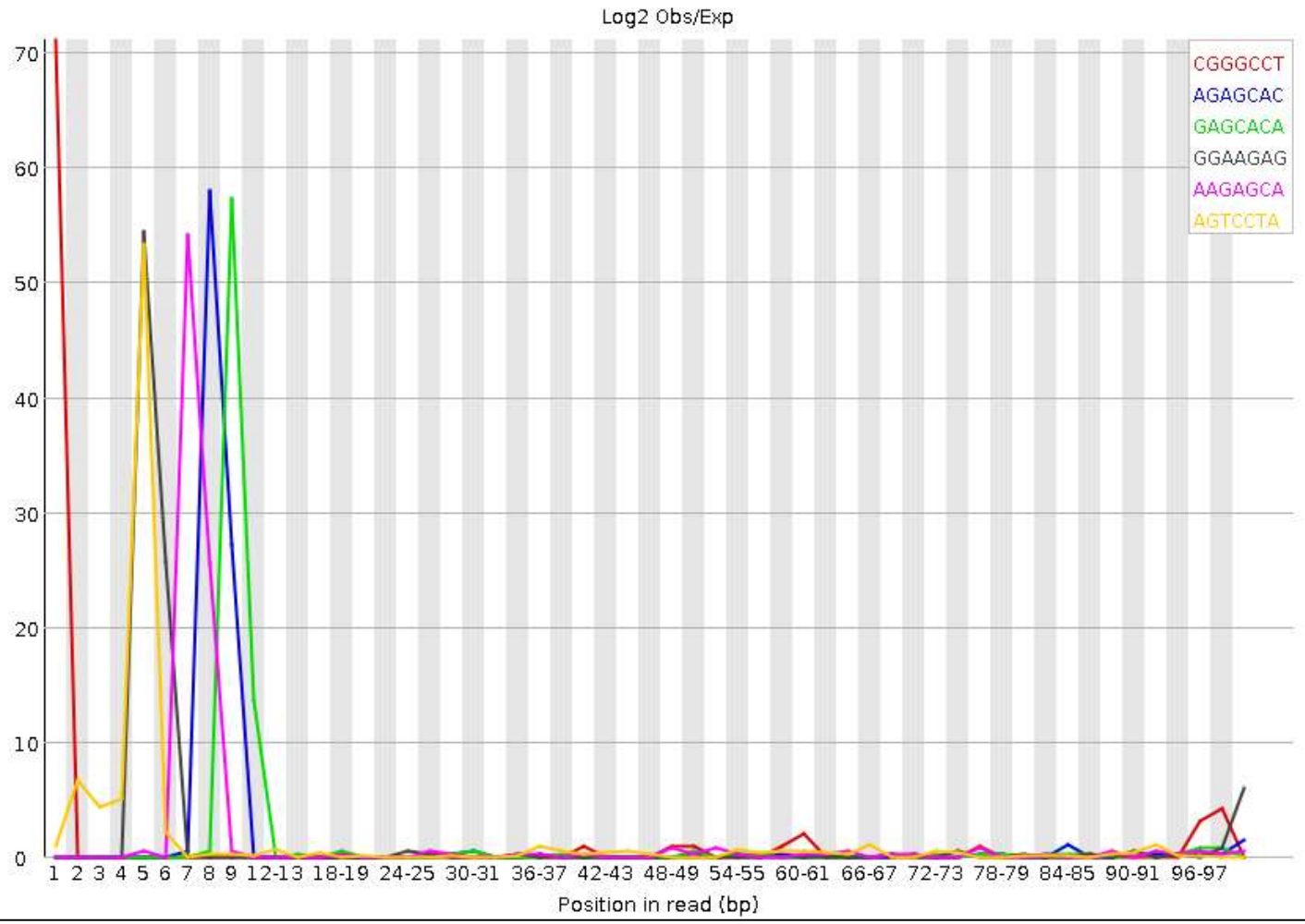
FastQC Report

Wed 16 Sep 2015
SRR616268sub_1.fastq.gz

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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 ①

Kmer Content



W19-4：FastQC

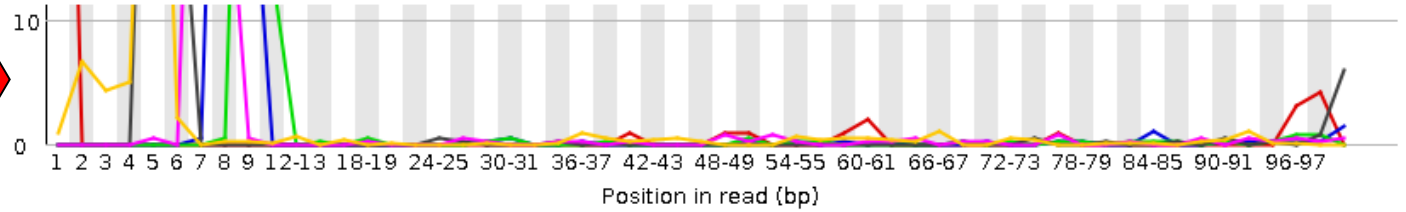
① Kmer Content項目のちょっと下のほうを表示。②上の折れ線グラフは、③赤枠で示す観測値/期待値が大きい上位6個をプロットしたもの

FastQC Report

Wed 16 Sep 2015
SRR616268sub_1.fastq.gz

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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 ①

② 

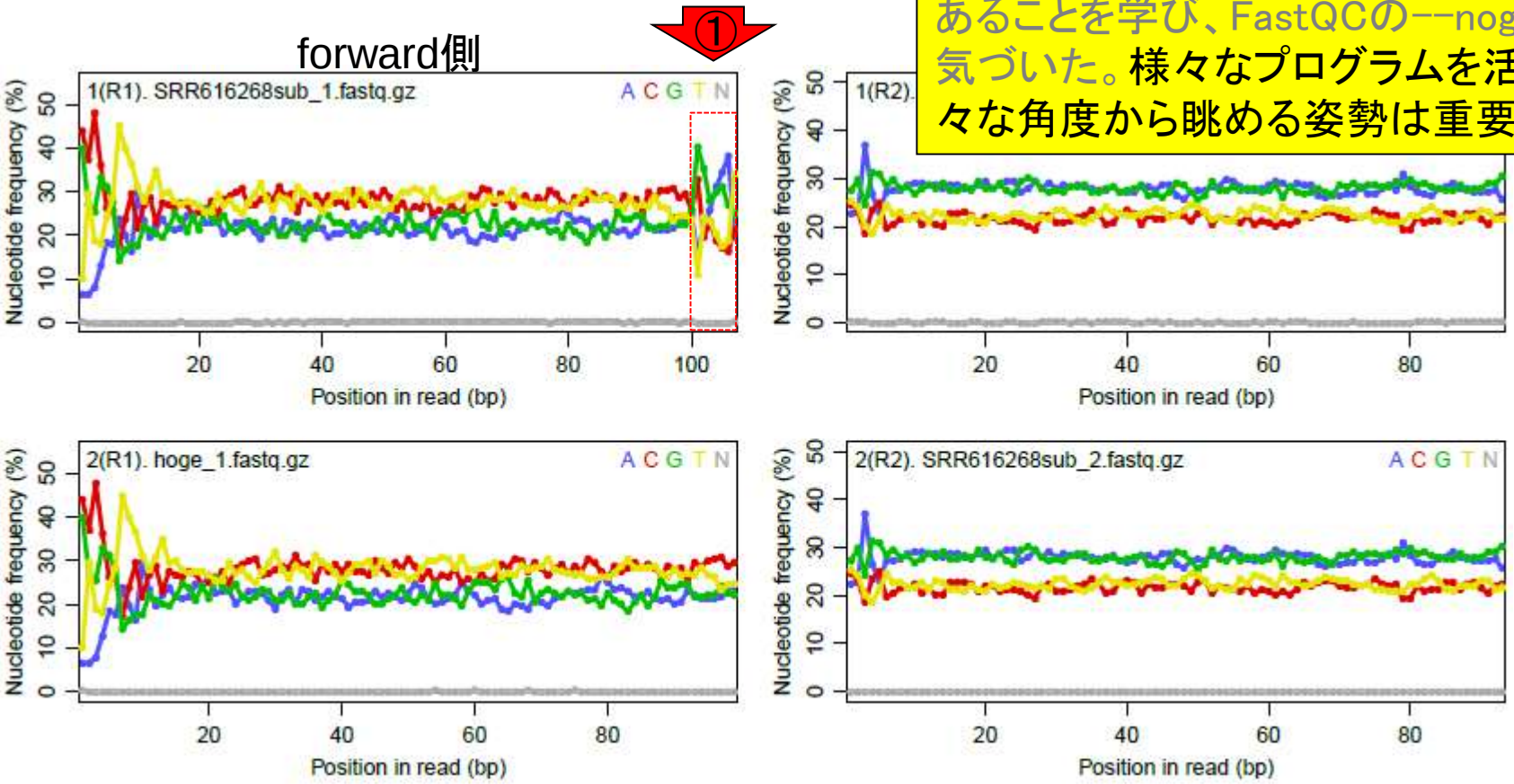
Sequence	Count	PValue	Obs/Exp Max	Max Obs/Exp Position
CGGGCCT	235	0.0	71.00822	1
AGAGCAC	835	0.0	58.01711	8
GAGCACA	845	0.0	57.330517	9
GGAAGAG	880	0.0	54.476883	5
AAGAGCA	885	0.0	54.169106	7
AGTCCTA	1570	0.0	53.35557	5
GTCCAGT	1555	0.0	53.00536	1
CCAGTCC	1655	0.0	51.225075	3
CAGTCCT	1650	0.0	51.07447	4
CCTCTAT	30	0.008129199	50.4628	2
GAAGAGC	975	0.0	49.68645	6
GTCCTAC	1720	0.0	49.289246	6
TCCTACA	1760	0.0	48.169033	7
TCGGAAG	1005	0.0	47.701153	3
CCCCCCT	160	0.0	47.40606	1

③

遺言（独り言）

FastQC実行時に「--nogroup」オプションをつけた場合(W19-2)と、つけないデフォルトの場合(W19-4)でKmer_Content項目の結果が変わるのはいかがなものか…(個人の感想です)。門田は①QuasR実行結果(PDFファイル)を眺めることで、アダプター除去プログラム(FaQCs)でもとりきれないものがあることを学び、FastQCの--nogroupオプションに気づいた。様々なプログラムを活用し、結果を様々な角度から眺める姿勢は重要ではないだろうか

forward側



第5回原稿PDFのp200

第5回は、乳酸菌RNA-seqデータ解析の話。
①第6回は、乳酸菌ゲノムデータの *de novo* ゲノムアセンブリの話。第5回(*L. casei* 12A)と第6回(②*L. hokkaidonensis* LOOC260^{T})では、乳酸菌株が異なる点に注意!

もしれない。

話の展開上本文中では省略したが、結論としては $f_{100\text{-}107}$ 問題に QC 段階で気づくことはできる［W15-5］。具体的には、--nogroup オプションをつけて FastQC を実行した結果を眺めればよい。特に Kmer Contents の項目は、ゲノムアセンブリのところでも述べた *k*-mer（ver. 0.11.3 のデフォルトは *k*=7）の出現頻度をリードのポジションごとに調べ、出現頻度の期待値に比べて実測値が極端に多い上位の *k*-mer とその位置をリストアップしたものである。また、--nogroup は「長いリードの場合に 10 番目以降のポジションを一定幅でグループ化する（デフォルト）」機能をオフにするオプションである［W19-1］。著者らは、--nogroup オプションの有無によって Kmer Contents 項目の結果までが異なることを最近まで知らなかった。つまり、--nogroup オプションをつけずにデフォルトで実行し

た Fa
conte
けなかったのである［W19-4］。第 6 回は、アセンブルプログラム Velvet をオプションつきでインストールすることで指定可能な数値範囲を変更できること、複数の異なる *k*-mer で実行した乳酸菌ゲノムアセンブル結果の違いなどを紹介する予定である。

①

謝　辞

本連載の一部は、国立研究開発法人科学技術振興機構バイオサイエンスデータベースセンター（NBDC）との共同研究の成果によるものです。乳酸菌 *Lactobacillus hokkaidonensis* LOOC260^{T} ゲノム配列決定部分については、原著論文著者（遠野雅徳氏、谷澤靖洋氏、神沼英里氏、中村保一氏、有田正規氏）より詳細情報をいただきました。

②

Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

ここからは…

日本乳酸菌学会誌の①連載第6回ゲノムアセンブリの話。公共DBやデータ取得関連の話は省略するので、必要に応じて②オリジナルのウェブ資料で自習してください

(Rで)塩基配列解析

~NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデ
(last modified 2016/06/03, since 2011)

What's

- このウ
  リーソ
  法(W

- 書籍 | 日本乳酸菌学会誌 | について (last modified 2016/05/12) NEW
- 書籍 | 日本乳酸菌学会誌 | 第1回イントロダクション (last modified 2015/09/11)
- 書籍 | 日本乳酸菌学会誌 | 第2回GUI環境からコマンドライン環境へ (last modified 2
- 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 modi
- 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
- 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 2
- 書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ (last modified 2016/04/23)
- 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 | 一般 | ランダムに行を抽出 (last modified 2014/07/17)

①

書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ

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で、これらの内容は含まれていないためです(2016年4月23日追加)。

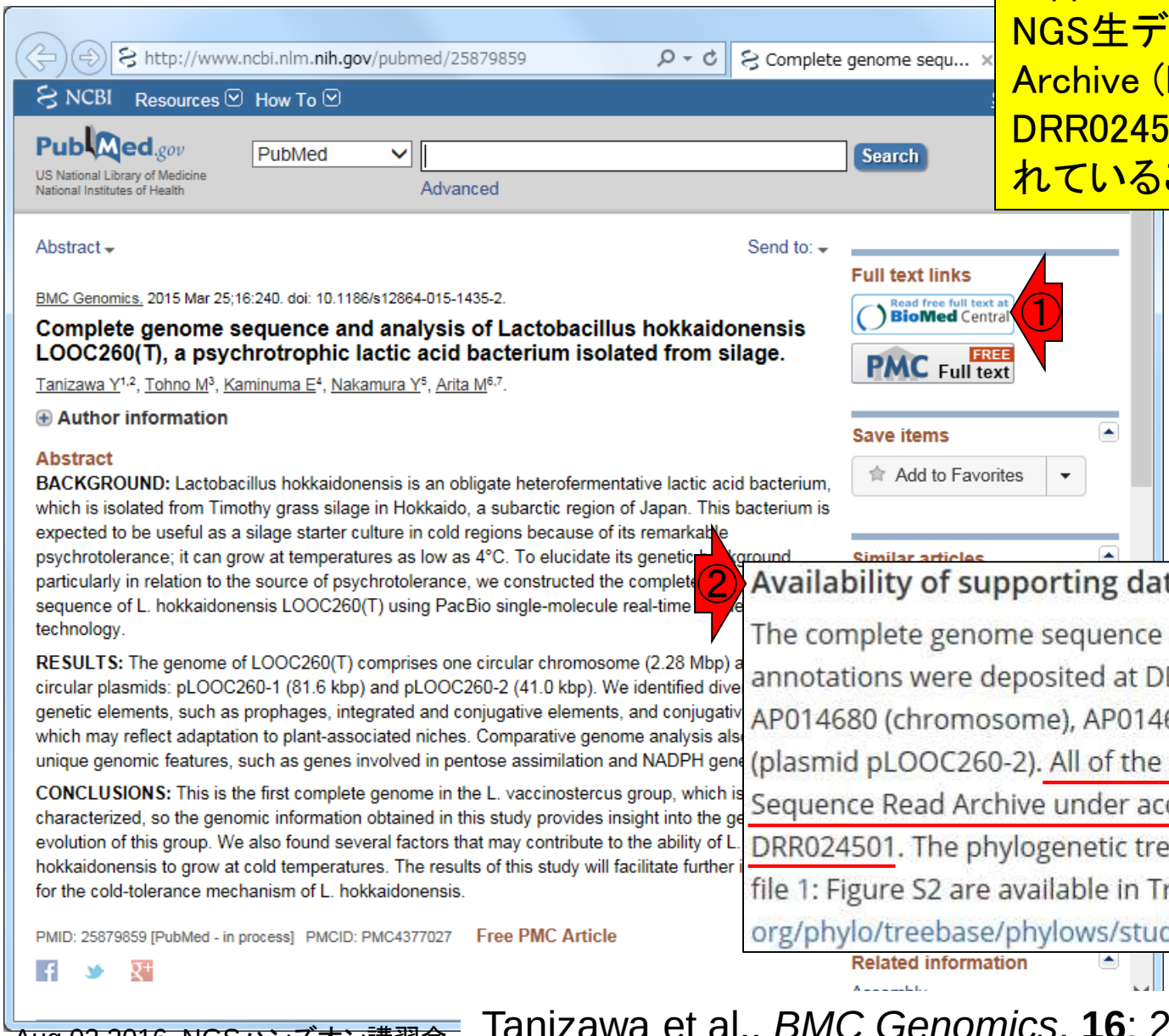
- 原稿PDF
- ウェブ資料PDF
  - Windows用(2016.03.29版; 約25MB)
  - Macintosh用

- (共有フォルダ設定情報を含む)連載第3回終了時点以降のovaファイルからのインストール手順:
  - Windows用(2015.12.28版; 約2MB)
  - Macintosh用(2015.12.28版; 約2MB)

②

W2-1：乳酸菌データ

原著論文(PMID: 25879859)の①Full textリンク先で全文を見られる。②Availability of supporting dataという項目をよく眺めると、NGS生データがDDBJ Sequence Read Archive (DDBJ SRA; 略してDRA)にDRR024500とDRR024501というIDで登録されていることがわかる。スライドを見るだけ

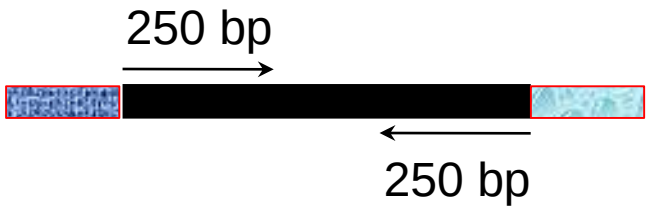


Tanizawa et al., *BMC Genomics*, **16**: 240, 2015

解析データ

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

- 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)



① Tanizawa et al., *BMC Genomics*, **16**: 240, 2015

W4-1：FastQC

DRR024501の最初の30万リードに限定したファイルに対して、①W4-1のFastQCを実行するところから行います。②をコピペ

書籍 | 日本乳酸菌学会誌 | 第6回ゲノムアセンブリ

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で、これらの内容は含まれ

- 原稿PDF
- ウェブ資料PDF
  - Windows用(2016.03.29版; 約25MB)
  - Macintosh用
- (共有フォルダ設定情報を含む)連載第3回終了時
  - Windows用(2015.12.28版; 約2MB)
  - Macintosh用(2015.12.28版; 約2MB)

```
wget -cq ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/DRA002/DRA002643/DRX0
ls -lh
```

①

- FastQC (ver. 0.11.4)[W4-1]
  FastQC実行結果を共有フォルダ(/home/iu/Desktop/mac_share)に保存している。

②

```
cd ~/Documents/DRR024501

pwd
ls -l
time fastqc2 -q DRR024501sub_1.fastq.gz --outdir=/home/iu/Desktop/mac_share
time fastqc2 -q DRR024501sub_2.fastq.gz --outdir=/home/iu/Desktop/mac_share
ls -l ~/Desktop/mac_share
date
```

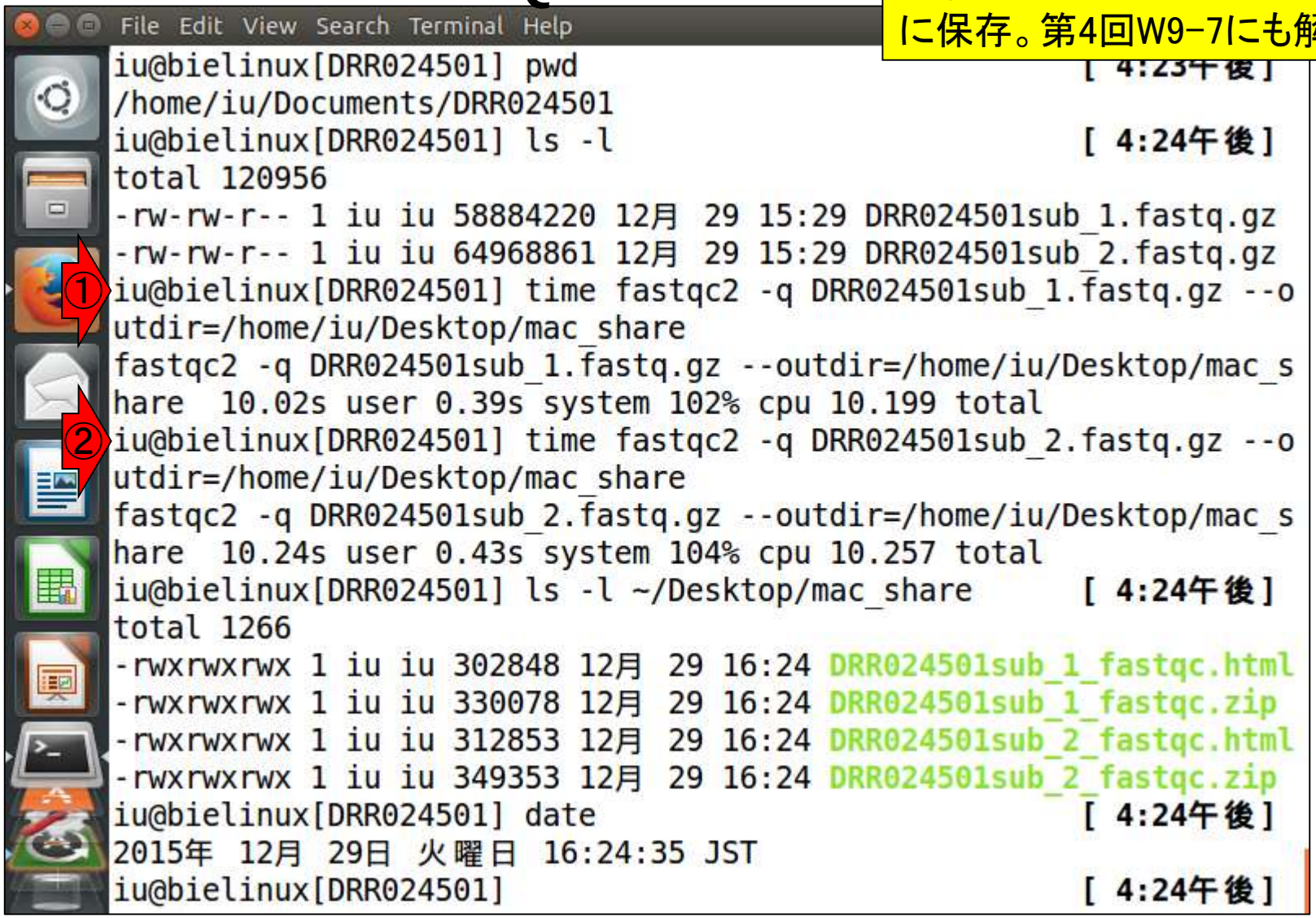
- FastQC実行結果を眺める(forward側)[W4-2]
  DRR024501sub_1_fastqc.html
- FastQC実行結果を眺める(reverse側)[W4-3]
  DRR024501sub_2_fastqc.html

アダプタートリミング

- FaQCs (Lo and Chain, BMC Bioinformatics, 2014)を実行[W5-1]
  アダプター配列や低クオリティリードの除去を行うプログラムFaQCsを実行し、結果をresultディレクトリに

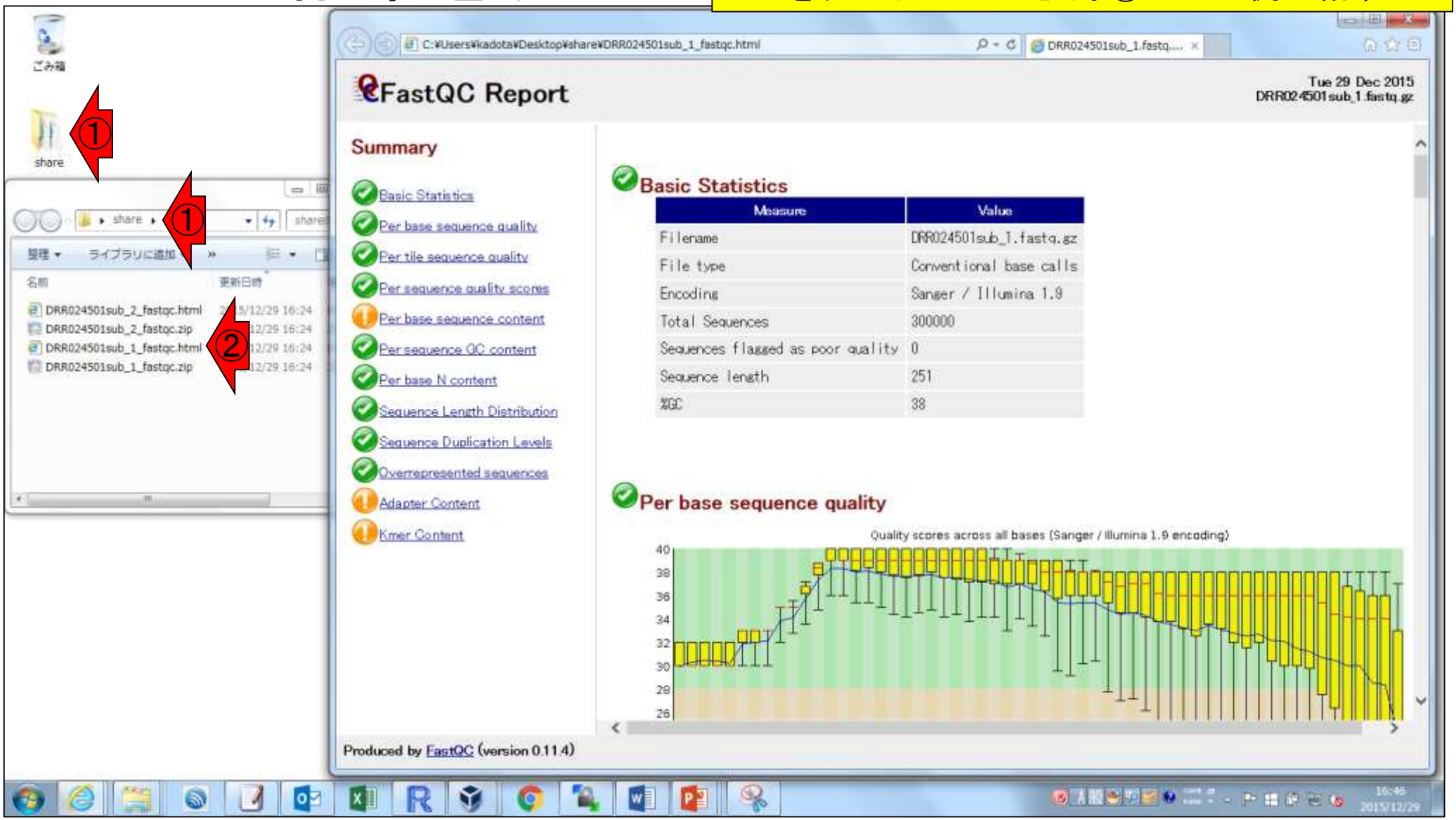
W4-1：FastQC

①②FastQC (ver. 0.11.4)をtimeコマンドをつけて実行。各ファイルにつき約10秒。実行結果は共有フォルダ(/home/iu/Desktop/mac_share)に保存。第4回W9-7にも解説あり



W4-2：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FastQC実行結果ファイルを眺めることができる。②forward側の結果



W4-2：結果を眺める

FastQC実行結果の解説は第4回W8-2とW17-2にもあり。①入力ファイル。これはforward側の結果。②リード数。30万リードであることがわかる。③配列長。251 bpであることがわかる

FastQC Report

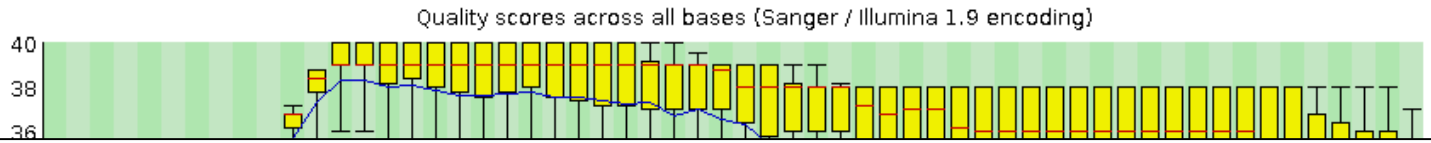
Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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	DRR024501sub_1.fastq.gz ①
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	300000 ②
Sequences flagged as poor quality	0
Sequence length	251 ③
%GC	38

Per base sequence quality



W4-2：結果を眺める

全体的なクオリティは、①赤枠内の色でわかる。概ね、信号通りの理解でよい。第4回W17-2のRNA-seqデータのFastQC結果と比較するとよい。ゲノムデータの場合はRNA-seqデータよりもcoverageが一定なので、一般によりよい結果になる

FastQC Report

Summary

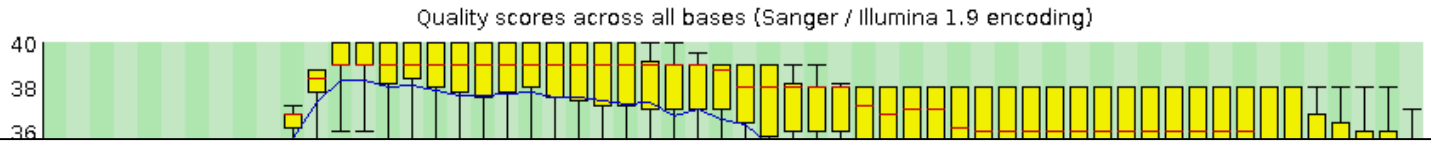
- Basic Statistics
- Per base sequence quality
- Per tile sequence quality
- Per sequence quality scores
- Per base sequence content
- Per sequence GC content
- Per base N content
- Sequence Length Distribution
- Sequence Duplication Levels
- Overrepresented sequences
- Adapter Content
- Kmer Content

①

Basic Statistics

Measure	Value
Filename	DRR024501sub_1.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	300000
Sequences flagged as poor quality	0
Sequence length	251
%GC	38

Per base sequence quality



W4-2：結果を眺める

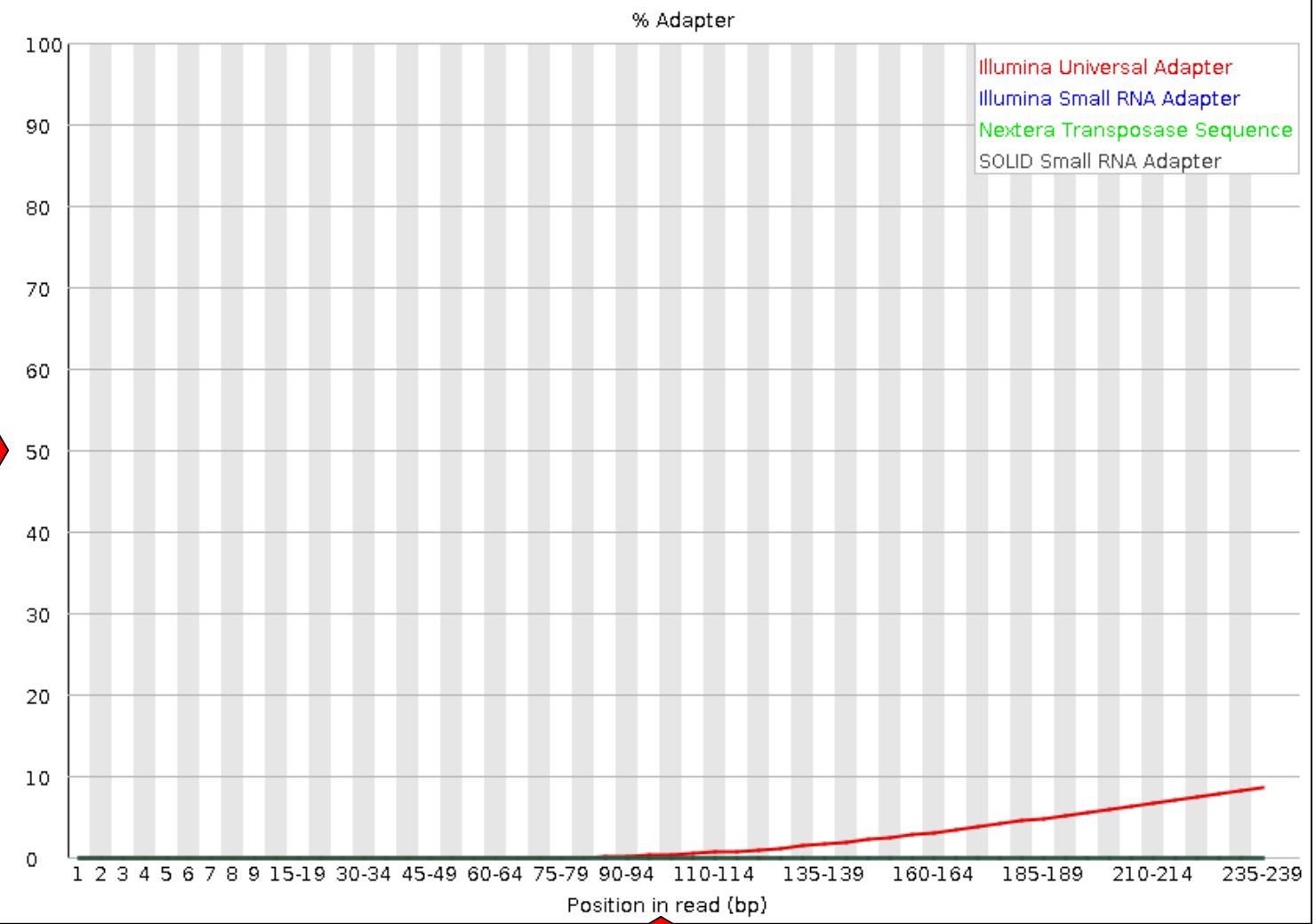
例えば黄色の項目①Adapter Contentを眺める。②横軸は全251 bpからなるリードのポジション、③縦軸はポジションごとの既知のアダプター配列を含む割合を示す

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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

① ③

Adapter Content



②

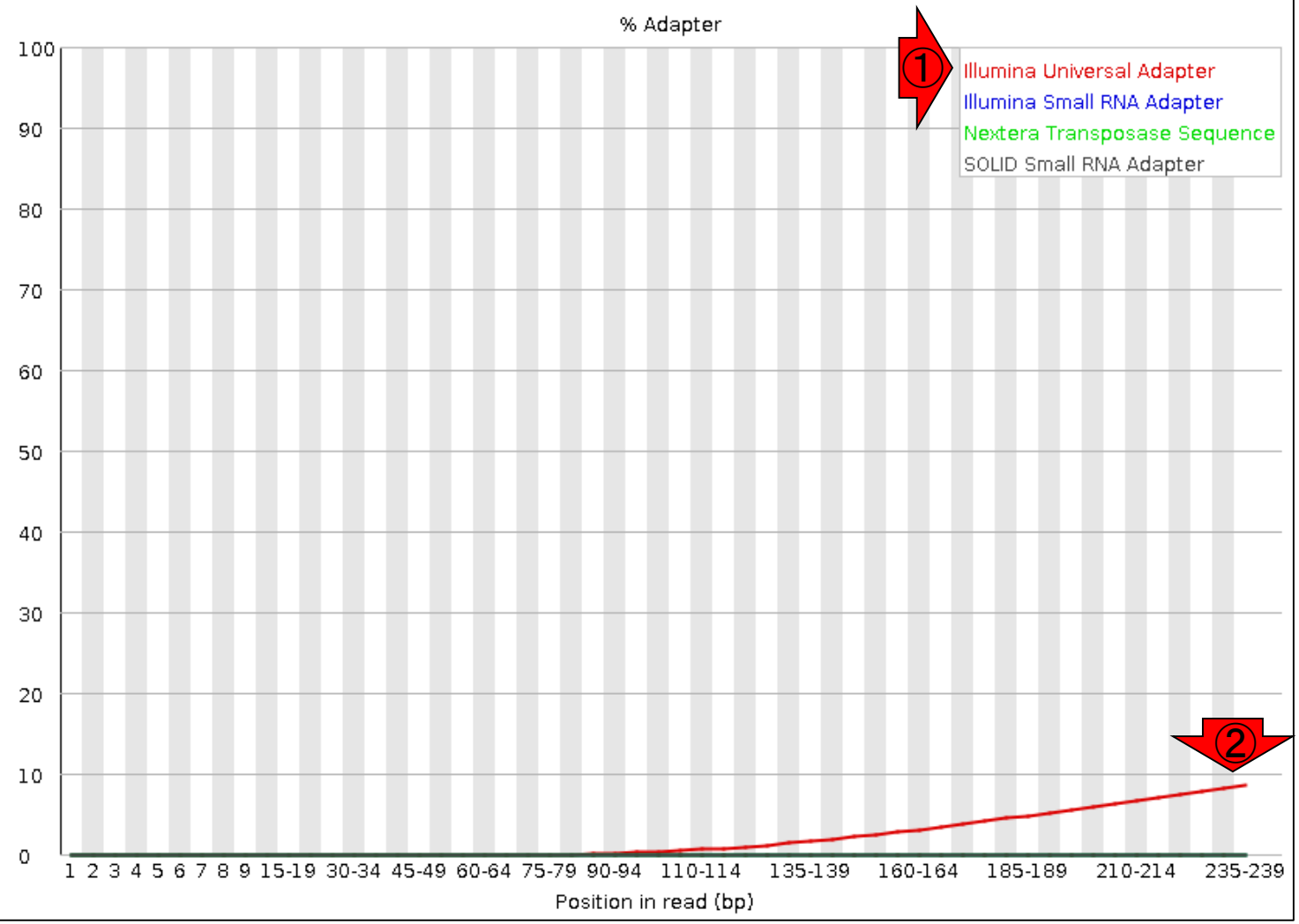
W4-2：結果を眺める

①Illumina Universal Adapterが3’側に多く含まれており、②終端付近では全リードの10%弱に含まれるほどであることもわかる

Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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

Adapter Content



W4-3：結果を眺める

①reverse側のFastQC結果。②リード数、③配列長ともにforward側と同じ

FastQC Report

Tue 29 Dec 2015
DRR024501sub_2.fastq.gz

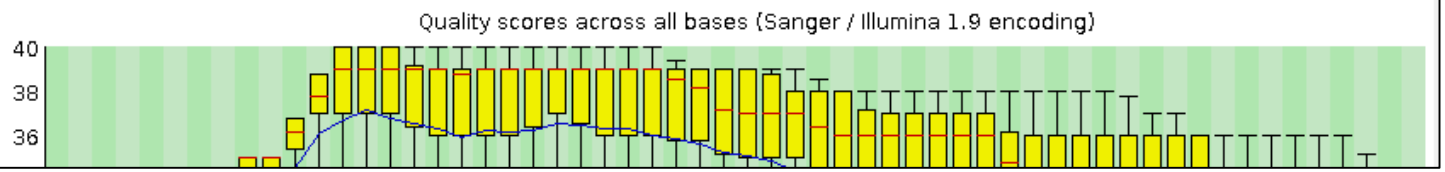
Summary

- Basic Statistics
- Per base sequence quality
- Per tile 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	DRR024501sub_2.fastq.gz ①
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	300000 ②
Sequences flagged as poor quality	0
Sequence length	251 ③
%GC	38

Per base sequence quality



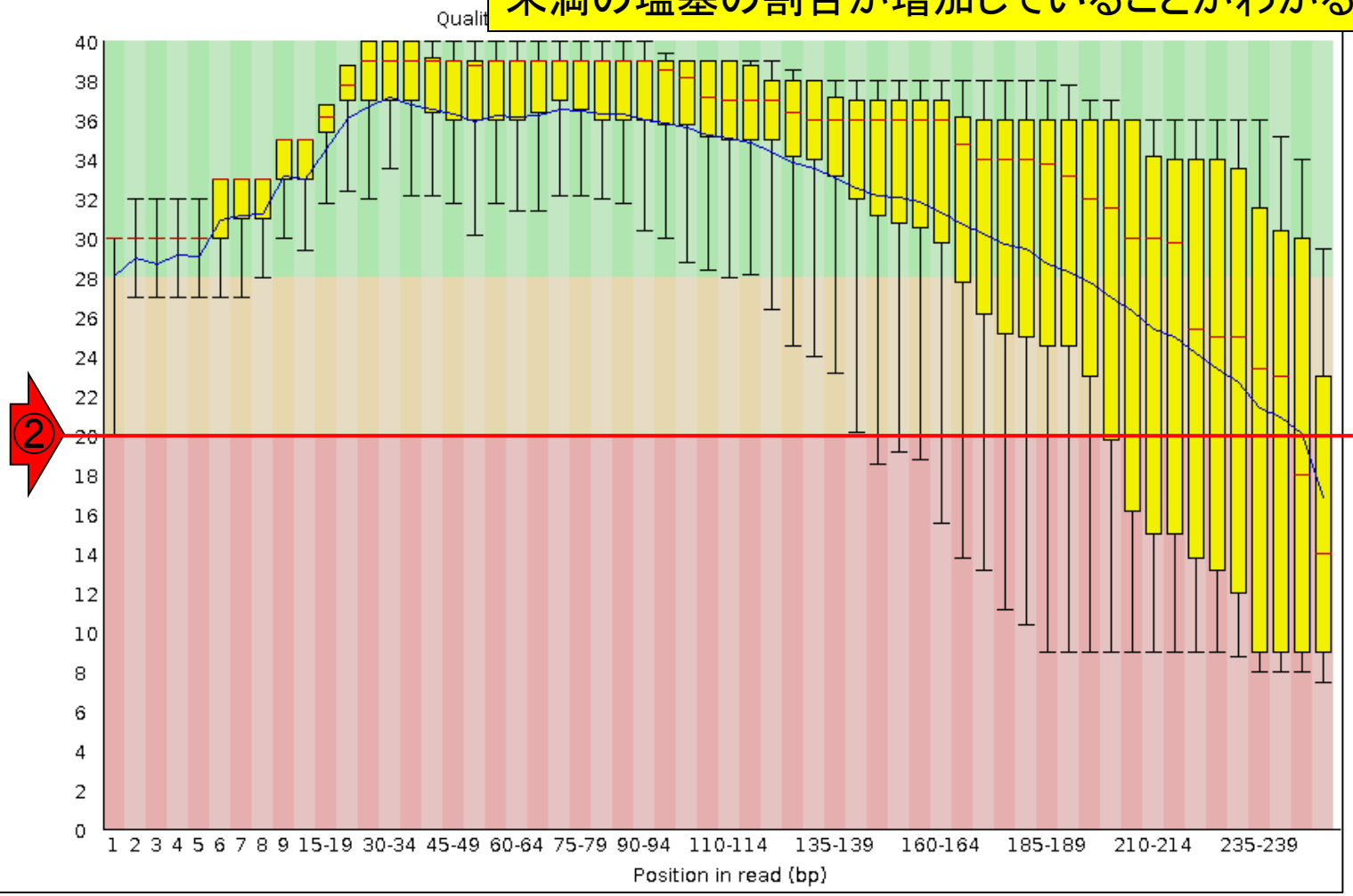
W4-3：結果を眺める

①Per base sequence qualityの項目が赤になっているので眺める。この図の縦軸はクオリティスコア。赤線のスコア20を超えているかどうかが1つの目安。この場合、3’末端にいくほどスコア20未満の塩基の割合が増加していることがわかる

Summary

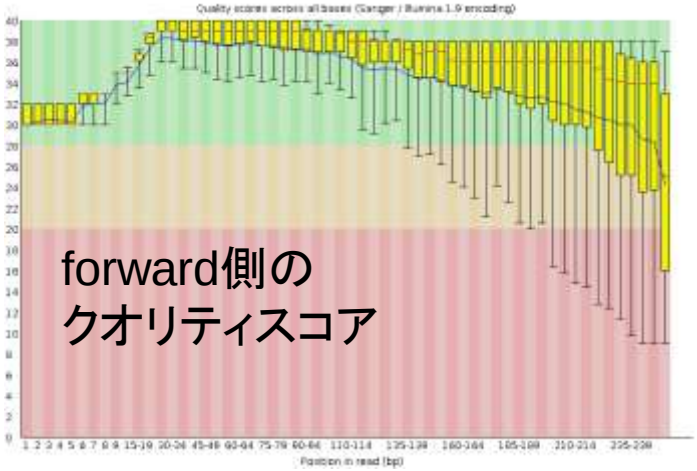
- Basic Statistics
- Per base sequence quality ①
- Per tile sequence quality
- Per sequence quality scores
- Per base sequence content
- Per sequence GC content
- Per base N content
- Sequence Length Distribution
- Sequence Duplication Levels
- Overrepresented sequences
- Adapter Content
- Kmer Content

Per base sequence quality

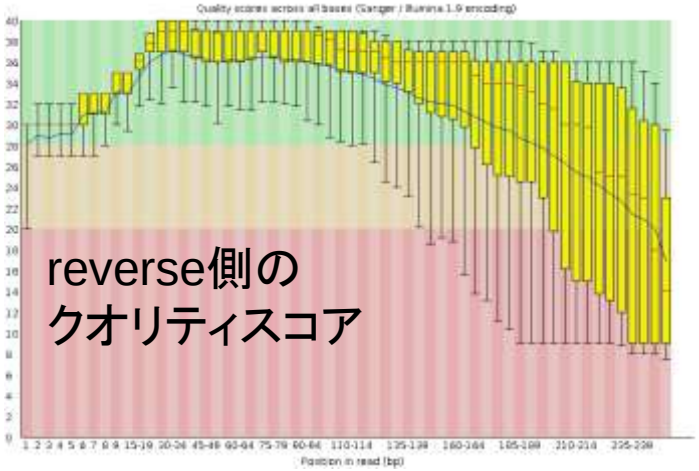


第6回原稿PDF中の図1と同じ

W4-4：FastQCまとめ

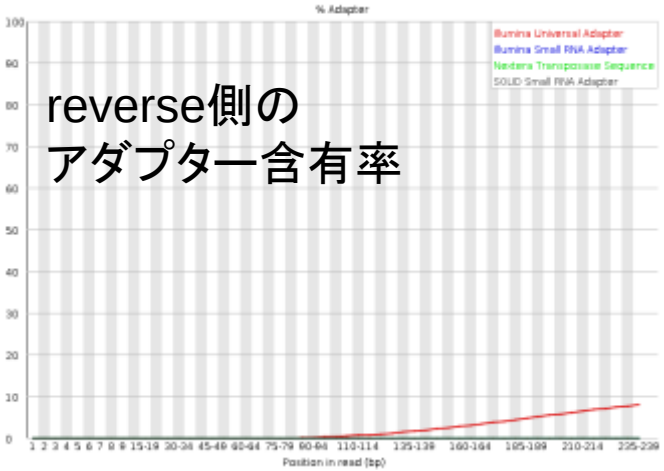


forward側の
クオリティスコア



reverse側の
クオリティスコア

forward側の
アダプター含有率



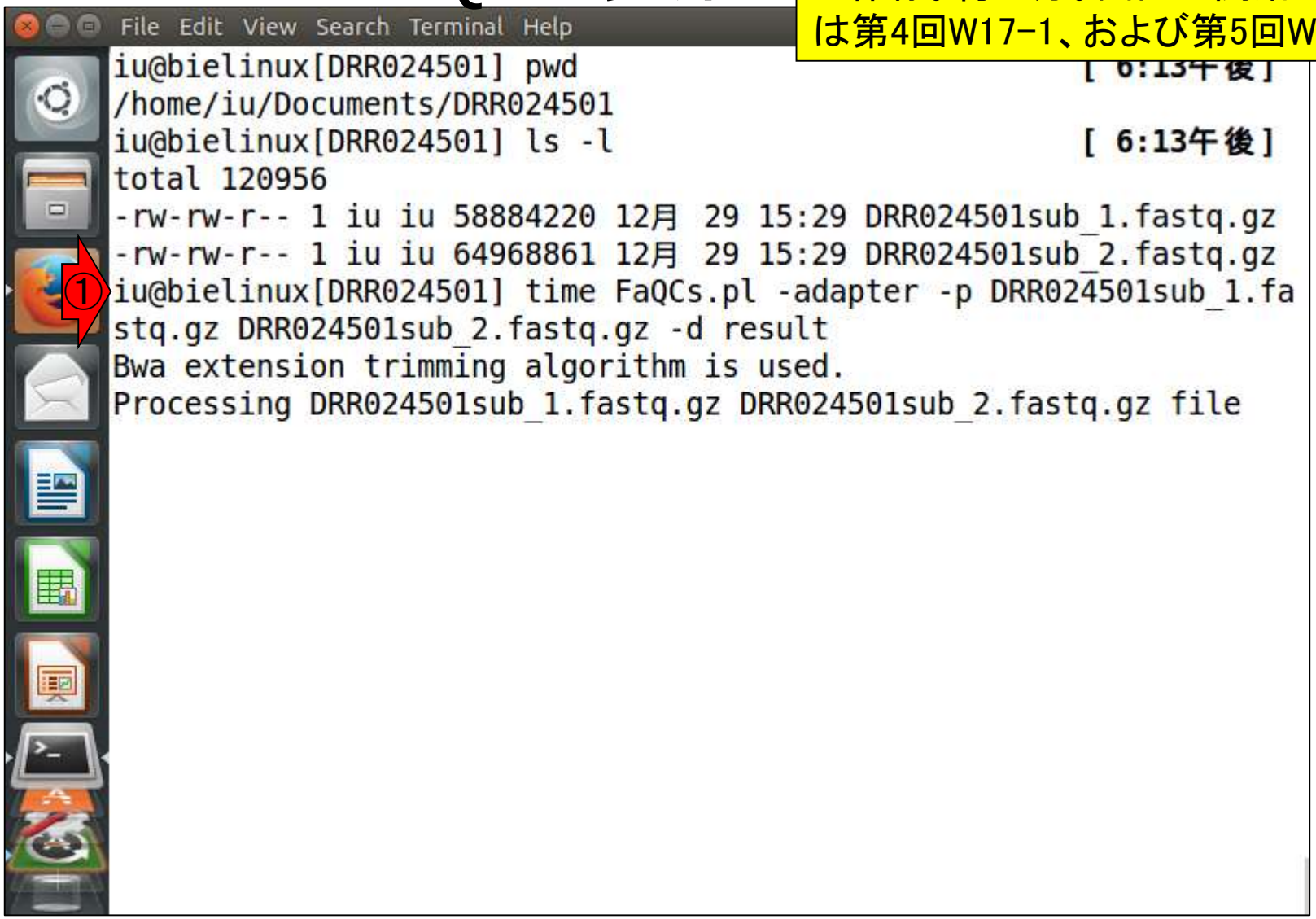
reverse側の
アダプター含有率

Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

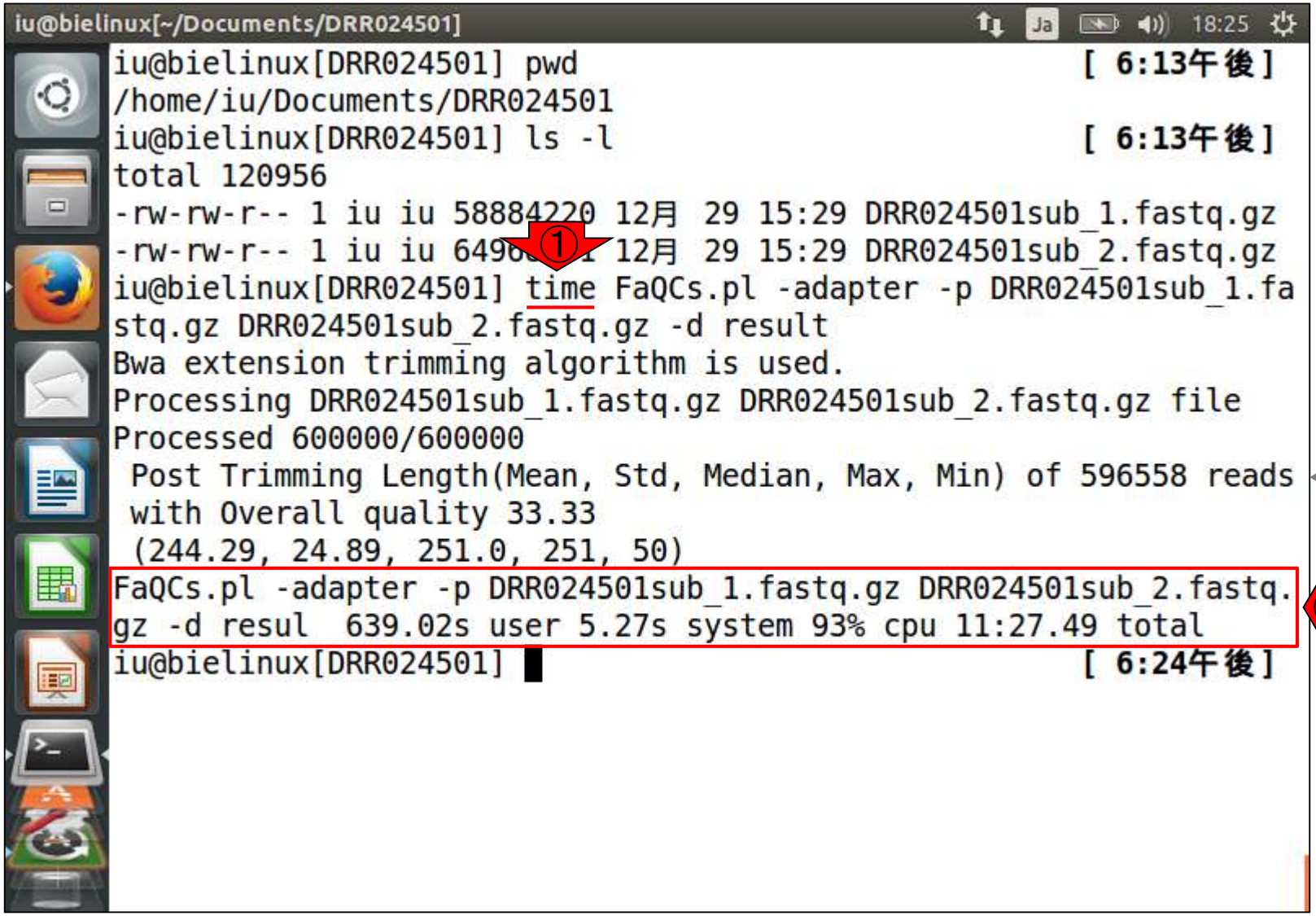
W5-1：FaQCs実行

①アダプター配列や低クオリティリードの除去を行うプログラムFaQCsを実行し、結果をresultディレクトリに保存。約12分。画面は開始1分後の状況。FaQCsは第4回W17-1、および第5回W1-1でも実行している



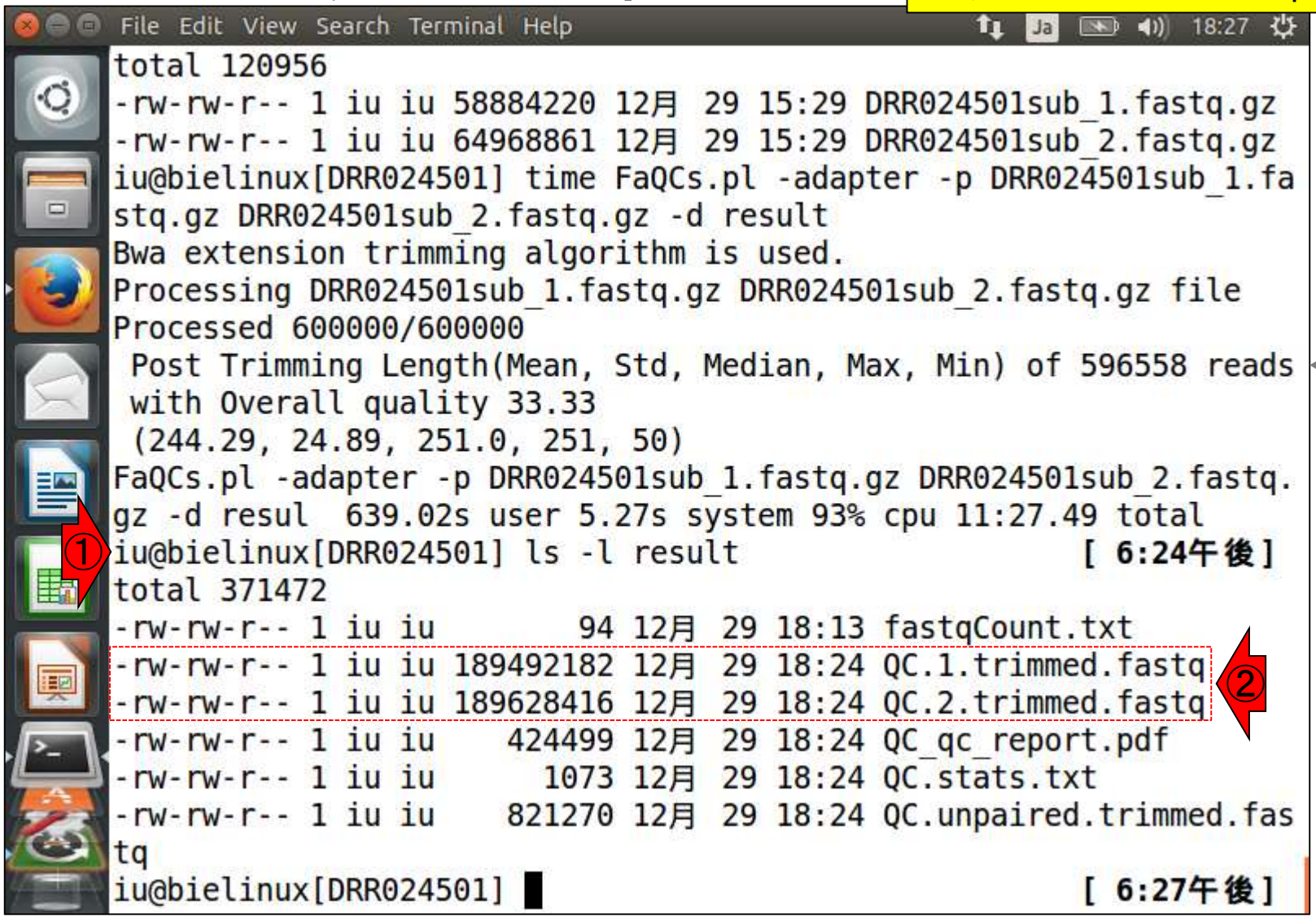
W5-2：実行結果

FaQCs実行後の状態。①timeコマンドをつけて実行したので、②赤枠分が余分に表示されている



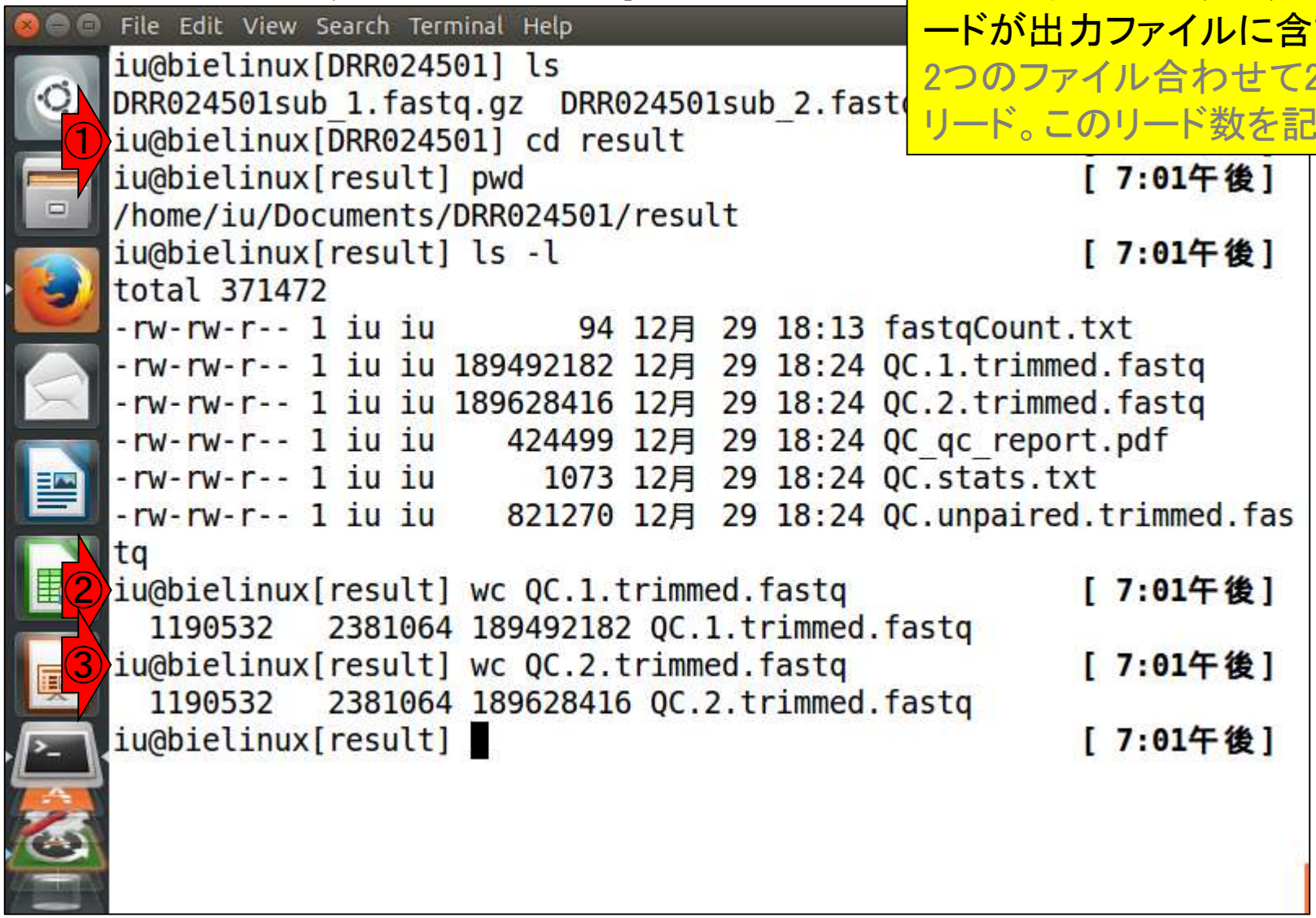
W5-2：実行結果

結果が保存されているresultディレクトリを①lsしている。②FaQCsの主な実行結果ファイルは、QC.1.trimmed.fastqとQC.2.trimmed.fastq



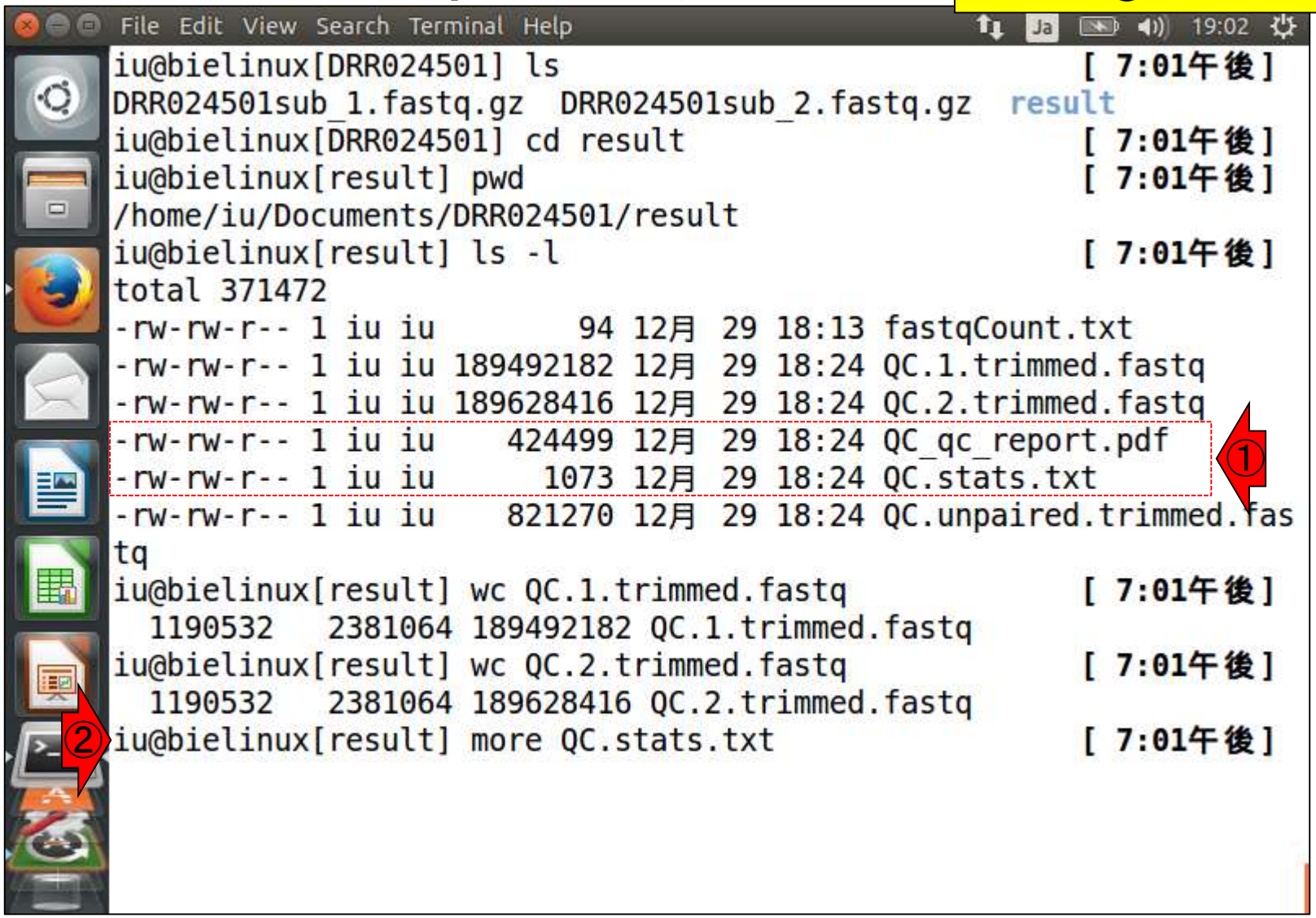
W5-2：実行結果

①resultディレクトリに移動し、FaQCs実行結果ファイルの行数確認。②forward側、③reverse側ともに1,190,532行 / 4 = 297,633リードが出力ファイルに含まれることがわかる。2つのファイル合わせて297,633 × 2 = 595,266リード。このリード数を記憶に留めておく



W5-3：結果の概要

①FaQCs実行結果の概要は、この2つのファイル中にある。どちらも同じ内容なので、ここでは②moreでQC.stats.txtを眺める



W5-3：結果の概要

①FaQCs実行前(Before Trimming)の入力データの概要。片側30万リードなので、Readsのところが600000になっているのは妥当

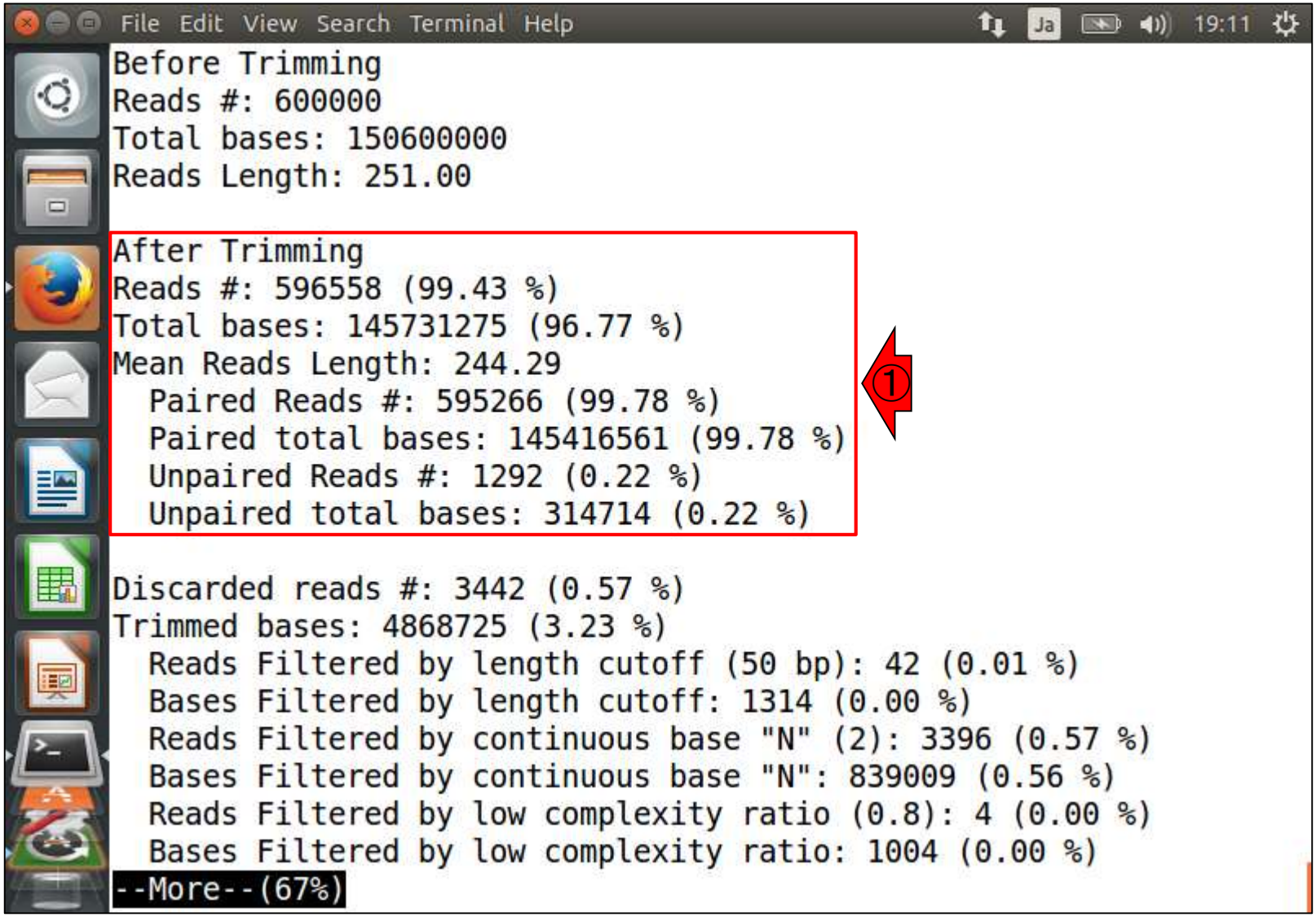
File Edit View Search Terminal Help

Ja 19:11

```
Before Trimming
Reads #: 600000
Total bases: 150600000
Reads Length: 251.00

After Trimming
Reads #: 596558 (99.43 %)
Total bases: 145731275 (96.77 %)
Mean Reads Length: 244.29
  Paired Reads #: 595266 (99.78 %)
  Paired total bases: 145416561 (99.78 %)
  Unpaired Reads #: 1292 (0.22 %)
  Unpaired total bases: 314714 (0.22 %)

Discarded reads #: 3442 (0.57 %)
Trimmed bases: 4868725 (3.23 %)
  Reads Filtered by length cutoff (50 bp): 42 (0.01 %)
  Bases Filtered by length cutoff: 1314 (0.00 %)
  Reads Filtered by continuous base "N" (2): 3396 (0.57 %)
  Bases Filtered by continuous base "N": 839009 (0.56 %)
  Reads Filtered by low complexity ratio (0.8): 4 (0.00 %)
  Bases Filtered by low complexity ratio: 1004 (0.00 %)
--More--(67%)
```

①FaQCs実行後(After Trimming)の概要

W5-3：結果の概要



W5-3：結果の概要

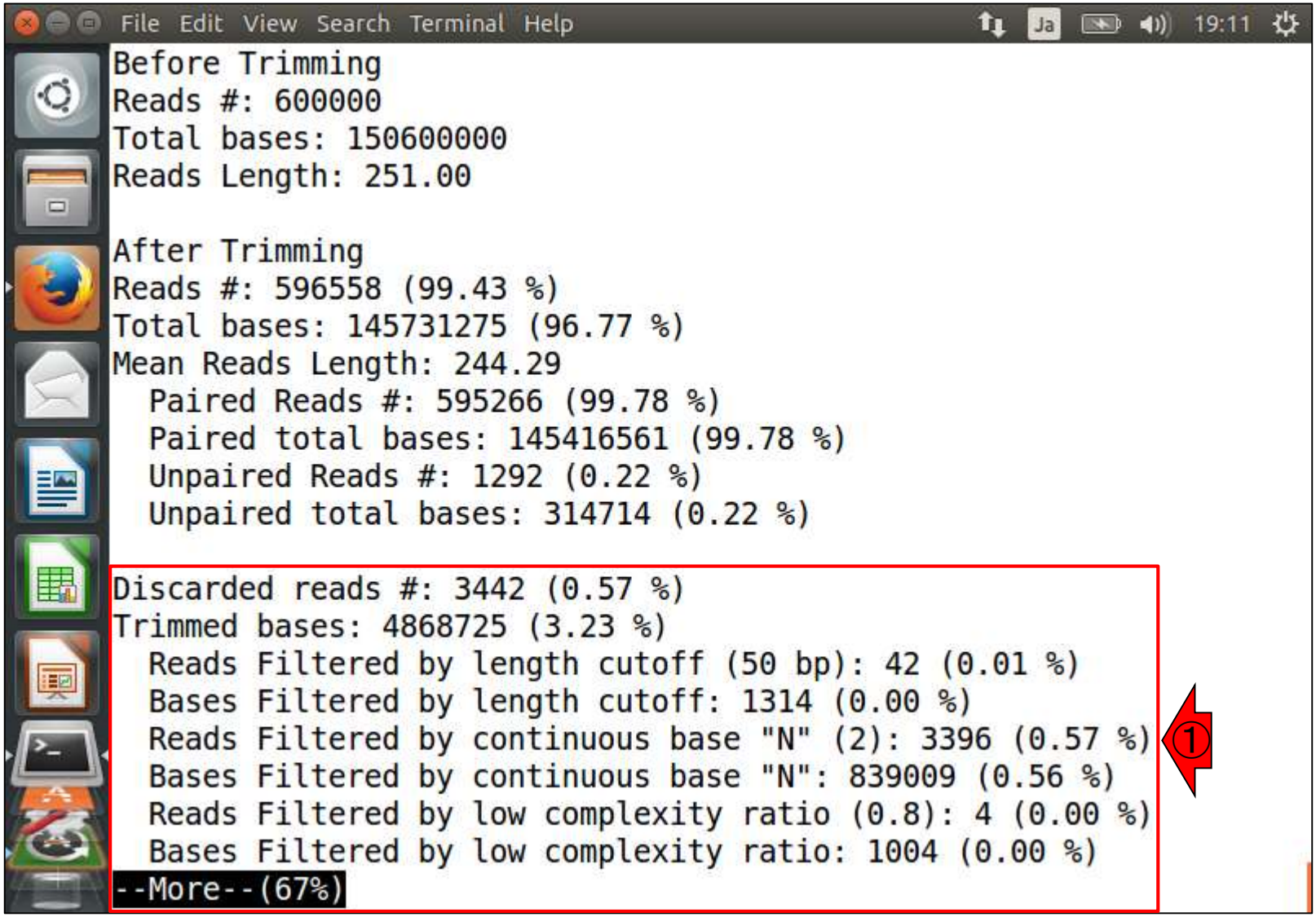
①トリム後に596,558リード残ったことがわかる。FaQCsはforward側とreverse側を独立に実行したのち、両方で生き残ったものを②Paired Reads (595,266個)、片側のみで生き残ったものを③Unpaired Reads (1,292個)としてレポートしている

File Edit View Search Terminal Help

```
Before Trimming
Reads #: 600000
Total bases: 150600000
Reads Length: 251.00

After Trimming
Reads #: 596558 (99.43 %)
Total bases: 145731275 (96.77 %)
Mean Reads Length: 244.29
  Paired Reads #: 595266 (99.78 %)
  Paired total bases: 145416561 (99.78 %)
  Unpaired Reads #: 1292 (0.22 %)
  Unpaired total bases: 314714 (0.22 %)

Discarded reads #: 3442 (0.57 %)
Trimmed bases: 4868725 (3.23 %)
  Reads Filtered by length cutoff (50 bp): 42 (0.01 %)
  Bases Filtered by length cutoff: 1314 (0.00 %)
  Reads Filtered by continuous base "N" (2): 3396 (0.57 %)
  Bases Filtered by continuous base "N": 839009 (0.56 %)
  Reads Filtered by low complexity ratio (0.8): 4 (0.00 %)
  Bases Filtered by low complexity ratio: 1004 (0.00 %)
--More--(67%)
```

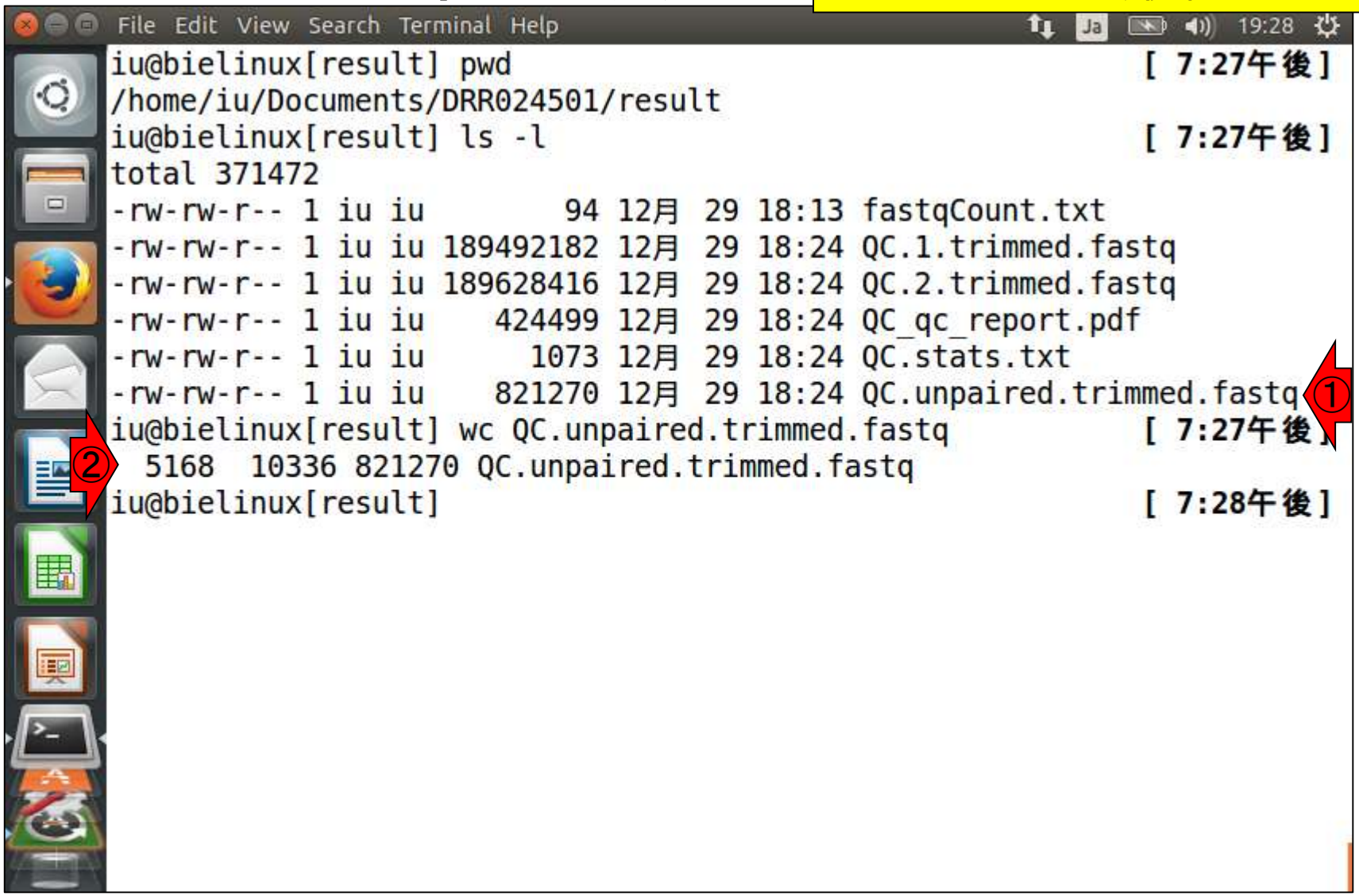
W5-3：結果の概要

①捨てられた(Discarded)リードの内訳も書かれている



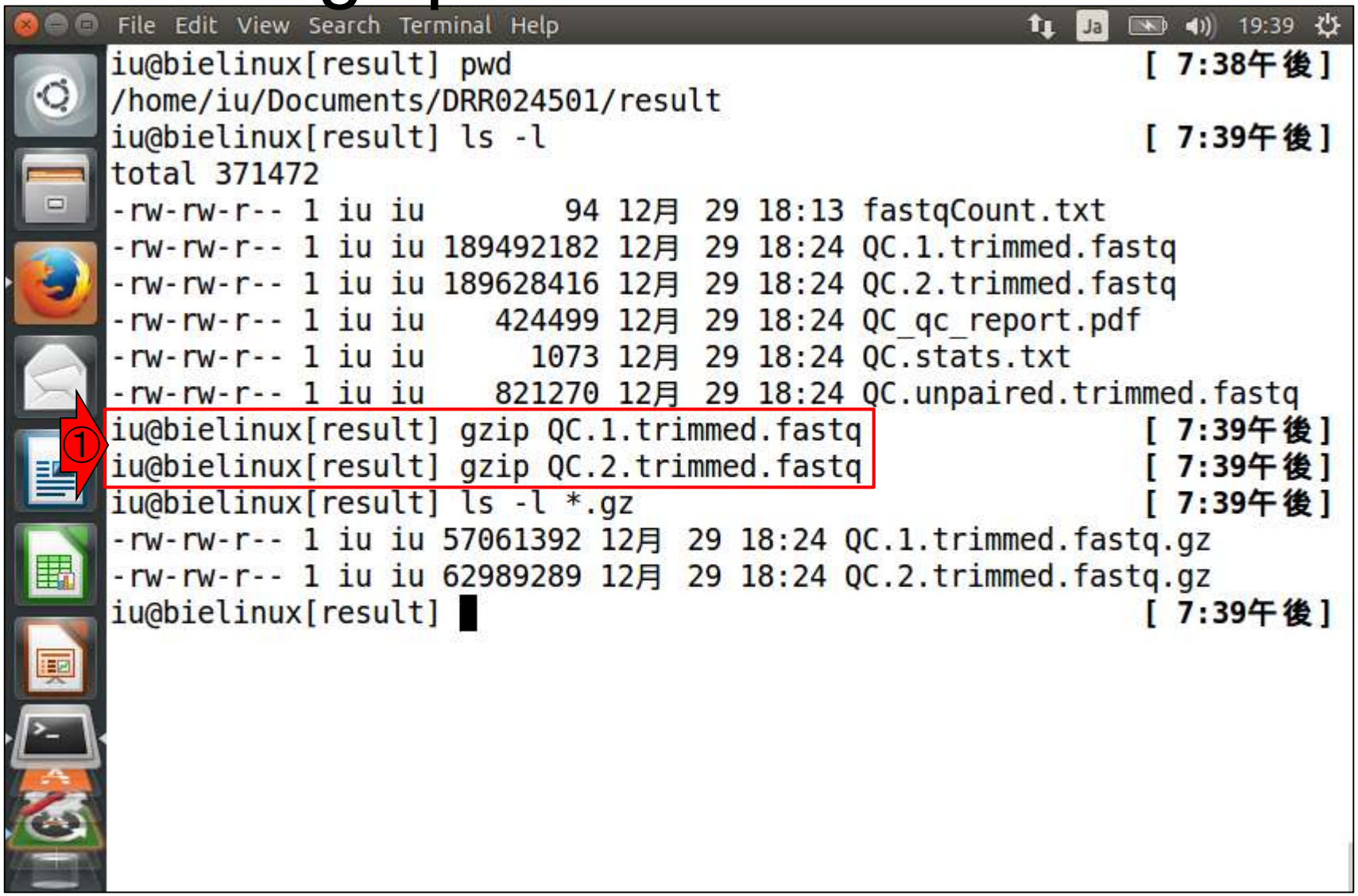
W5-3：結果の概要

①特に言及しなかったQC.unpaired.trimmed.fastqファイルの中身は想像がつく。②なぜ行数が5,168になるのかまでは、使わないので深追いしない



①gzip圧縮しておく

W5-4：gzip圧縮

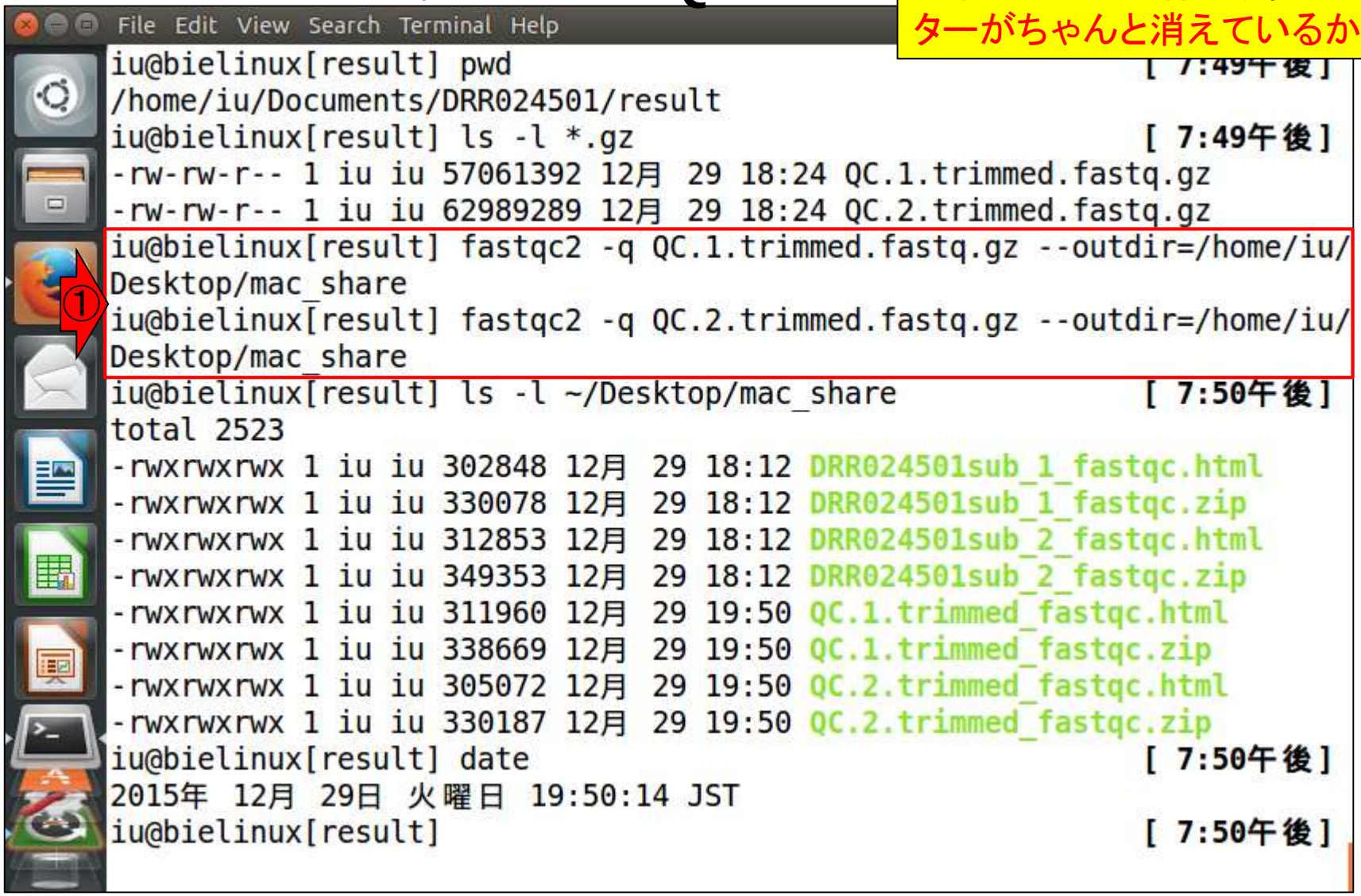


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

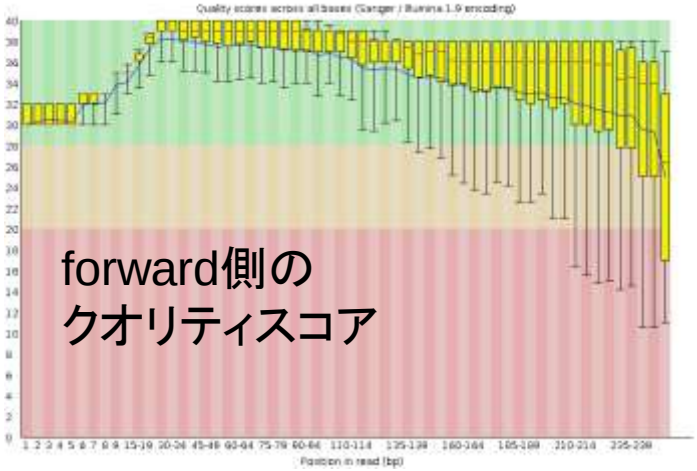
W6-1：再度FastQC

①FaQCs実行結果のgzip圧縮ファイルを入力として、FastQC (ver. 0.11.4)を実行し、結果を共有フォルダに保存。約1分。目的は、アダプターがちゃんと消えているかどうかの確認

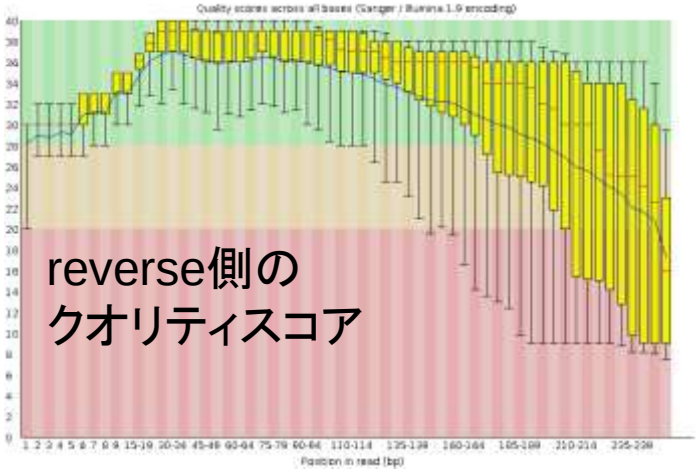


第6回原稿PDF中の図2と同じ

W6-2：FastQCまとめ（FaQCs実行後）

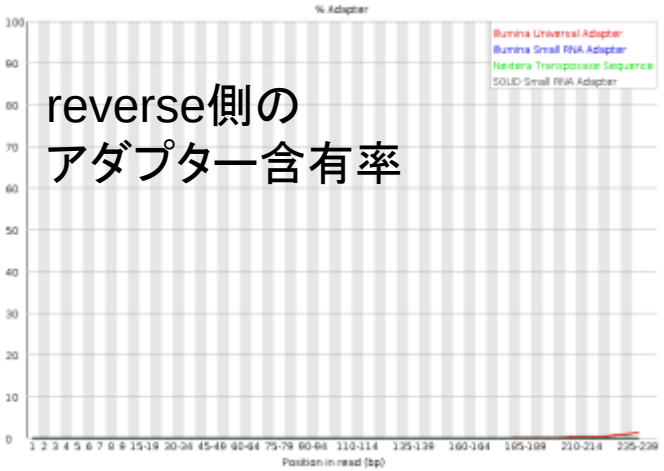


forward側の
クオリティスコア



reverse側の
クオリティスコア

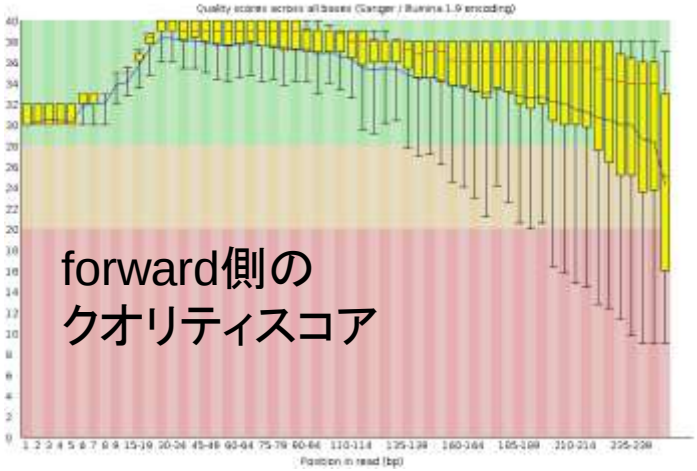
forward側の
アダプター含有率



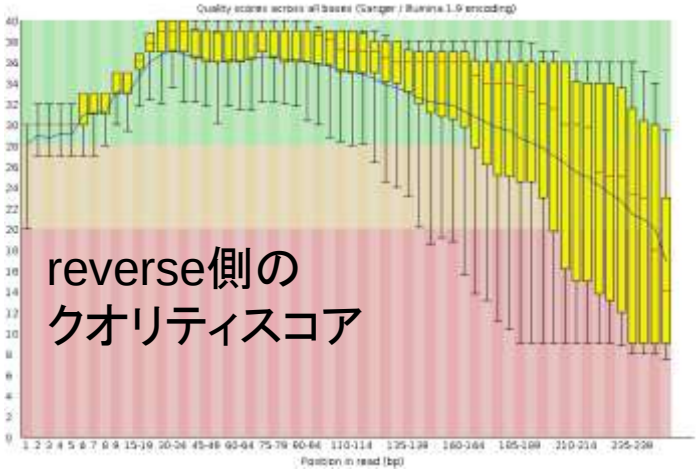
reverse側の
アダプター含有率

第6回原稿PDF中の図1と同じ

W4-4：FastQCまとめ（FaQCs実行前）

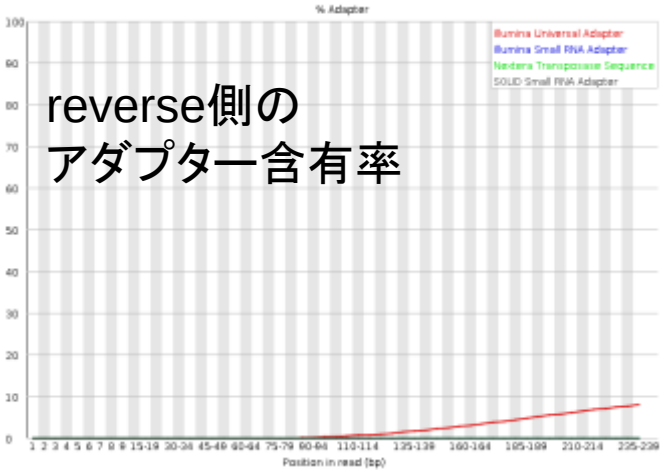


forward側の
クオリティスコア



reverse側の
クオリティスコア

forward側の
アダプター含有率



reverse側の
アダプター含有率

参考

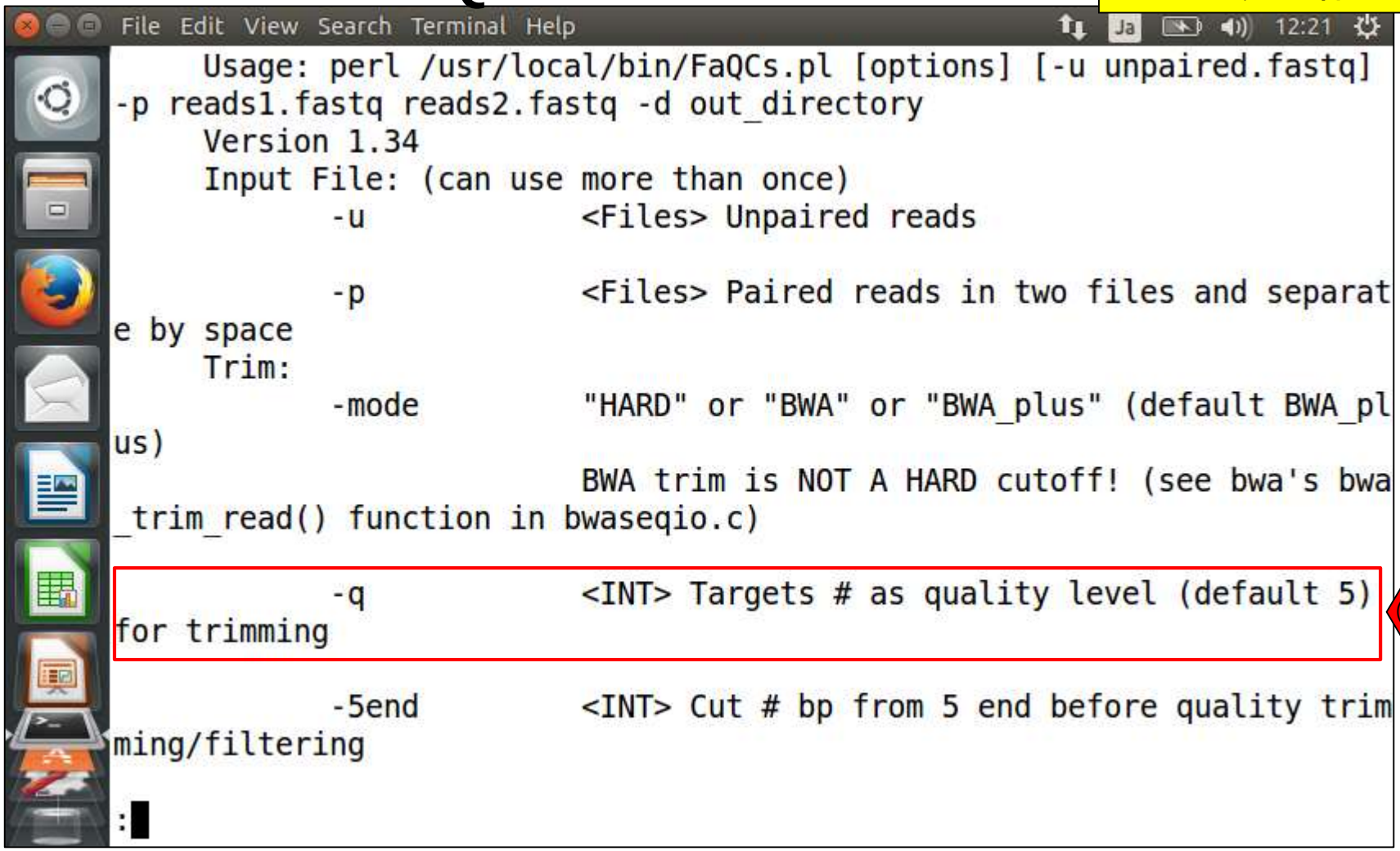
①「FaQCs.pl –h」をlessつきで実行

W6-3：FaQCsのオプション

参考

例えば①-qオプションのデフォルトが5なので、10にするとトリムされる塩基数が増えるのだろうとか…

W6-3：FaQCsのオプション

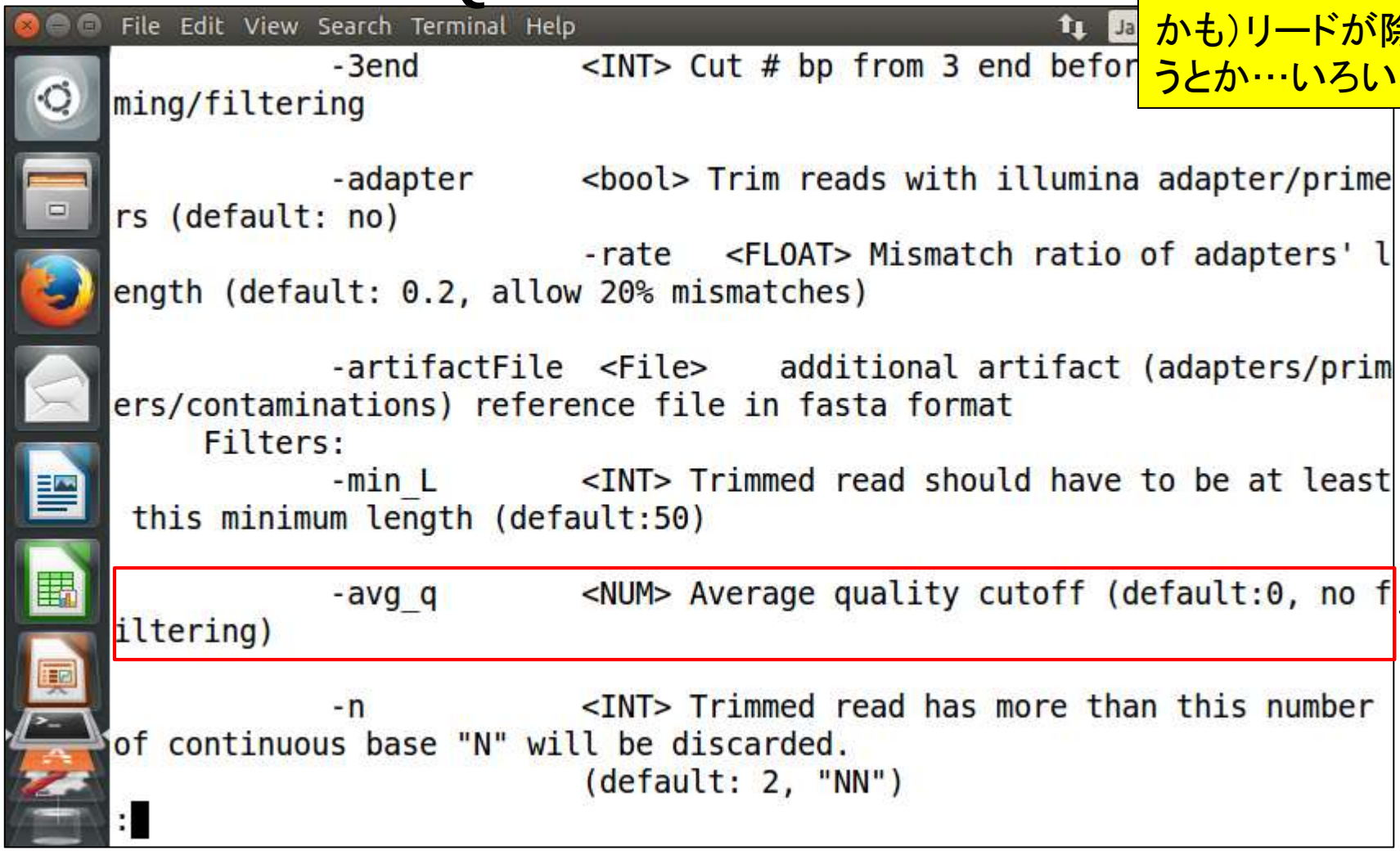


①

参考

W6-3：FaQCsのオプション

①-avg_qオプションのデフォルトが0なので、20にすると平均クオリティスコアが20未満の（以下かも）リードが除去されるのだろうとか…いろいろと妄想する

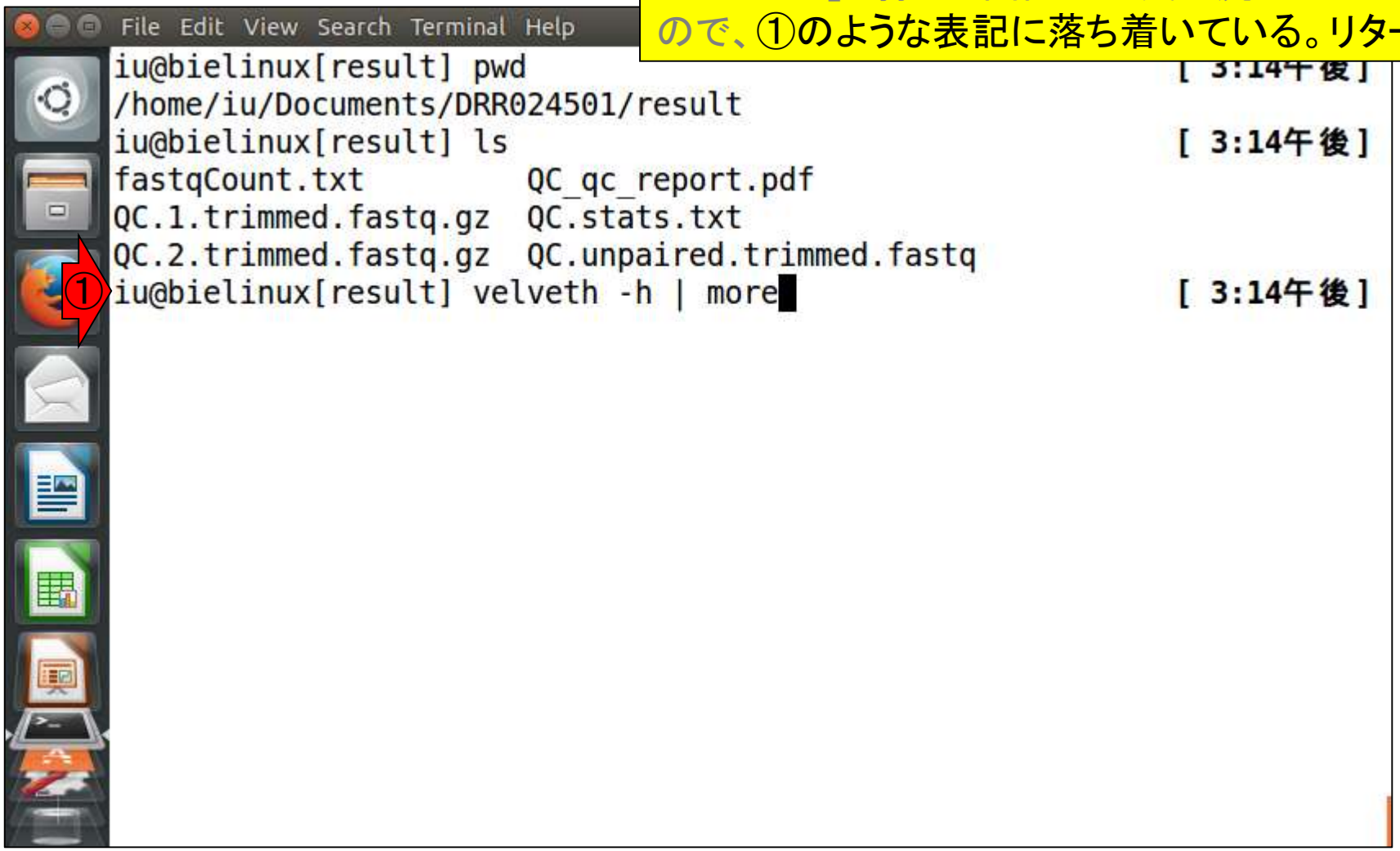


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

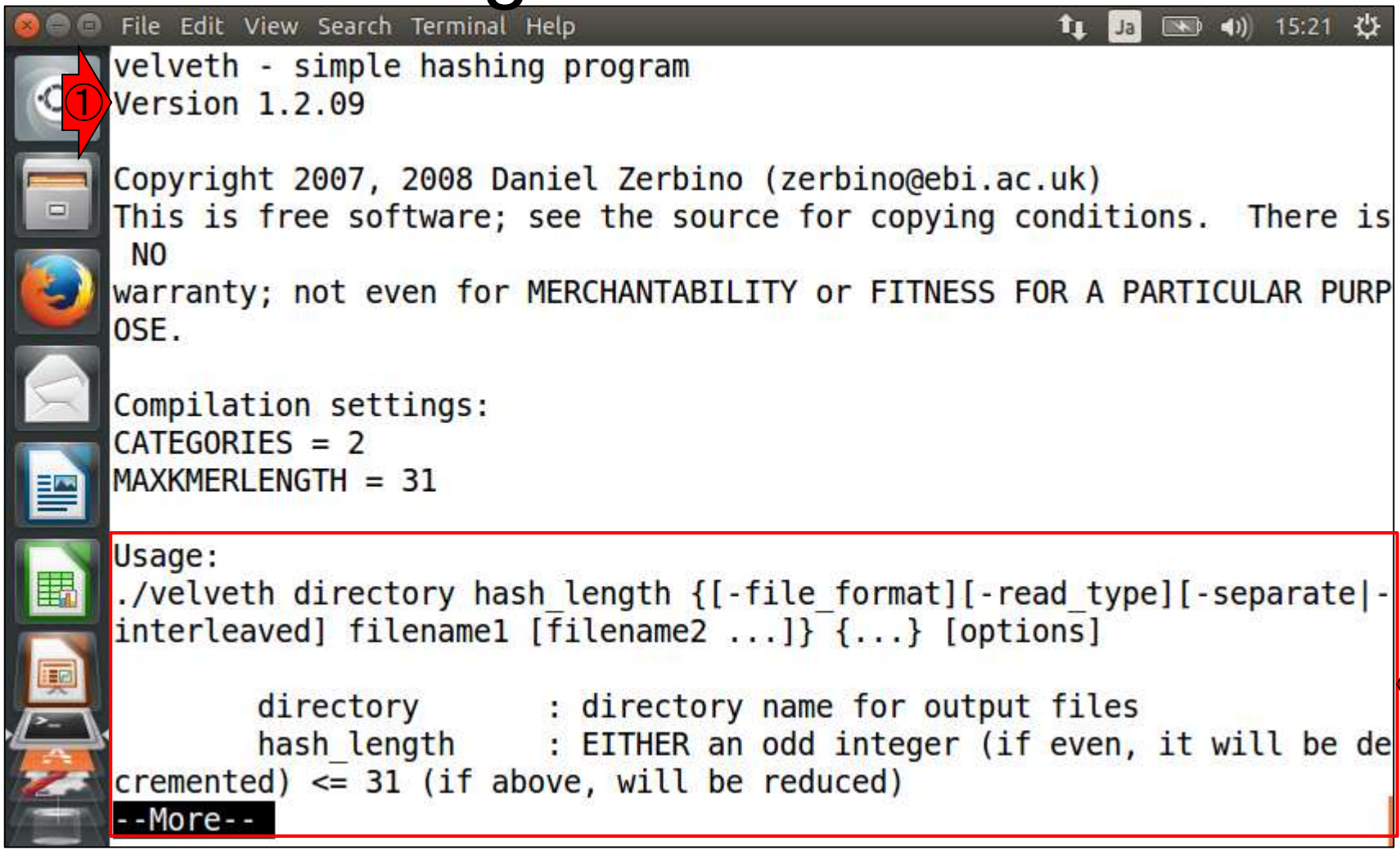
W7-1：Velvet

①いきなり「velveth –h | more」と書いてヘルプを表示させているが、「velvet –h」と打つとvelvethに修正を試みること、「velveth –h」と打つと画面が一気に流れてしまうことがわかったので、①のような表記に落ち着いている。リターンキーを押す



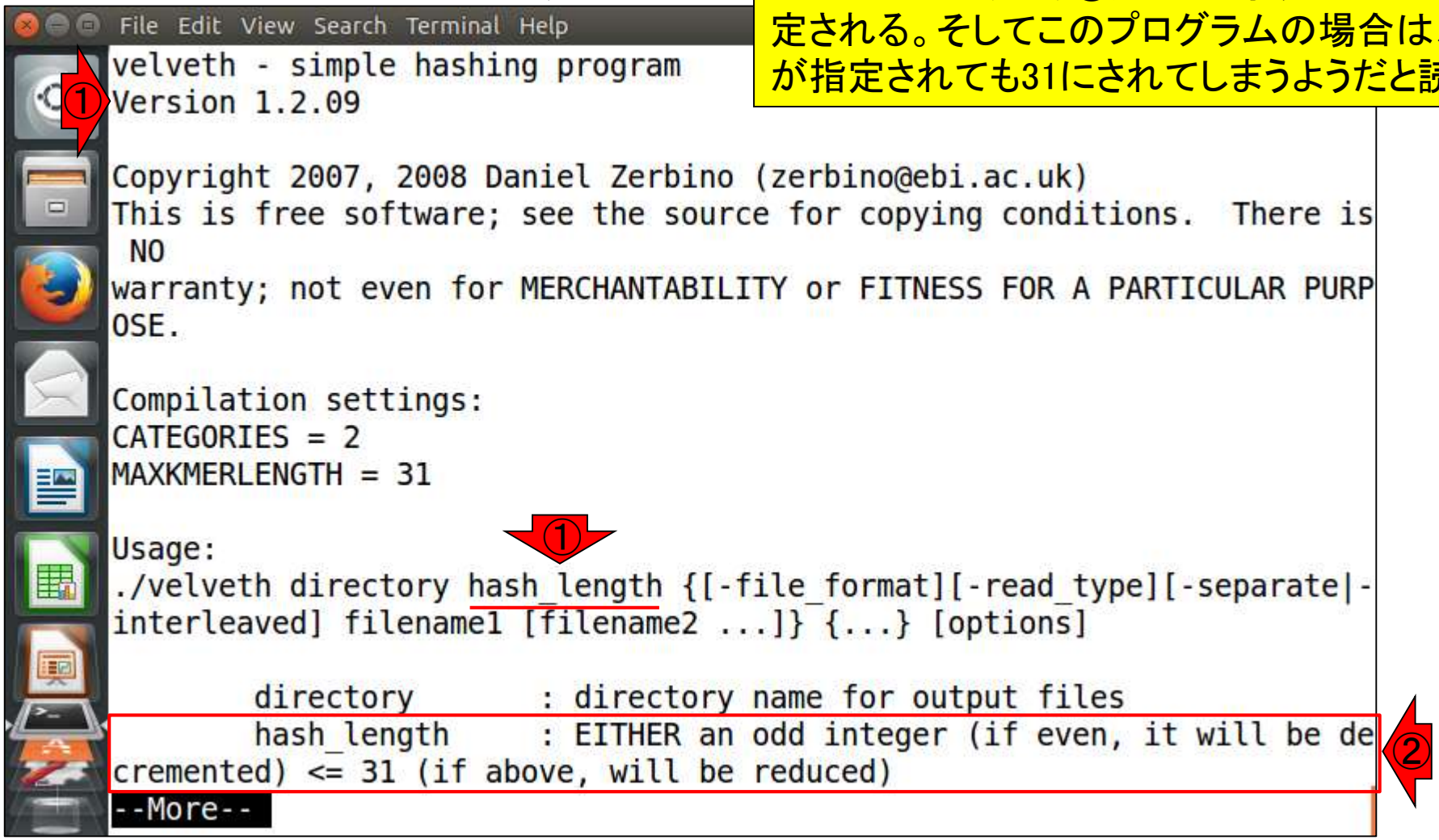
W7-2：Usage

①Bio-Linuxにプレインストールされ ているVelvetのバージョンは1.2.09。②が基本的な利用法



W7-3：k値指定

①赤下線のhash_lengthのところで指定する数値がアセンブリ時の主要なオプションであるk-merのk値。第5回でも述べたように、②通常は奇数(an odd integer)が指定される。そしてこのプログラムの場合は、31以上の値が指定されても31にされてしまうようだと読み解く



W7-3：k値指定

スペースキーを1回押して、次のページに移動。①
まだhash_lengthの説明部分で、「either A or B」の②ORに相当する部分。W10-2で利用しているが、あまり有難みを感じないので講習会ではやりません

②

①

```
File  Edit  View  Search  Terminal  Help
                        : OR: m,M,s where m and M are odd integers (if
not, they will be decremented) with m < M <= 31 (if above, will be redu
ced)
                                    and s is a step (even number). Velvet w
ill then hash from k=m to k=M with a step of s
        filename        : path to sequence file or - for standard input

File format options:
        -fasta  -fastq  -raw    -fasta.gz       -fastq.gz       -raw.gz
        -sam    -bam    -fmtAuto
        (Note: -fmtAuto will detect fasta or fastq, and will try the fo
llowing programs for decompression : gunzip, pbunzip2, bunzip2

File layout options for paired reads (only for fasta and fastq formats)
:
        -interleaved    : File contains paired reads interleaved in the
 one file (default)
        -separate       : Read 2 separate files for paired reads

Read type options:
--More--
```

de novoアセンブリ

省略予定

*de novo*アセンブリとは、リードの塩基配列情報のみを頼りに、元のリード長よりも長い配列（コンティグ）を出力する作業。この例の場合、赤下線が一致部分。出力は、元のリード長よりも2 bp長いコンティグとなる

入力：NGSリードファイル（FASTA/FASTQ）

リード1: CACCAGGACATGAAGACGCG

リード2: CCAGGACATGAAGACGCGTT

出力：コンティグ（FASTA/FASTQ）

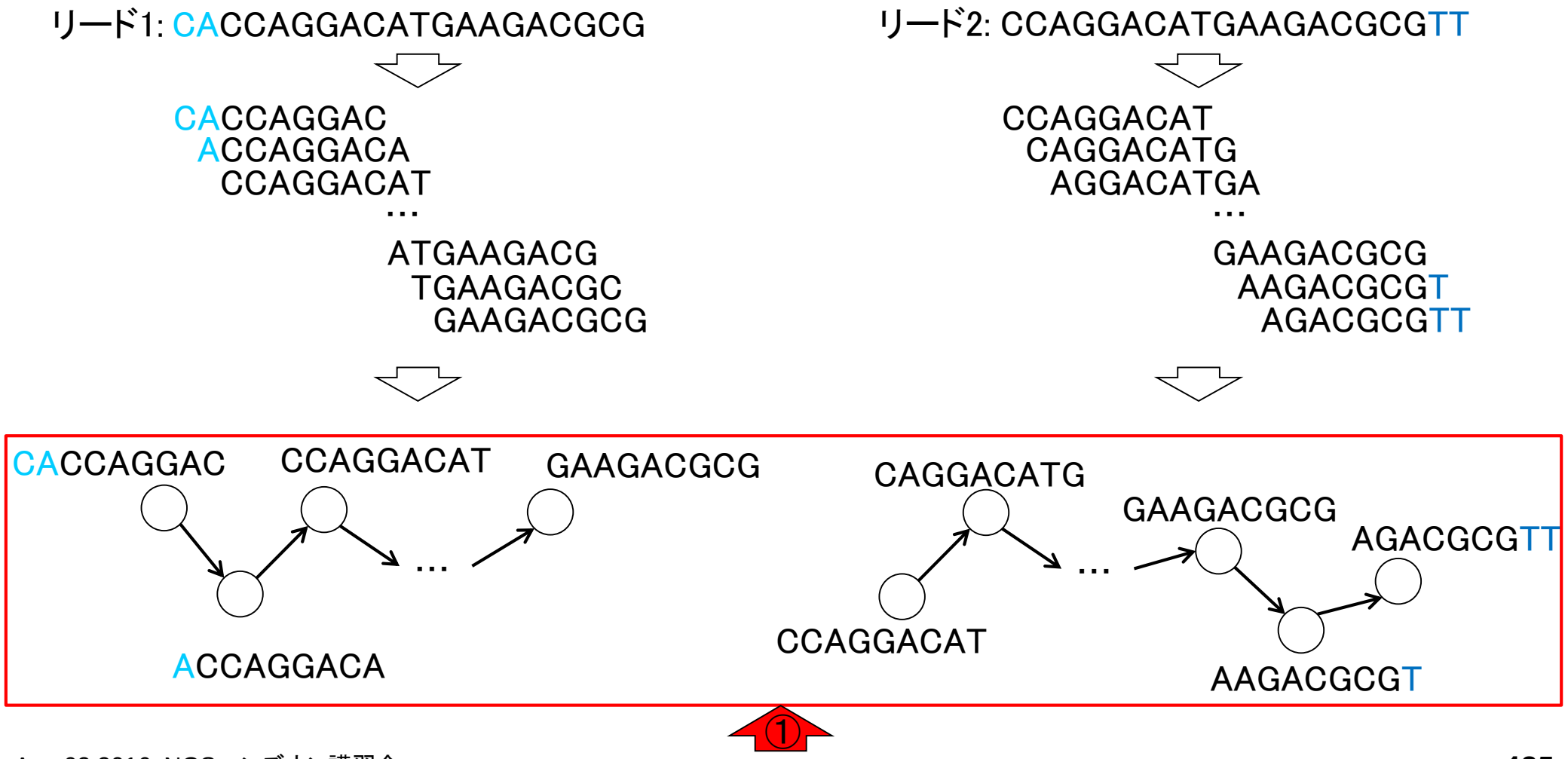
CACCAGGACATGAAGACGCGTT

El-Metwally et al., *PLoS Comput Biol.*, **9**: e1003345, 2013

de novoアセンブリ

省略予定

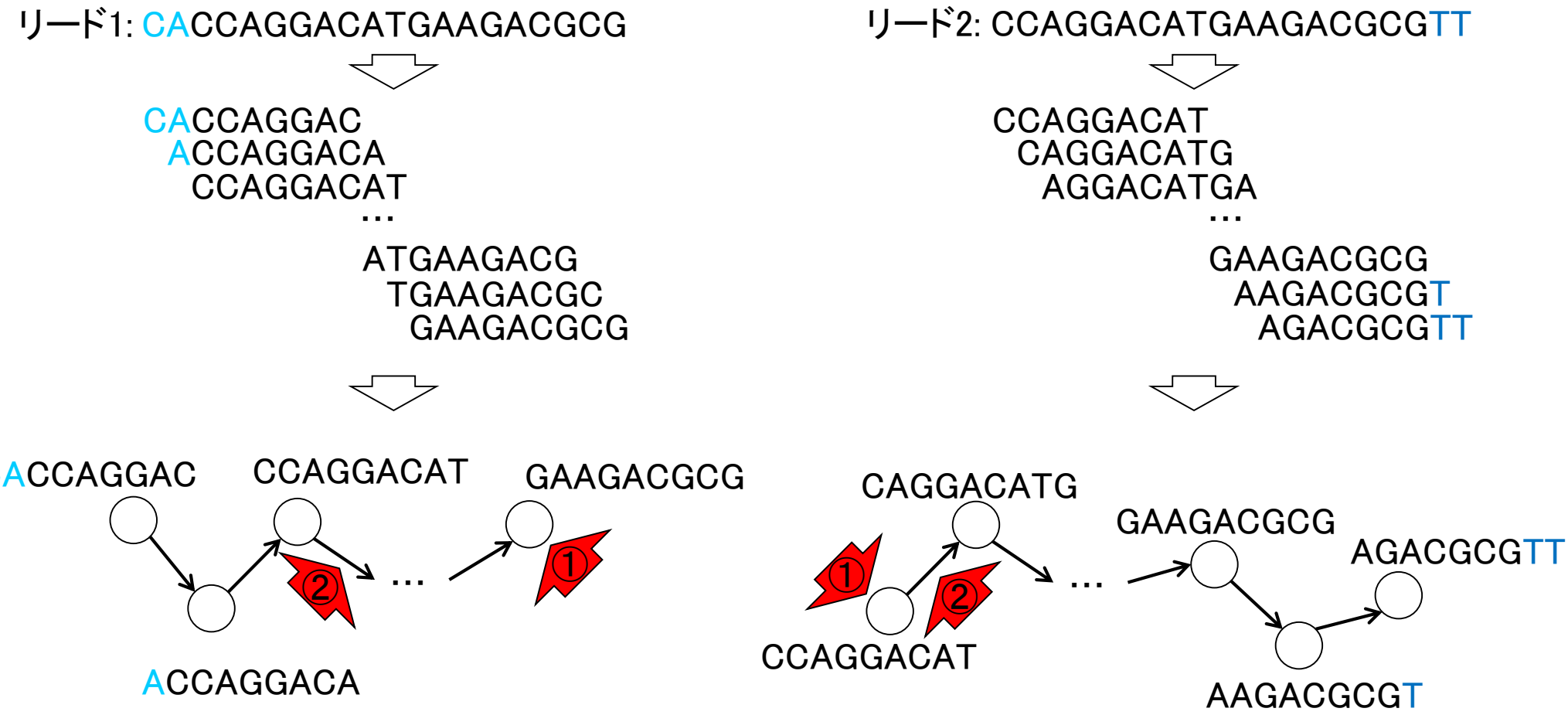
k-merアプローチ(de Bruijnグラフ)の説明。リードを全ての可能なk-merに分割し、①有向グラフを作成(これはk=9の例)。矢印で向きが決まっているので有向(directed)といいます。
私はこっち方面のヒトではないので多少ミスがあるかもしれませんが大筋では大丈夫



①がノード（node；頂点）、②がエッジ（edge；辺）と呼ばれるものです

de novoアセンブリ

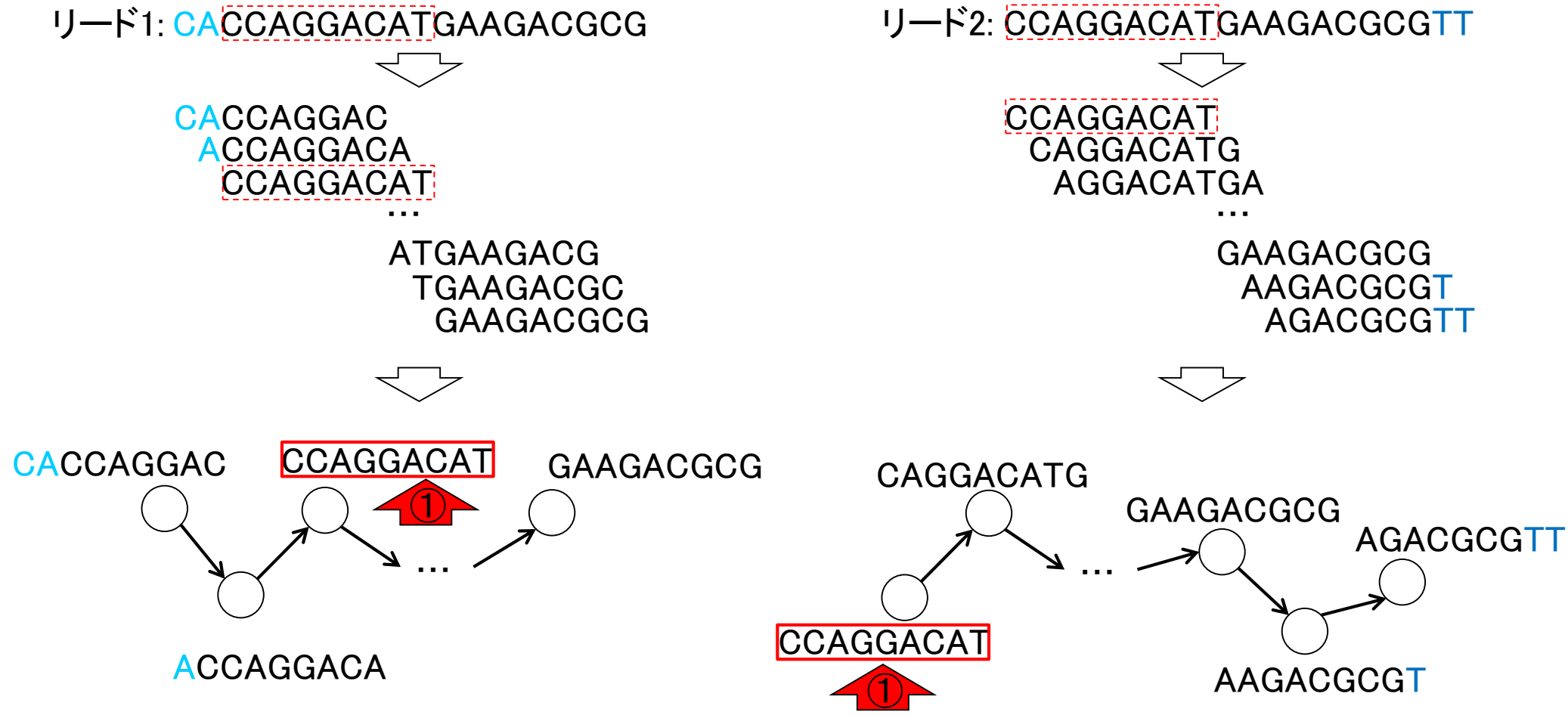
省略予定



de novoアセンブリ

省略予定

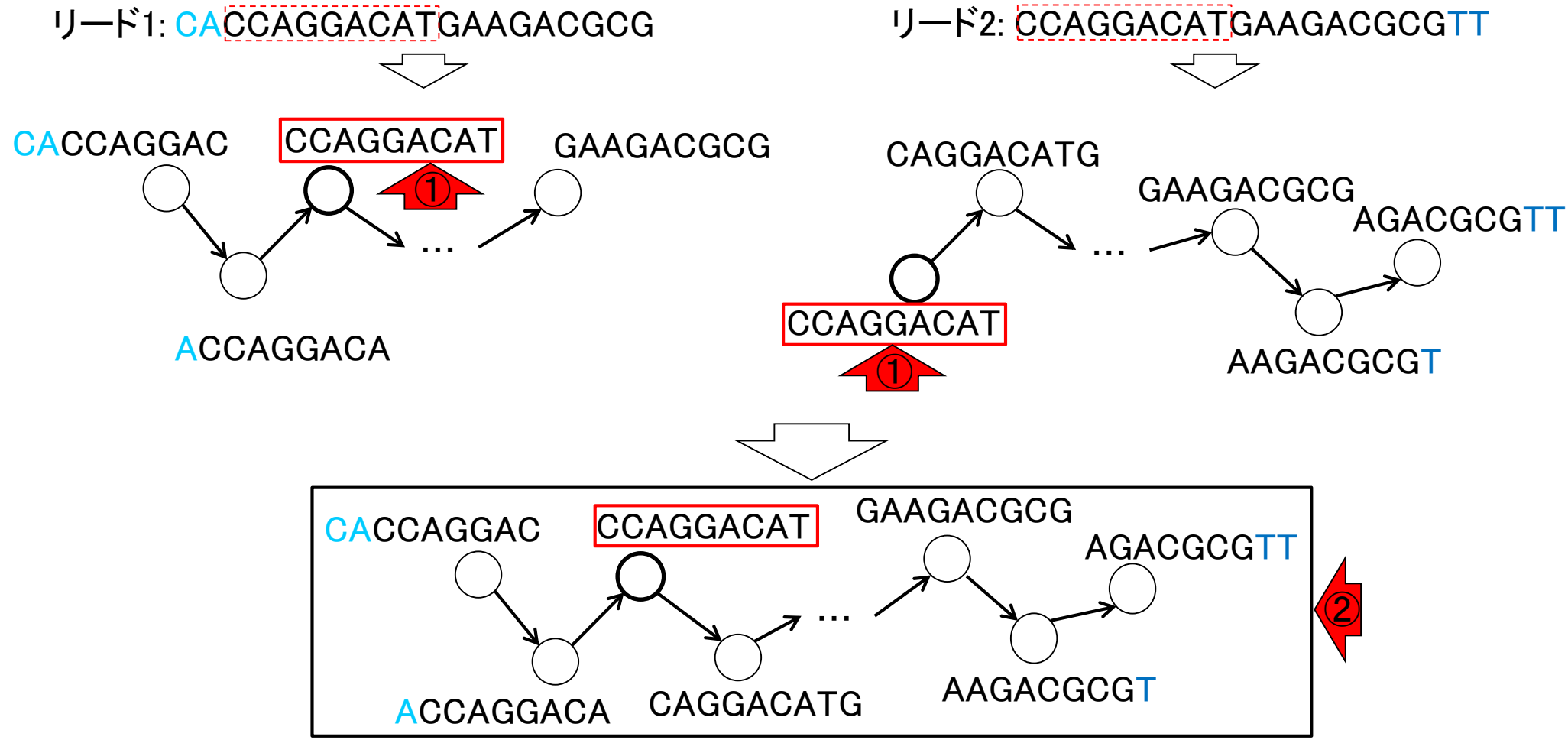
①赤枠のものが（異なるリード間で共有されている）同一ノードの例



de novoアセンブリ

省略予定

②同一ノードをマージしたグラフを作成。これが「k-merグラフ」とか「de Bruijnグラフ」と言われるものです。ここまでがグラフ構築(graph construction)作業



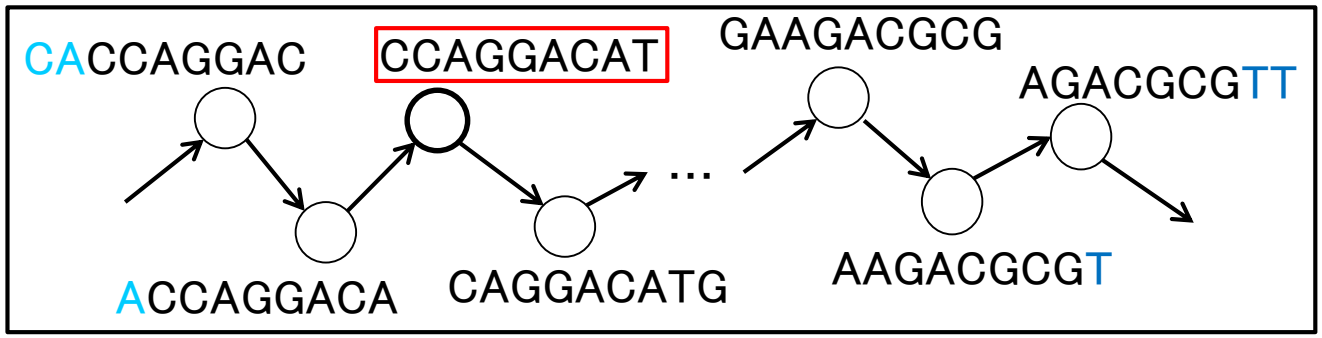
de novoアセンブリ

省略予定

③グラフ簡易化(graph simplification)作業のイメージ。実際には、大量のリードから複雑なde Bruijnグラフが作成されるのでできるだけシンプルにする必要がある。実際に行うのは、ここで示されているような「連続したノード（頂点）」や、次スライドで示す「バブル構造」のマージ

リード1: CACCAGGACATGAAGACGCG

リード2: CCAGGACATGAAGACGCGTT



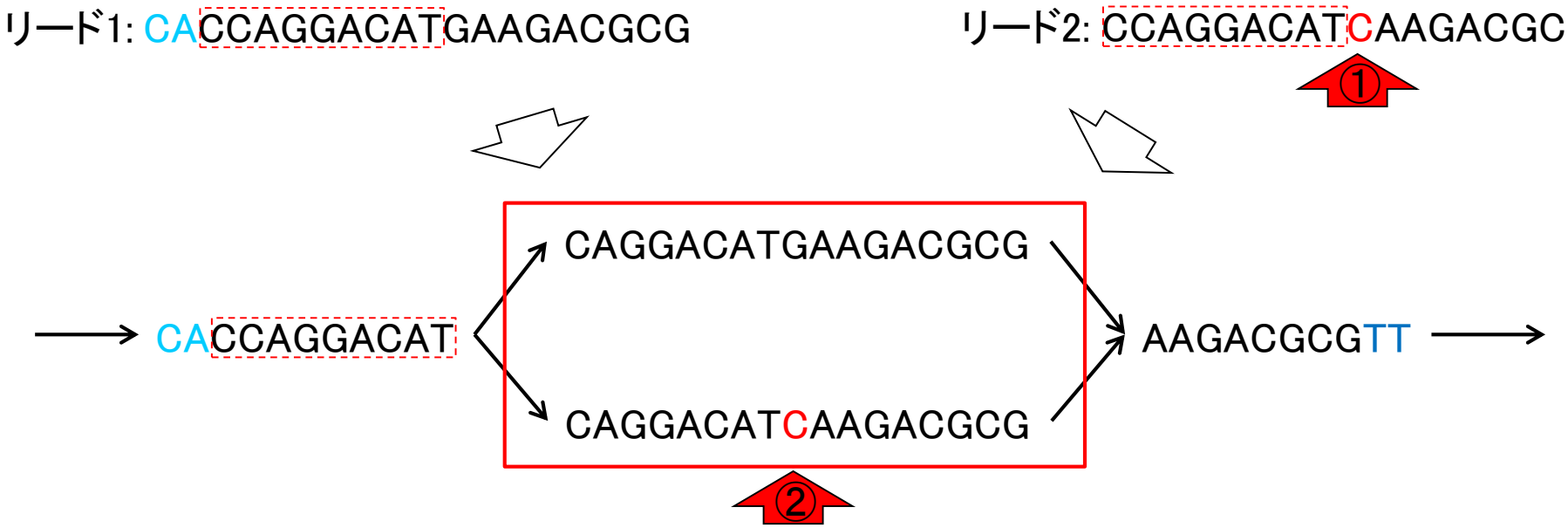
③

CACCAGGACATGAAGACGCGTT

de novoアセンブリ

省略予定

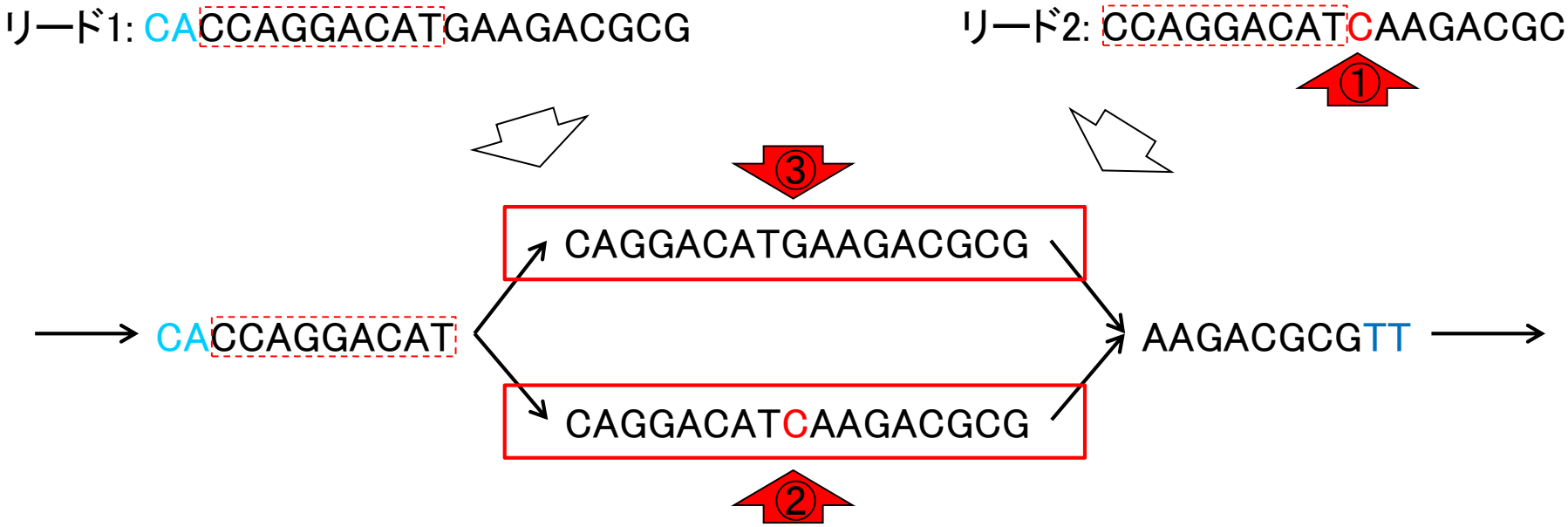
バブル（泡）構造の例。①シークエンスエラーやSNPなど、塩基に違いがあれば②バブル構造になります。エラーが沢山あると、バブルだらけになってグラフ構築や簡易化作業が困難になるであろうことは容易に想像がつきます。クオリティフィルタリングの重要性も分かるでしょう。
このあたりは、2014.06.25の講義資料をベースに作成



de novoアセンブリ

省略予定

通常、②のようなシークエンスエラー由来の部分配列の出現回数は、③のような本物のサンプル由来のものに比べて低い、といった事柄（多数決ルール）をおそらく内部的に利用しているはず



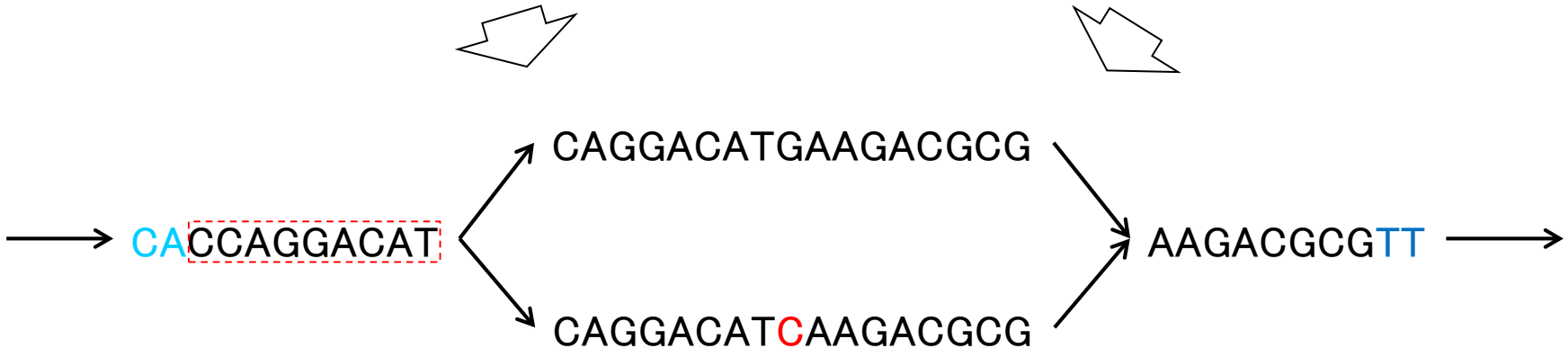
①は、k-mer出現頻度分布（2016.07.20の統計解析の最後のスライド）です

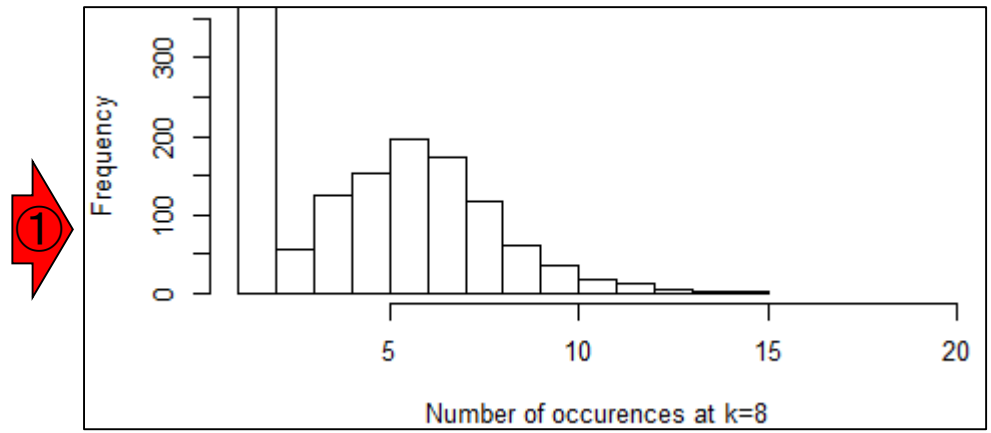
de novoアセンブリ

省略予定

リード1: CACCAGGACATGAAGACGCG

リード2: CCAGGACATCAAGACGCGTT





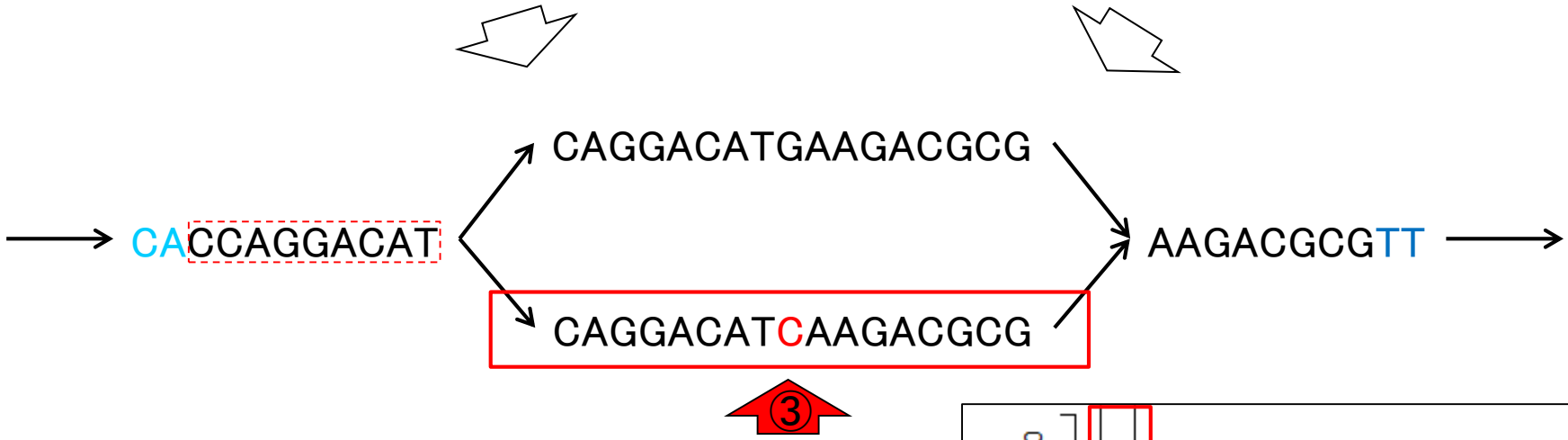
de novoアセンブリ

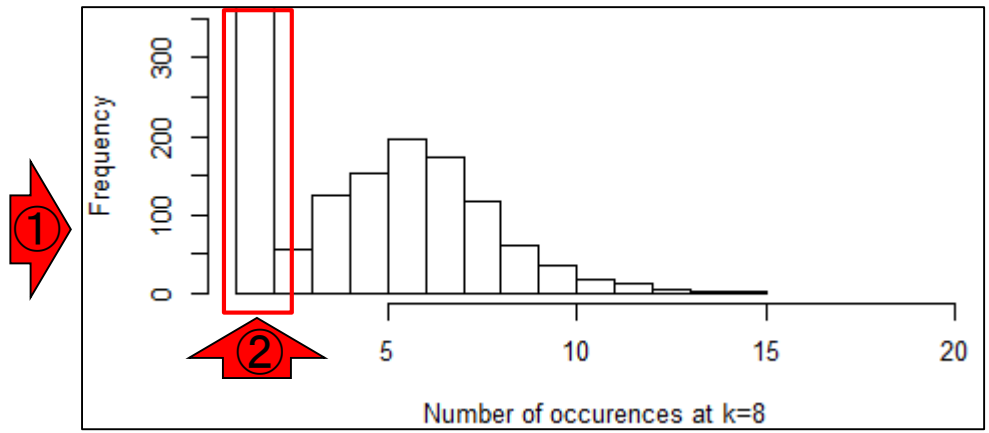
省略予定

②シークエンスエラー由来k-merをフィルタリングしておくことで、バブル構造中の③消されるべきパスがより明瞭になることでしょう

リード1: CACCAGGACATGAAGACGCG

リード2: CCAGGACATCAAGACGCGTT





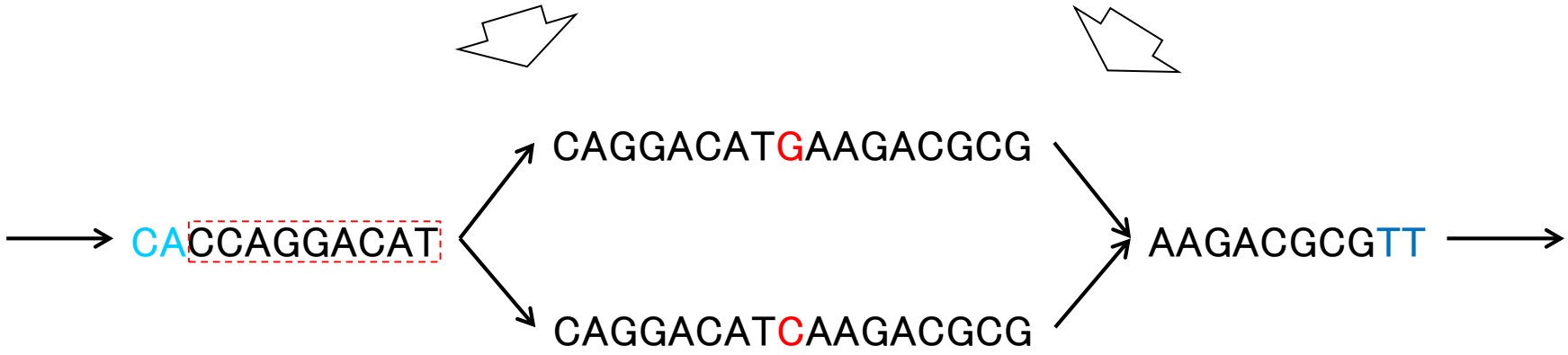
de novoアセンブリ

省略予定

ヘテロ接合度の高い2倍体ゲノム(highly heterozygous diploid genomes)の場合は、出現割合も同程度なのでバブル構造が解けにくい(どちらのパスを残すべきか決めづらい)のですが…

父親由来ゲノム
リード1: CACCAGGACATGAAGACGCG

母親由来ゲノム
リード2: CCAGGACATCAAGACGCGTT



父親由来ゲノム
…CACCAGGACATGAAGACGCGTTCA…
…CACCAGGACATCAAGACGCGTTCA…
母親由来ゲノム

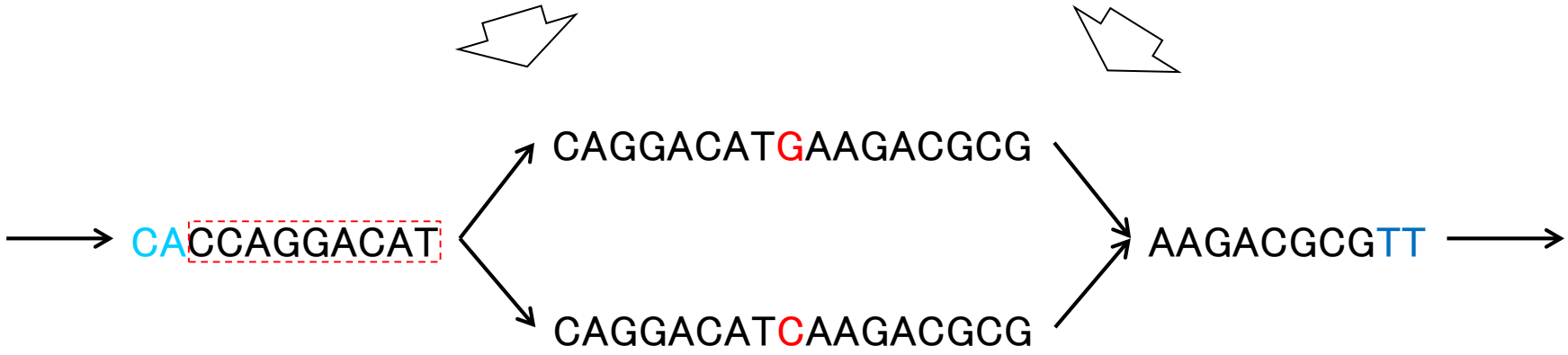
de novoアセンブリ

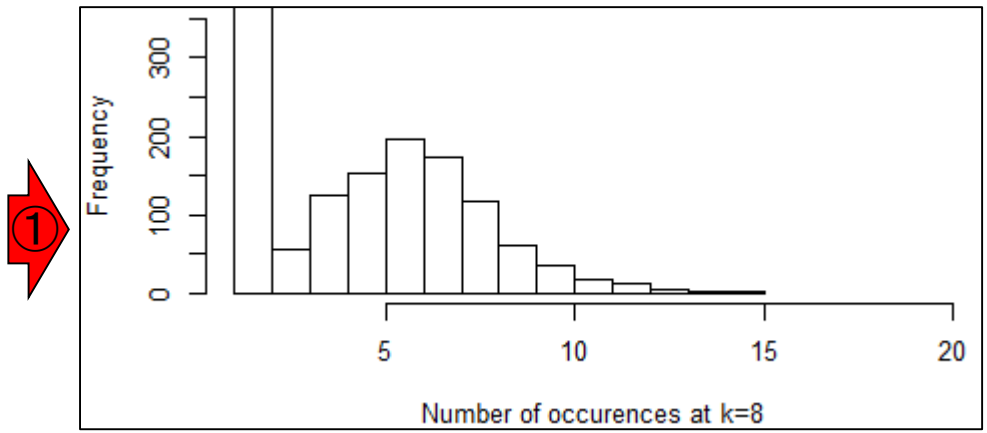
省略予定

ヘテロ接合性（heterozygosity）に関する知見も①のk-mer出現頻度分布を眺めることでわかります

父親由来ゲノム
リード1: CACCAGGACATGAAGACGCG

母親由来ゲノム
リード2: CCAGGACATCAAGACGCGTT





de novoアセンブリ

省略予定

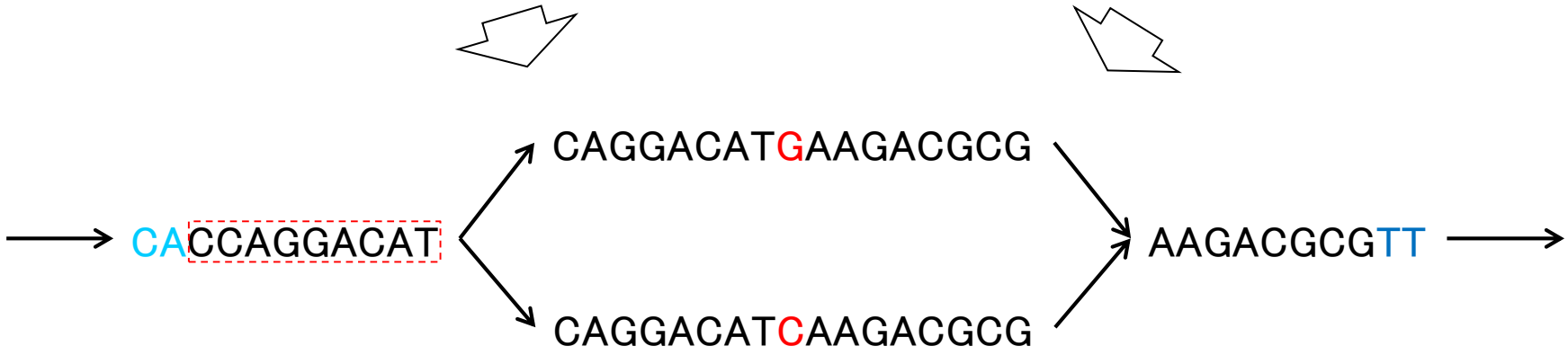
両親間で①配列の異なる領域を含むk-merの出現頻度は、②配列が同じ領域由来k-merの出現頻度(*c*)の1/2となる。この例の場合は、②の分布のみだが、もしheterozygosityがあれば、①の付近にピークのある分布も見られるのです

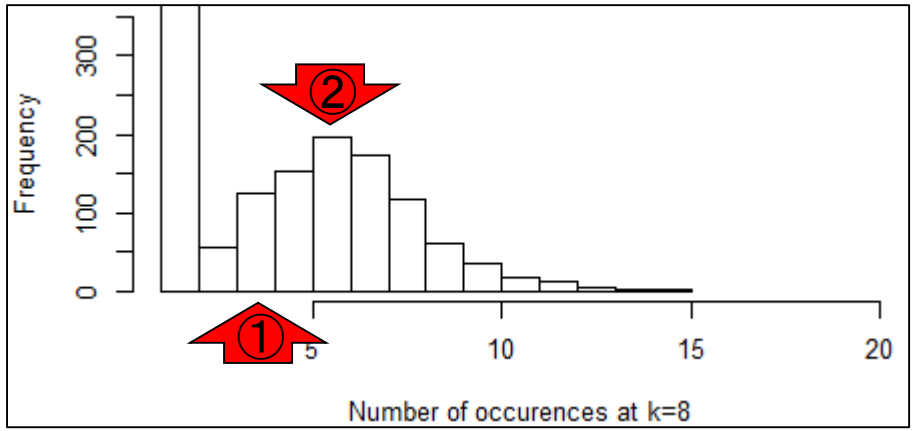
父親由来ゲノム

リード1: CACCAGGACATGAAGACGCG

母親由来ゲノム

リード2: CCAGGACATCAAGACGCGTT





de novoアセンブリ

ヘテロ接合度の高いゲノムもうまく取り扱えるde Bruijnグラフに基づくアセンブラがPlatanus。DDBJ Pipelineにも実装されており、2016.08.03に利用します

省略予定

Genome Res. 2014 Aug;24(8):1384-95. doi: 10.1101/gr.170720.113. Epub 2014 Apr 22.

Efficient de novo assembly of highly heterozygous genomes from whole-genome shotgun short reads.

Kajitani R[1], Toshimoto K[2], Noguchi H[3], Toyoda A[4], Ogura Y[5], Okuno M[1], Yabana M[1], Harada M[1], Nagayasu E[6], Maruyama H[6], Kohara Y[7], Fujiyama A[4], Hayashi T[5], Itoh T[1].

Author information

Abstract

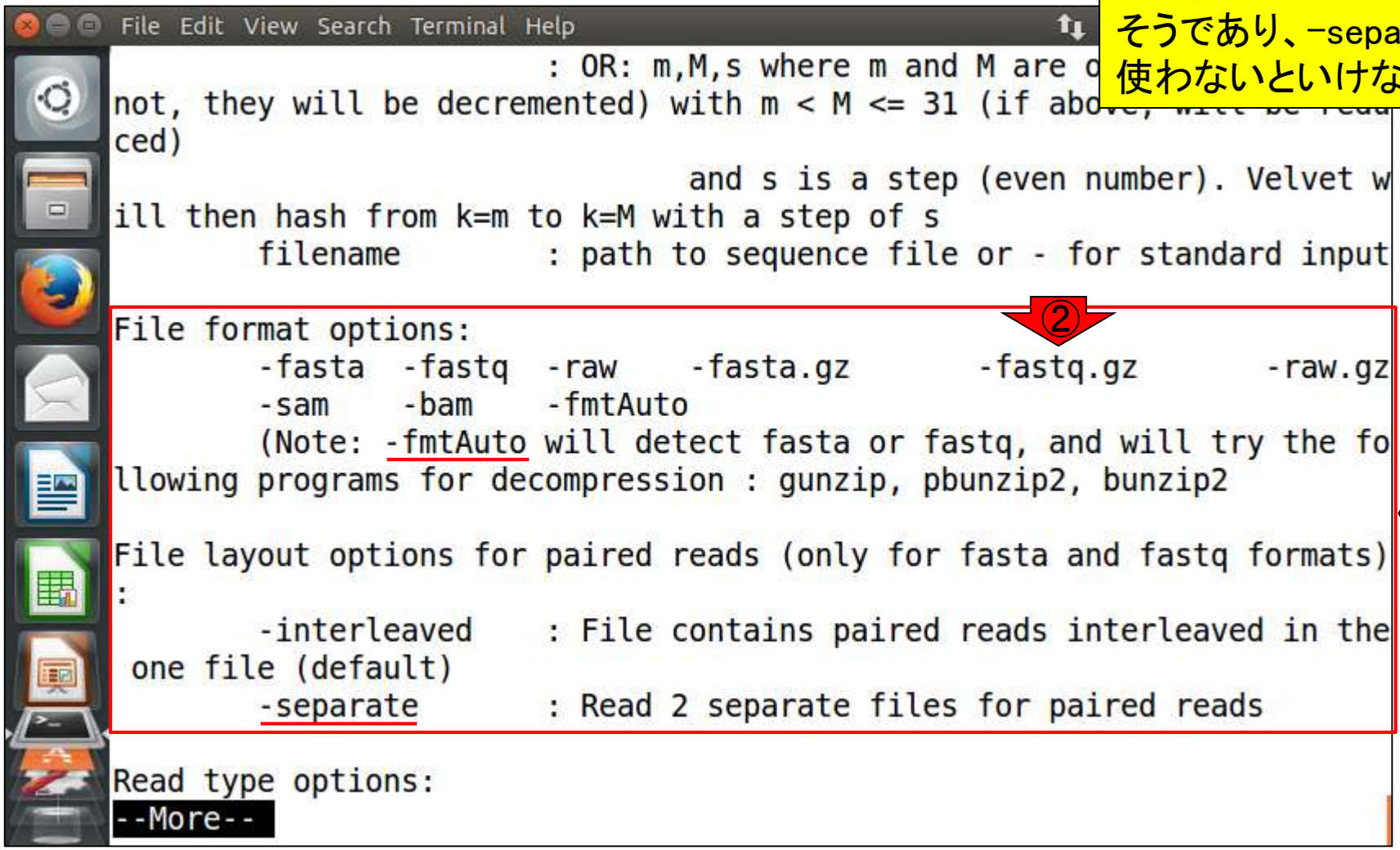
Although many de novo genome assembly projects have recently been conducted using high-throughput sequencers, assembling highly heterozygous diploid genomes is a substantial challenge due to the increased complexity of the de Bruijn graph structure predominantly used. To address the increasing demand for sequencing of nonmodel and/or wild-type samples, in most cases inbred lines or fosmid-based hierarchical sequencing methods are used to overcome such problems. However, these methods are costly and time consuming, forfeiting the advantages of massive parallel sequencing. Here, we describe a novel de novo assembler, Platanus, that can effectively manage high-throughput data from heterozygous samples. Platanus assembles DNA fragments (reads) into contigs by constructing de Bruijn graphs with automatically optimized k-mer sizes followed by the scaffolding of contigs based on paired-end information. The complicated graph structures that result from the heterozygosity are simplified during not only the contig assembly step but also the scaffolding step. We evaluated the assembly results on eukaryotic samples with various levels of heterozygosity. Compared with other assemblers, Platanus yields assembly results that have a larger scaffold NG50 length without any accompanying loss of accuracy in both simulated and real data. In addition, Platanus recorded the largest scaffold NG50 values for two of the three low-heterozygosity species used in the de novo assembly contest, Assemblathon 2. Platanus therefore provides a novel and efficient approach for the assembly of gigabase-sized highly heterozygous genomes and is an attractive alternative to the existing assemblers designed for genomes of lower heterozygosity.

①

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

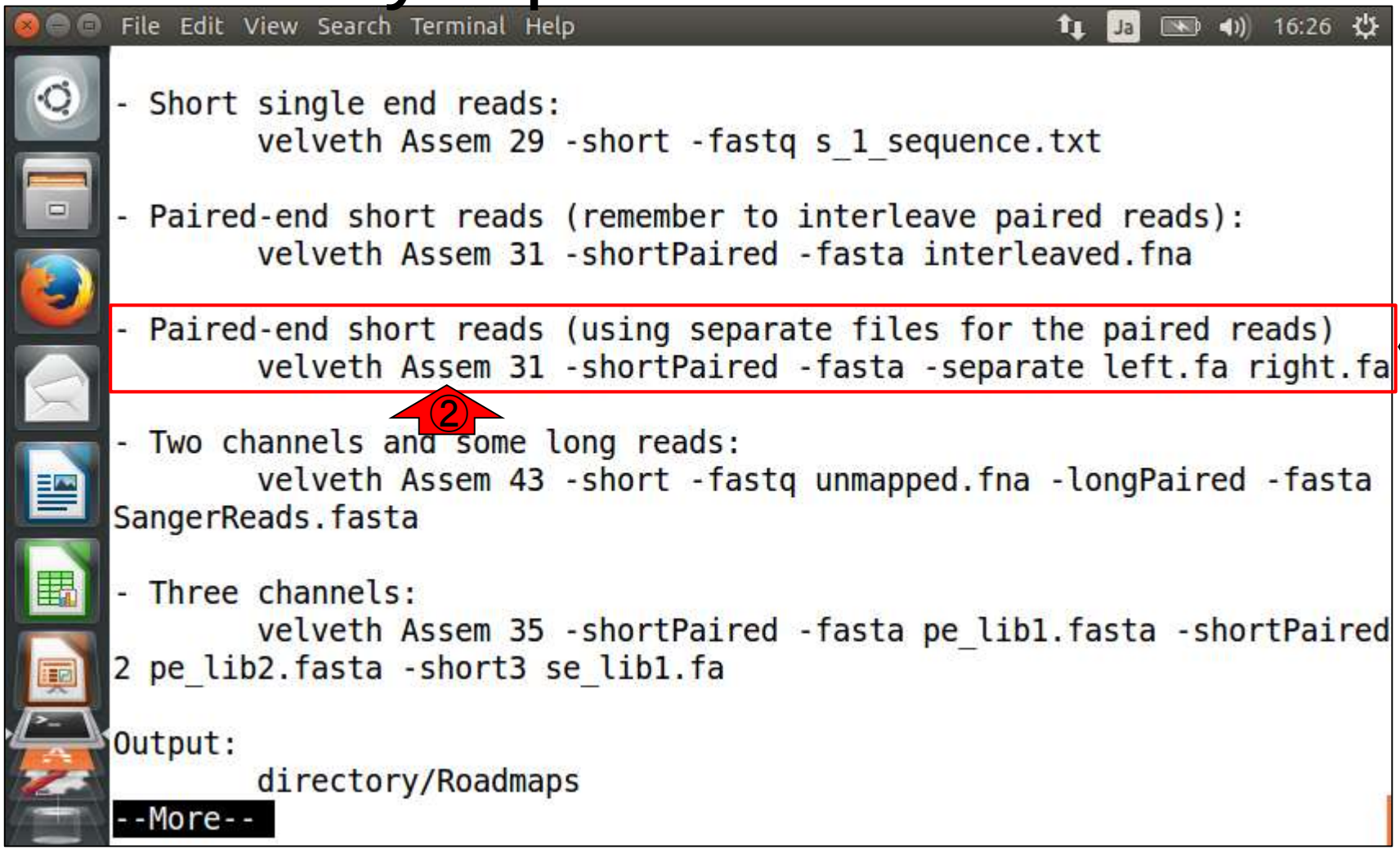
W7-4：入力ファイル形式

①入力ファイル形式の説明部分。
②*.fastq.gzは大丈夫なようだ。他にも、–fmtAutoオプションが便利そうであり、–separateオプションも使わないといけなそうだと学習



W7-5：Synopsis

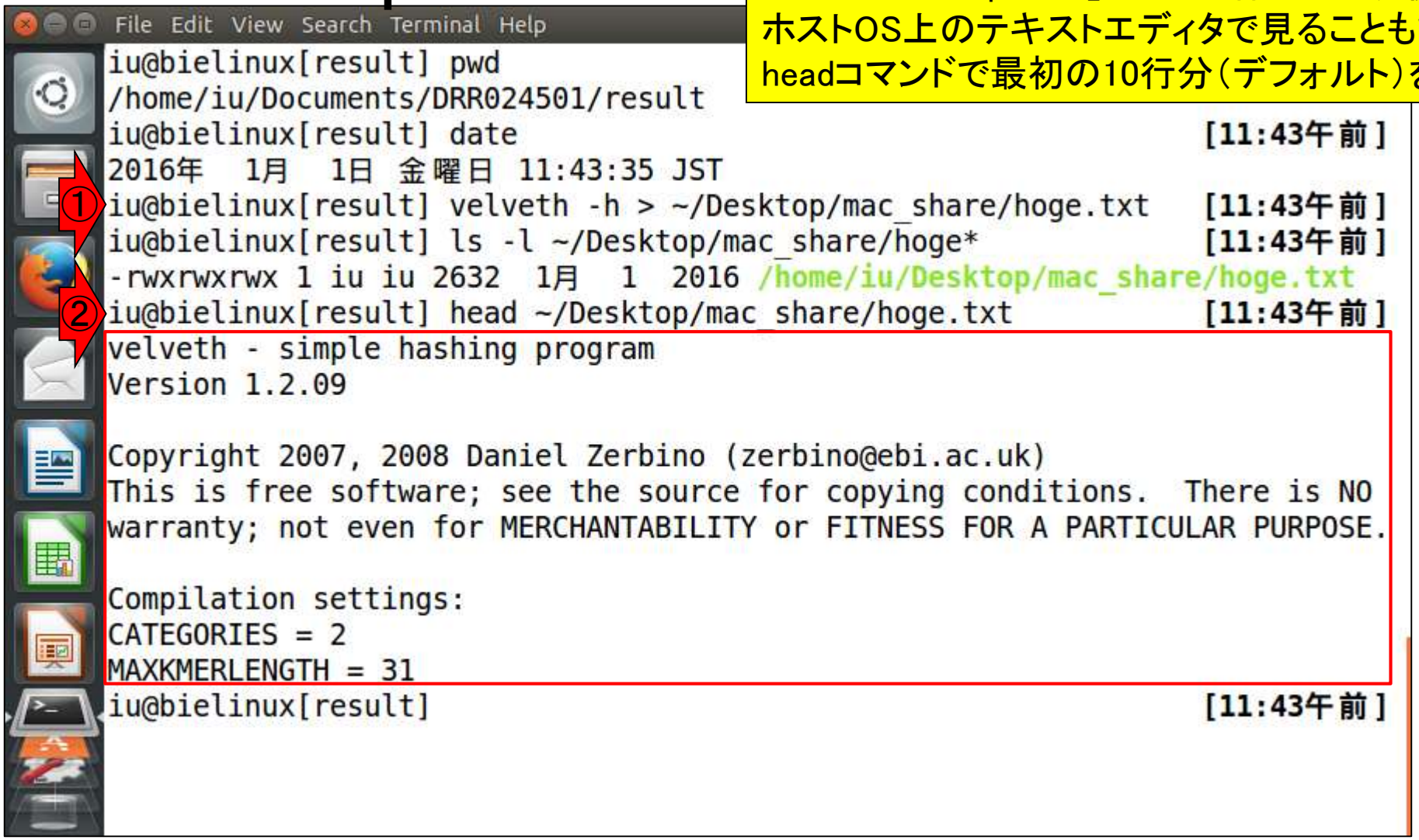
さらに先のページに進んだところ。①この書き方をテンプレートにすればよい。②Assemは出力先のディレクトリ名



参考

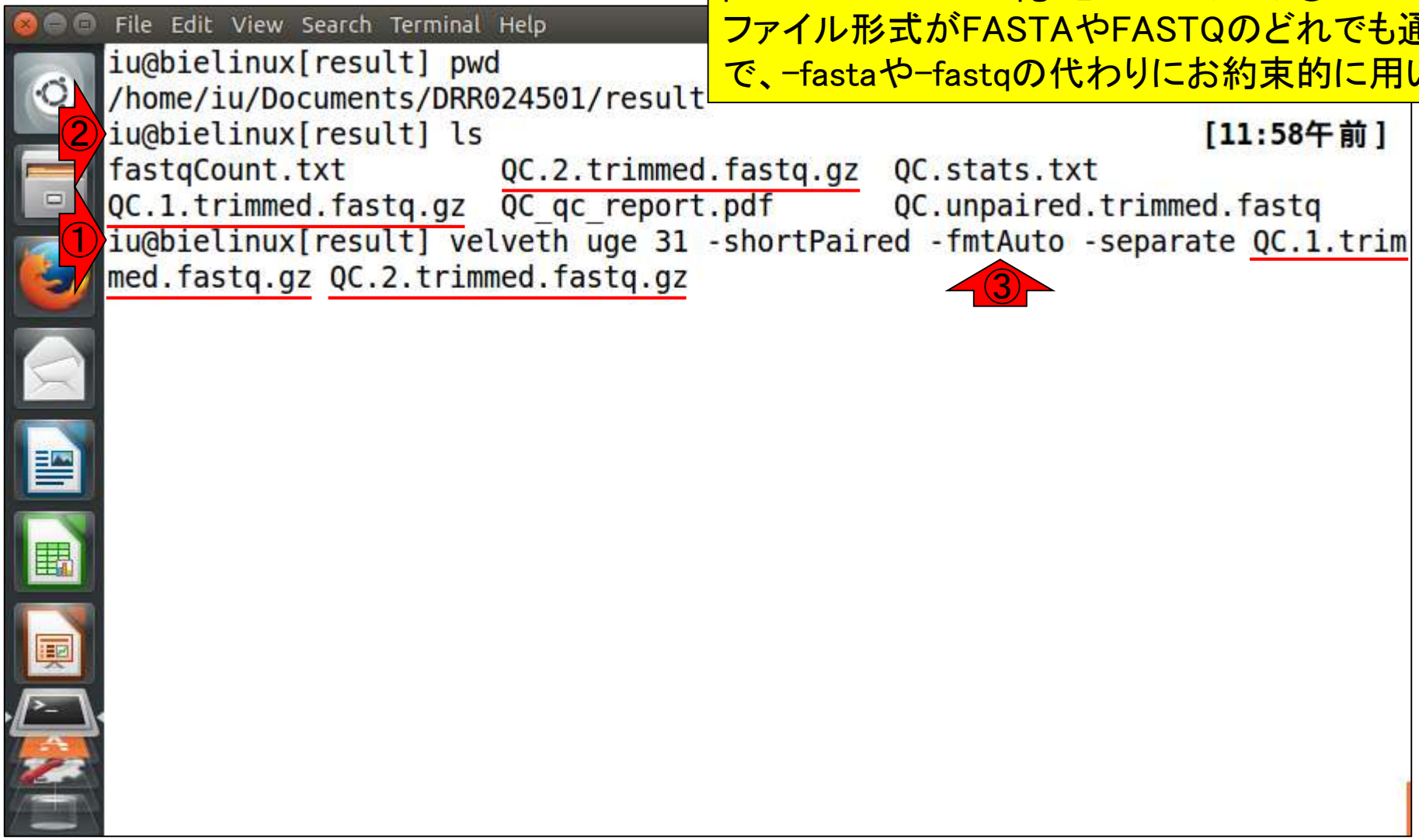
W7-6：Tips

作業ディレクトリはどこでもよい。①velvethのマニュアルをそのまま任意のファイル名（hoge.txt)で共有フォルダ(~/Desktop/mac_share)に保存して、使い慣れたホストOS上のテキストエディタで見ることもできる。②headコマンドで最初の10行分（デフォルト）を表示



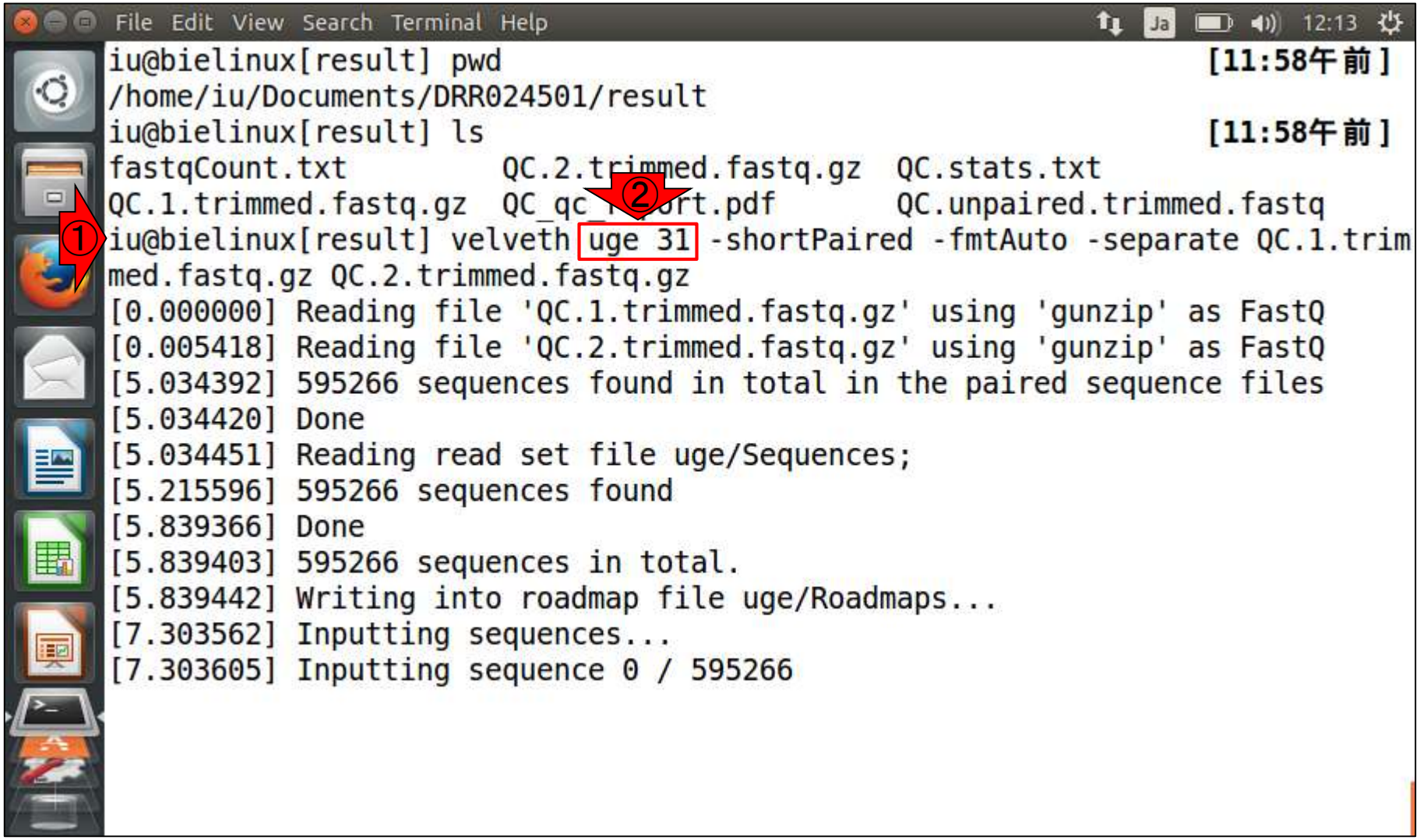
W7-7 : velveth

Velvetは、velvethとvelvetgの2つのコマンドから構成される。まずは①velvethコマンドの実行。②lsで見られるpairedの2つのfastq.gzを入力とする。③–fmtAutoは入力ファイル形式がFASTAやFASTQのどれでも通用するので、–fastaや–fastqの代わりにお約束的に用いてよい



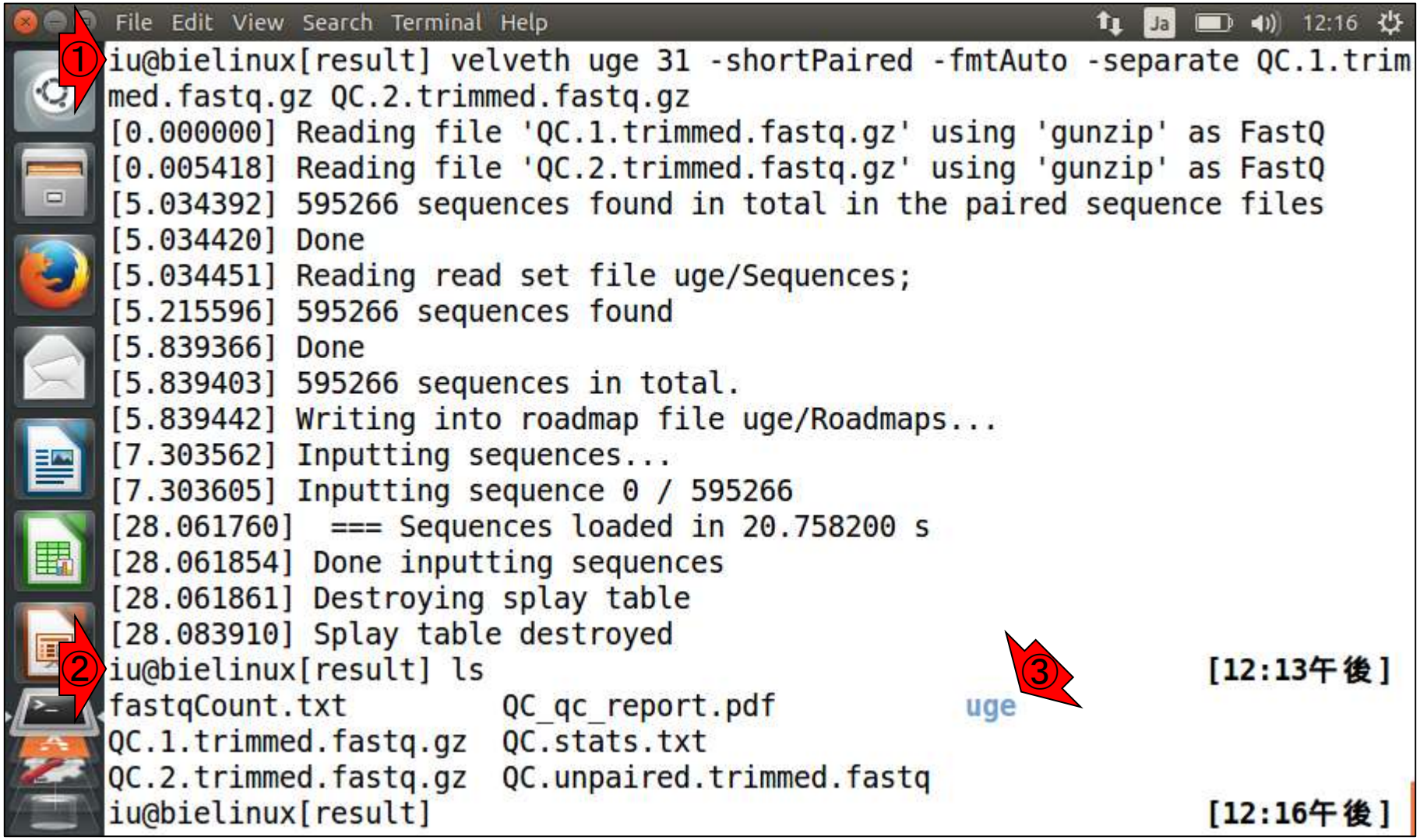
W7-7 : velveth

②k=31で実行し、結果をugeディレクトリに保存。画面はリターンキーを押して数秒後の状態。約1分



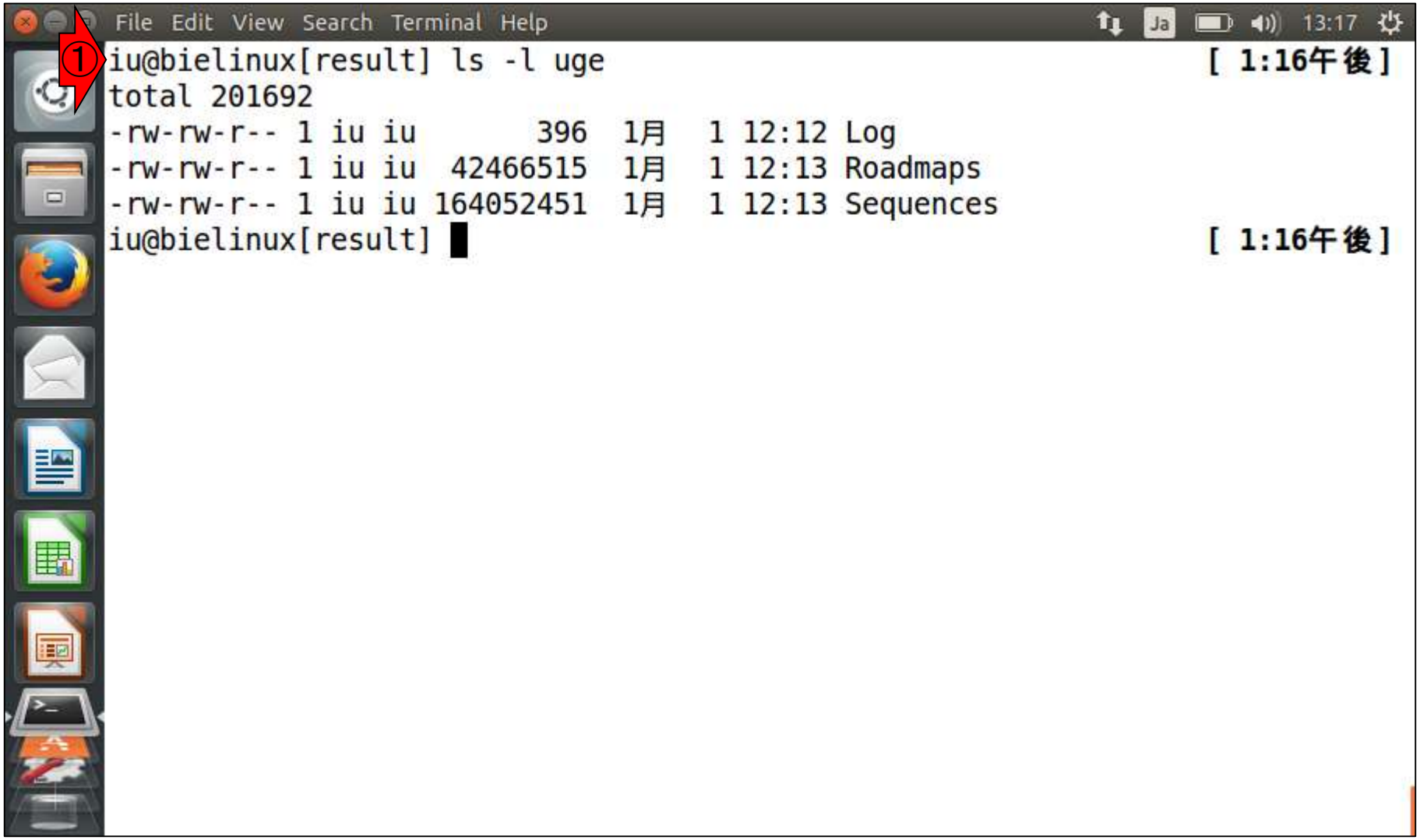
W7-7 : velveth

①velveth実行後の状態。②lsすると、③確かにugeディレクトリが作成されていることがわかる



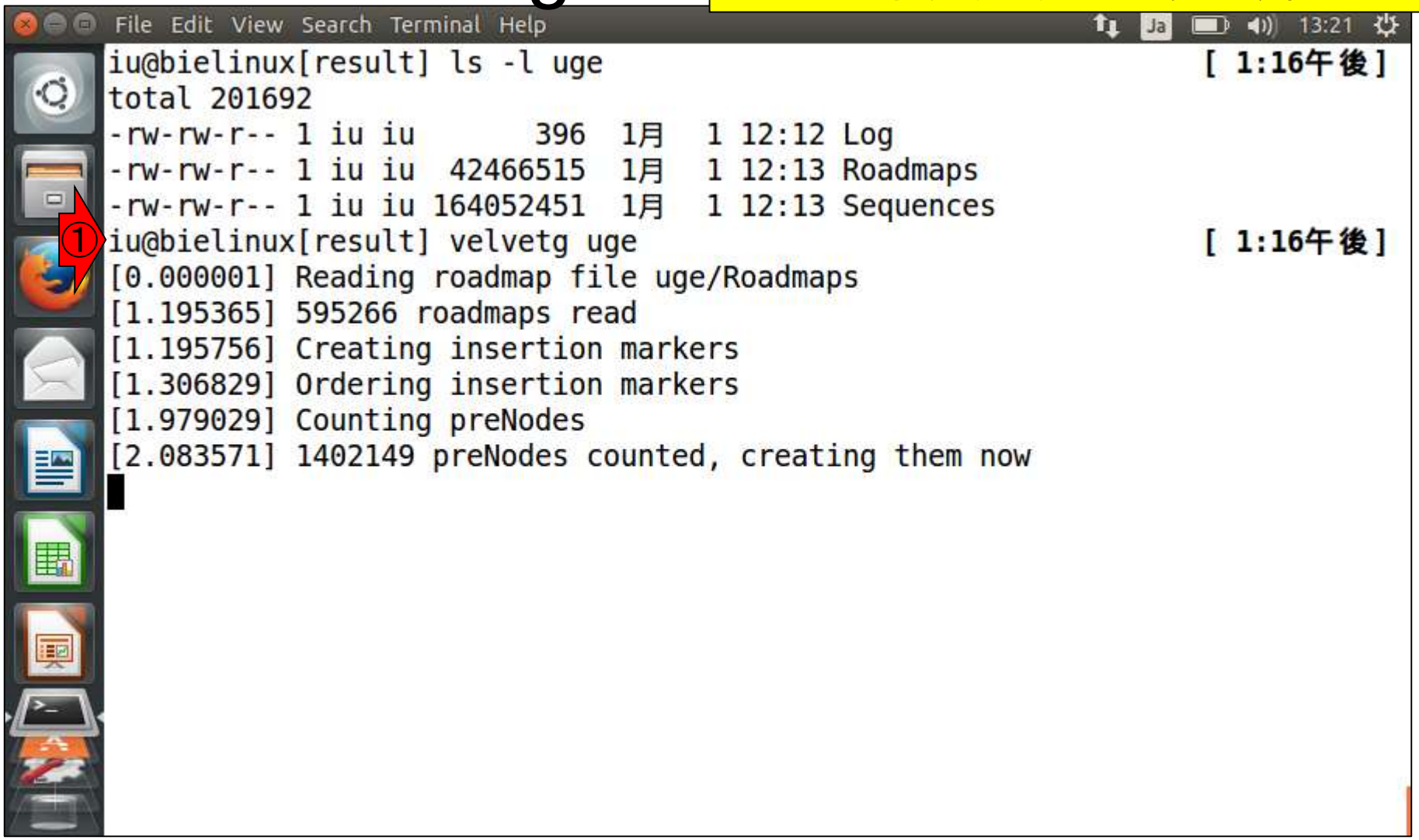
①ugeディレクトリの中身を表示

W7-7 : velveth



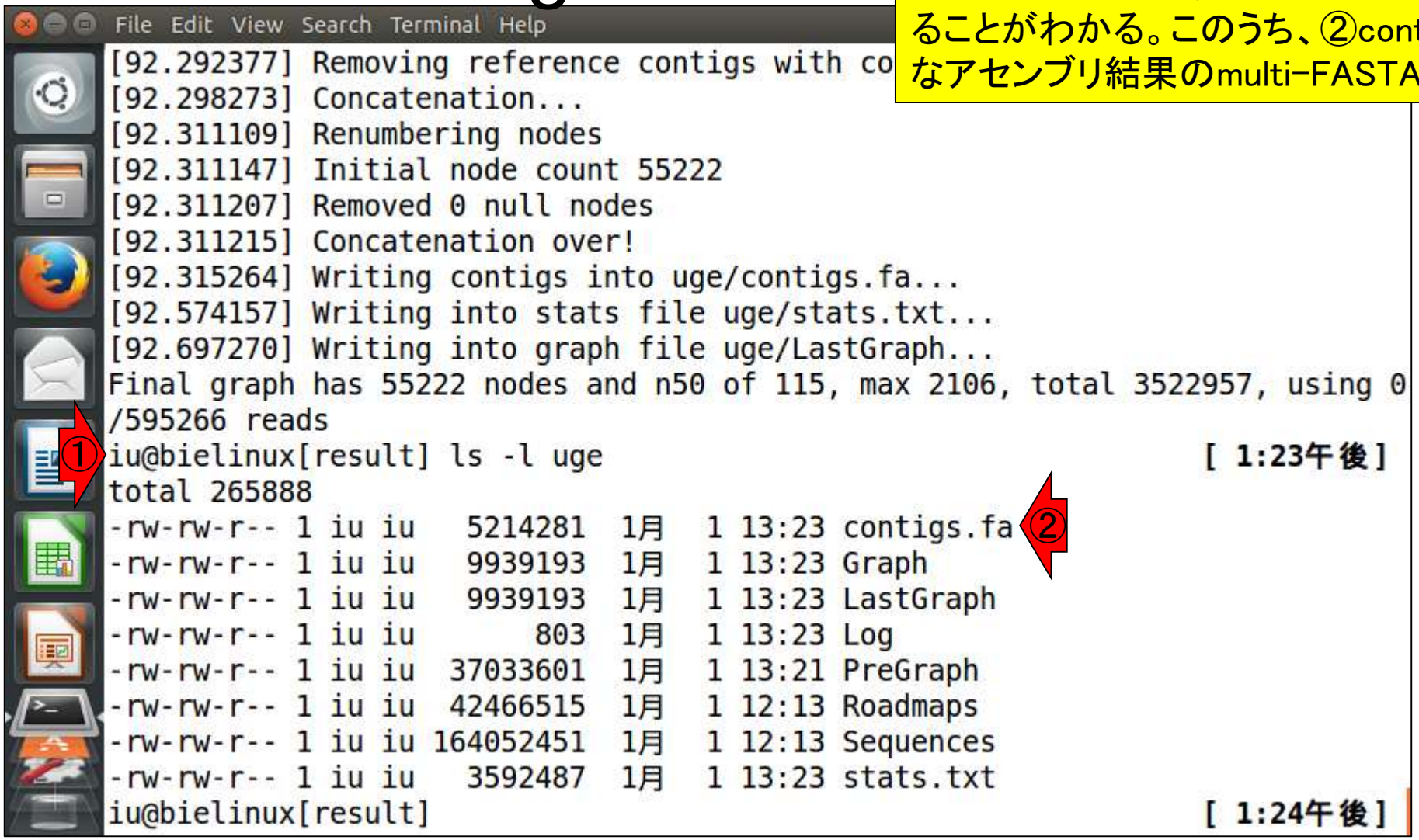
W7-8 : velvetg

Velvetは、velvethとvelvetgの2つのコマンドから構成される。①velvetgは、velvethで作成したugeディレクトリを入力として指定する。画面は実行数秒後の状態。約2分



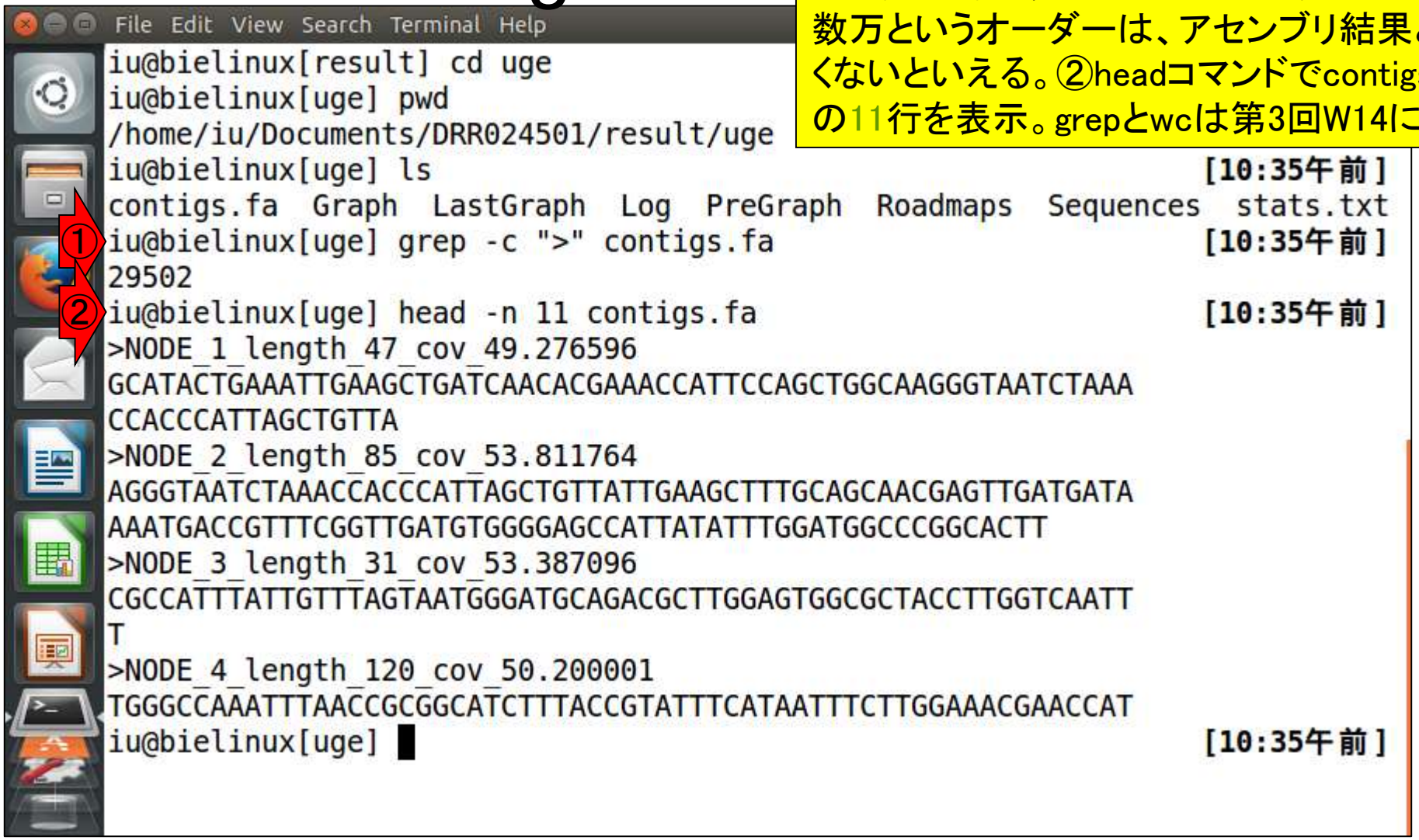
W7-8：velvetg

velvetg終了後の状態。①ugeディレクトリの中身を表示。velvetg実行前は3つのファイルしかなかったが、計8ファイルに増えていることがわかる。このうち、②contigs.faが主なアセンブリ結果のmulti-FASTAファイル



W7-8：velvetg

①配列数をカウント。grepのcオプションは、入力ファイル(contigs.fa)中の”>”を含む行数を出力するというもの。29,502個の配列があることがわかる。数万というオーダーは、アセンブリ結果としてはよくないといえる。②headコマンドでcontigs.faの最初の11行を表示。grepとwcは第3回W14にもあり



Contents

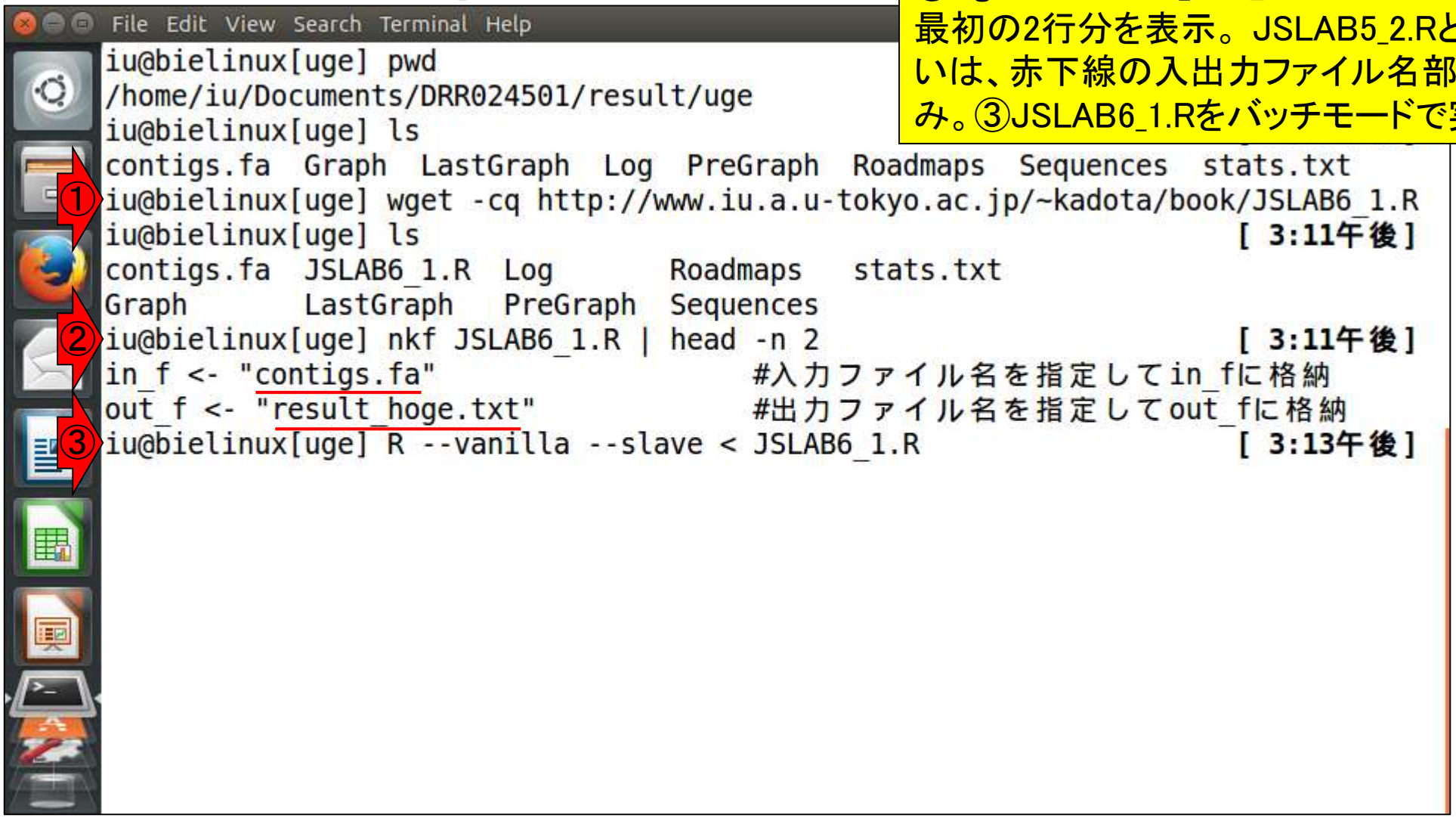
- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

参考

W8-1：Rで解析

状況次第で省略

第5回W12-3 (2016.08.01のスライド165)と同じ解析をcontigs.faを入力として実行。①wgetでJSLAB6_1.Rをダウンロード。②最初の2行分を表示。JSLAB5_2.Rとの違いは、赤下線の入出力ファイル名部分のみ。③JSLAB6_1.Rをバッチモードで実行

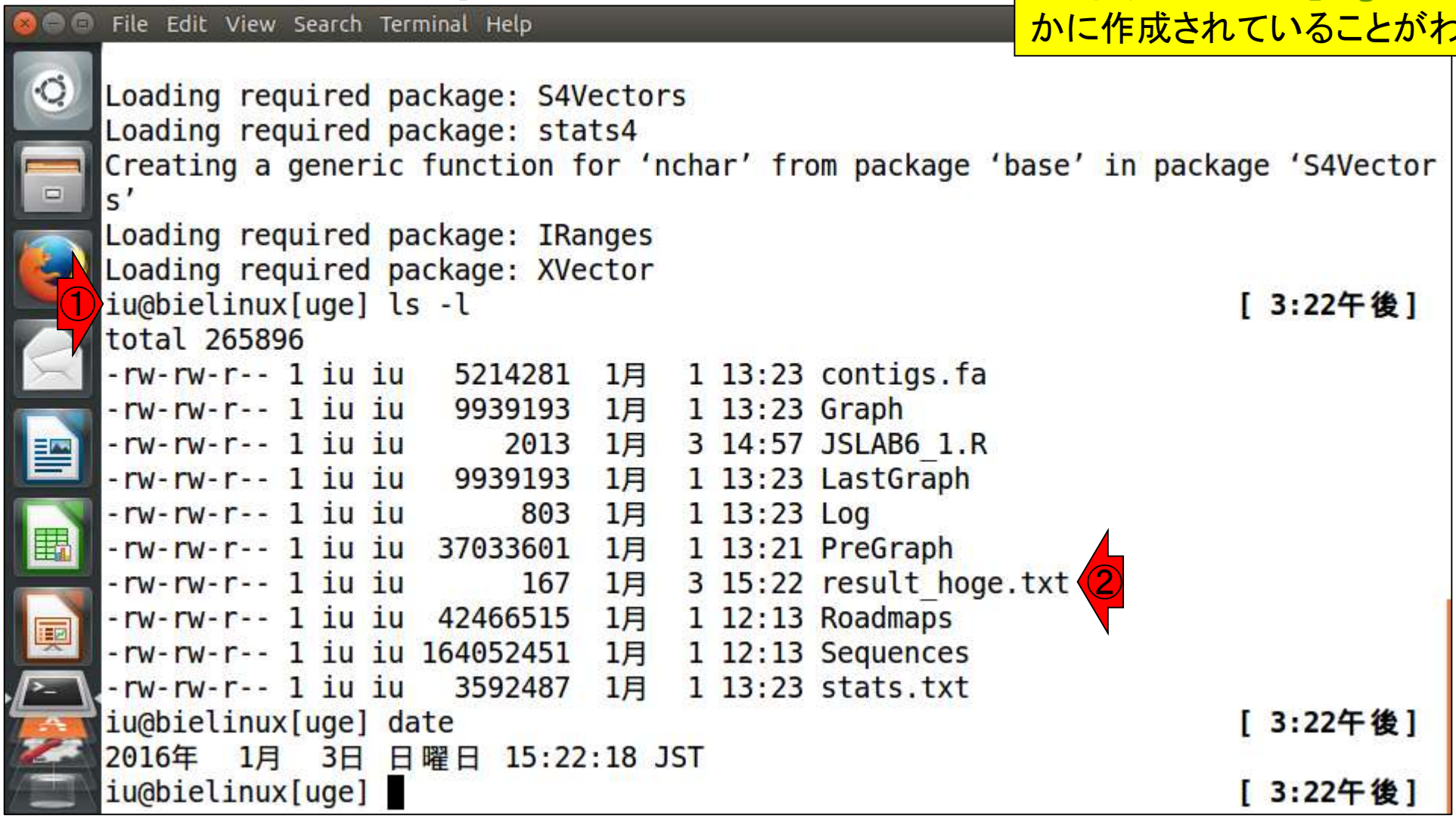


参考

W8-1：Rで解析

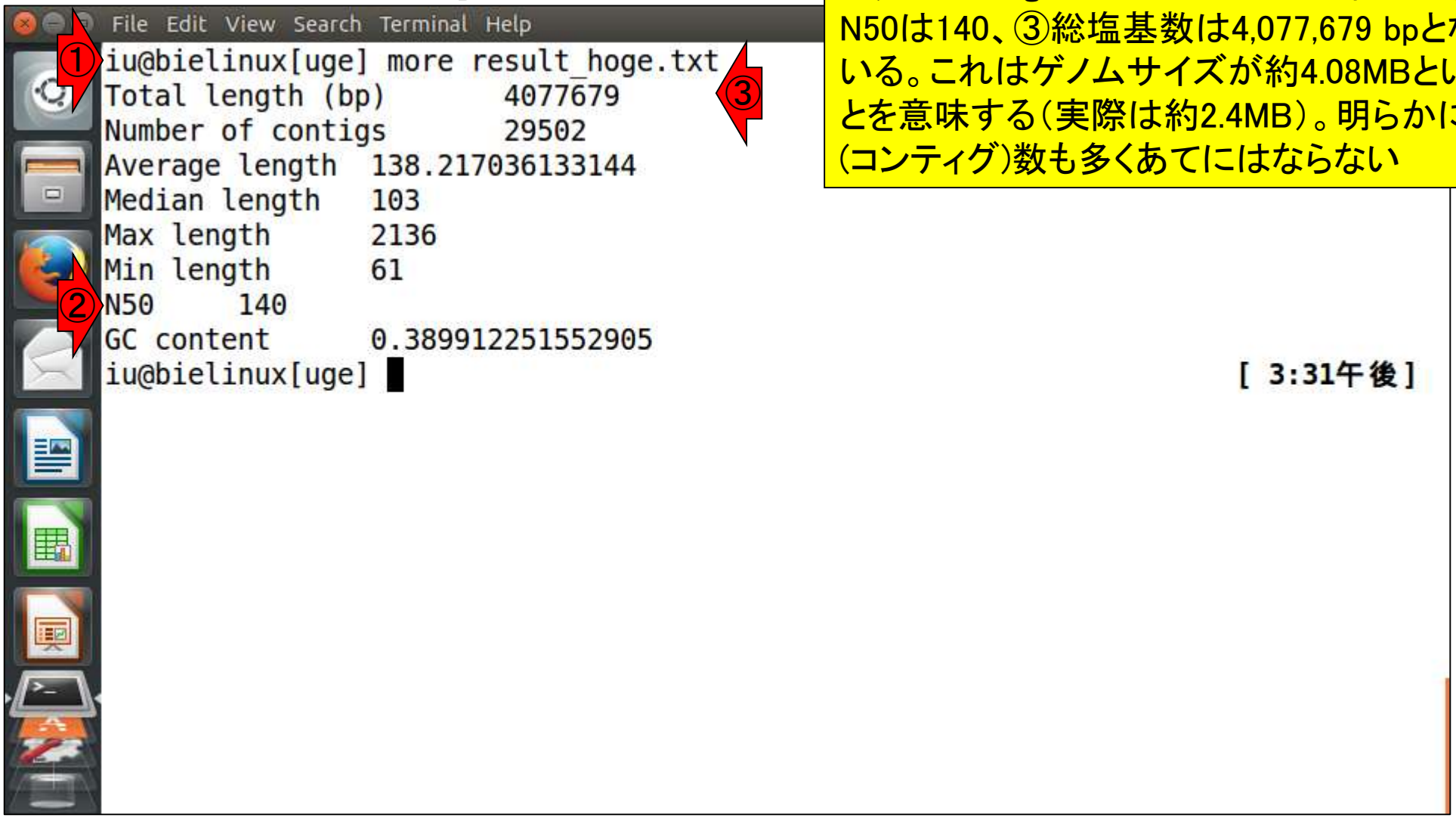
状況次第で省略

計算自体は数秒。①lsで確認。②JSLAB6_1.R上で出力ファイル名として指定したresult_hoge.txtが確かに作成されていることがわかる



参考 状況次第で

W8-1：Rで解析



①result_hoge.txtの中身をmoreで表示。W7-8の「grep -c “>” contigs.fa」の結果と同じく、29,502 contigsになっていることがわかる。②N50は140、③総塩基数は4,077,679 bpとなっている。これはゲノムサイズが約4.08MBということを意味する（実際は約2.4MB）。明らかに配列（コンティグ）数も多くあてにはならない

参考

W8-1：コードの中身は…

JSLAB6_1.Rの中身は、①の例題1と基本的に同じ。違いは③赤枠の入出力ファイル名部分のみ。スライドを見るだけ

状況次第で省略

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～

(last modified 2015/12/22, since 2011)

What's new?

- このウェブペ
  Rと必要なパ
  (Windows20
  (2015/04/03)
- 多群間比較

- イントロ | NGS | アノテーション情報取得 | TranscriptDb | GenomicFeatures(Lawrence 2013) (last modified 2015/02/19)
- イントロ | NGS | アノテーション情報取得 | TranscriptDb | GFF/GTF形式ファイルから (last modified 2015/03/04)
- イントロ | NGS | 読み込み | BSgenome | 基本情報を取得 (last modified 2015/09/12)
- イントロ | NGS | 読み込み | FASTA形式 | 基本情報を取得 (last modified 2015/09/12)
- イントロ | NGS | 読み込み | FASTA形式 | description行の記述を整形 (last modified 2014/04/05)
- イントロ | NGS | 読み込み | FASTQ形式 | 基礎 (last modified 2015/07/26)
- イントロ | NGS | 読み込み | FASTQ形式 | 応用 (last modified 2015/06/18)
- イントロ | NGS | 読み込み | FASTQ形式 | description行の記述を整形 (last modified 2014/08/21)
- イントロ | NGS | 読み込み | Illuminaの*_seq.txt (last modified 2013/06/13)
- イントロ | NGS | 読み込み | Illumina
- イントロ | ファイル形式の変換 | につ
- イントロ | ファイル形式の変換 | BAM

① ②

イントロ | NGS | 読み込み | FASTA形式 | 基本情報を取得

multi-FASTAファイルを読み込んで、Total lengthやaverage lengthなどの各種情報取得を行うためのやり方を示します。例題6以降は、ヒトやマウスレベルの巨大ファイルを取り扱うためのコードです。具体的には、塩基数を整数(integer)ではなく実数(real number)として取り扱うためのas.numeric関数を追加しています。
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し以下をコピペ。

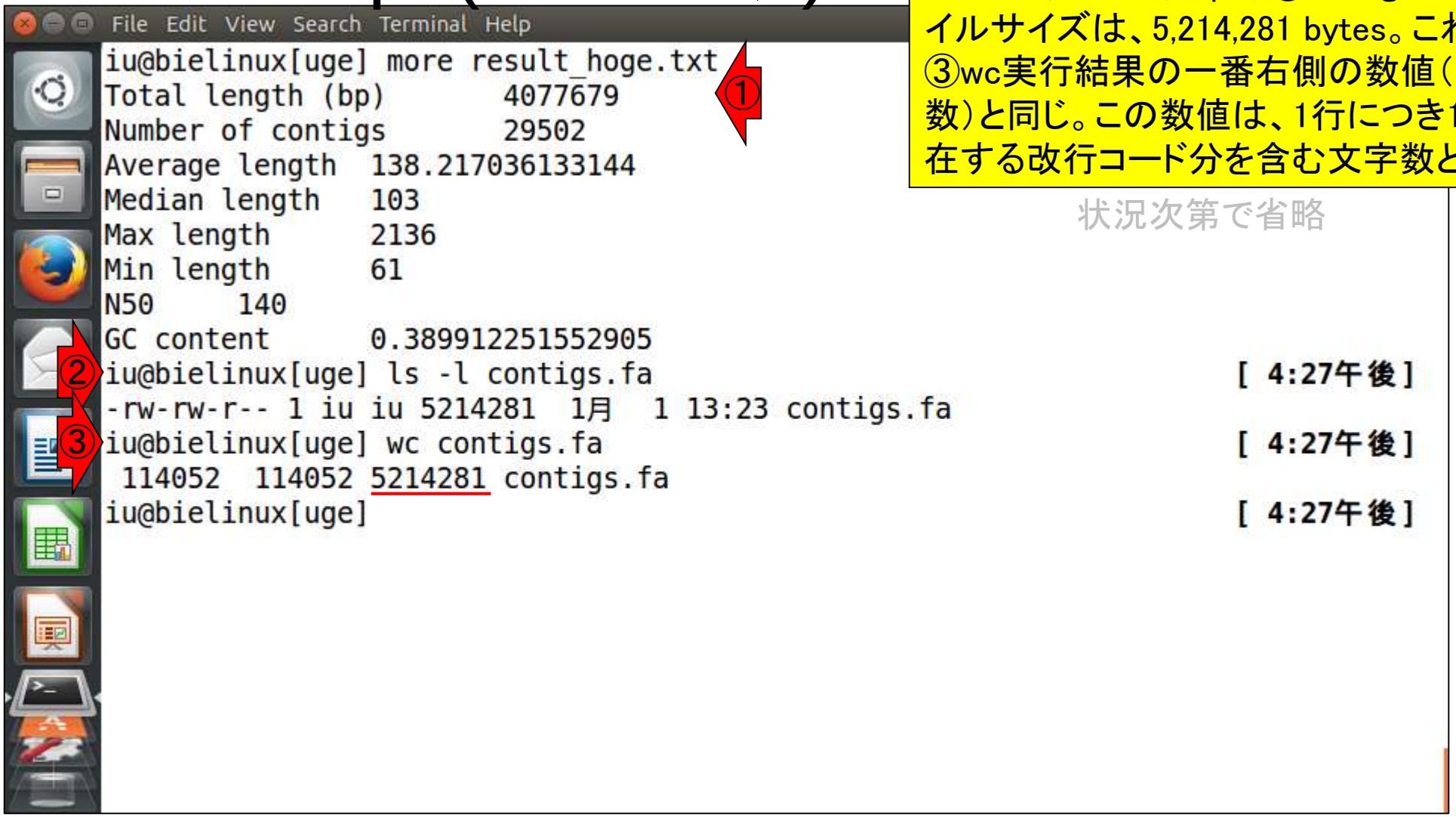
1. イントロ | 一般 | ランダムな塩基配列を作成の4.を実行して得られたmulti-FASTAファイル(hoge4.fa)の場合:

③

```
in_f <- "hoge4.fa"                    #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.txt"                  #出力ファイル名を指定してout_fに格納

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

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_f, format="fasta")#in_fで指定したファイルの読み込み

#本番(基本情報取得)
Total_len <- sum(width(fasta))        #配列の「トータルの長さ」を取得
Number_of_contigs <- length(fasta)    #「配列数」を取得
```

参考

W8-2：Tips(総塩基数)



①総塩基数(4,077,679 bp)は、contigs.fa中のdescription行を除いたものから把握可能であるというTips。②contigs.faのファイルサイズは、5,214,281 bytes。これは③wc実行結果の一番右側の数値（バイト数）と同じ。この数値は、1行につき1つ存在する改行コード分を含む文字数と同じ

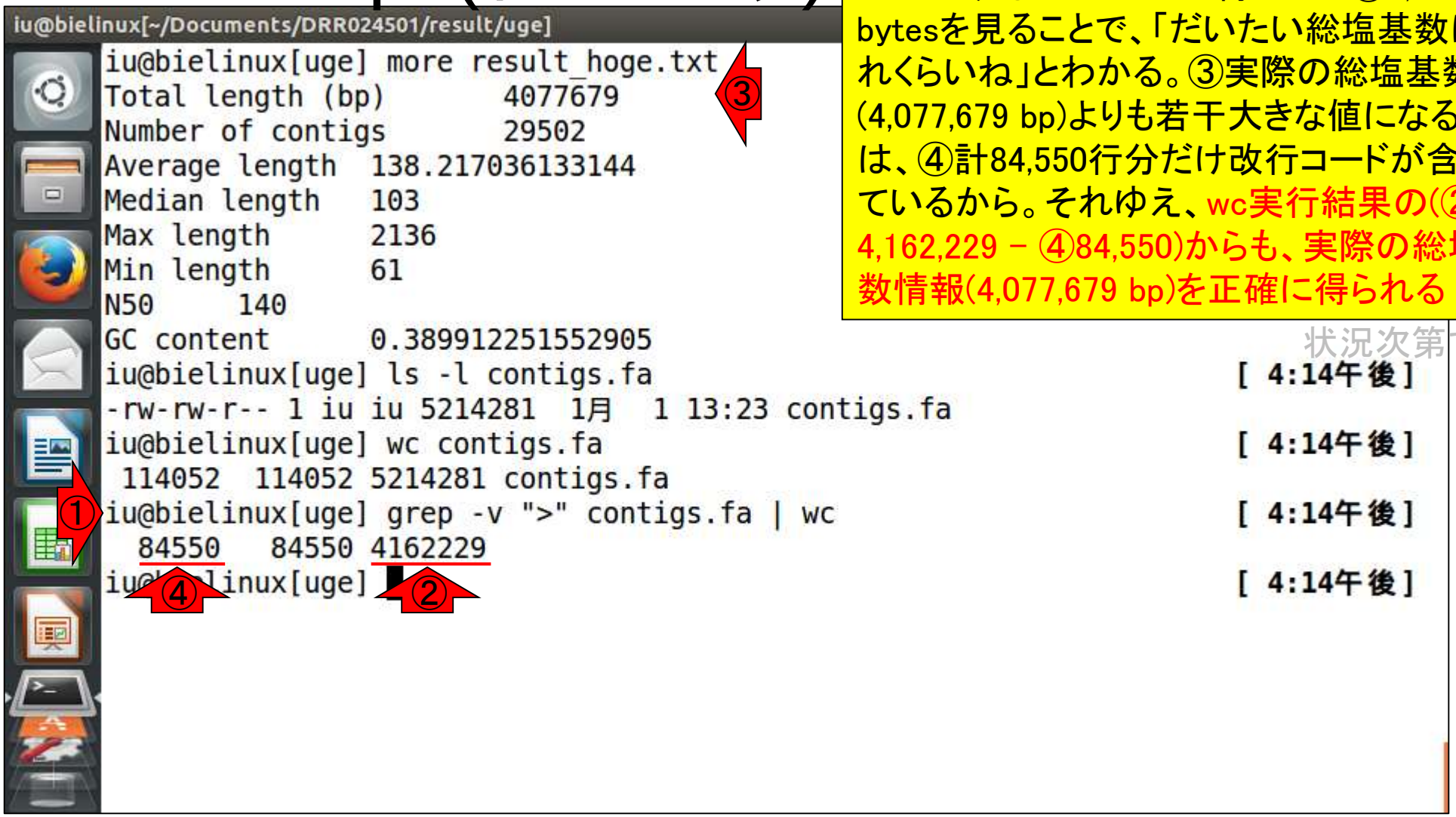
状況次第で省略

参考

W8-2：Tips(総塩基数)

そのため、①contigs.faからdescription行を除いた(grep -v “>”に相当)分を、パイプでつないでwcすることによって得られた②4,162,229 bytesを見ることで、「だいたい総塩基数はこれくらいね」とわかる。③実際の総塩基数(4,077,679 bp)よりも若干大きな値になる理由は、④計84,550行分だけ改行コードが含まれているから。それゆえ、wc実行結果の(②4,162,229 − ④84,550)からも、実際の総塩基数情報(4,077,679 bp)を正確に得られる

状況次第で省略

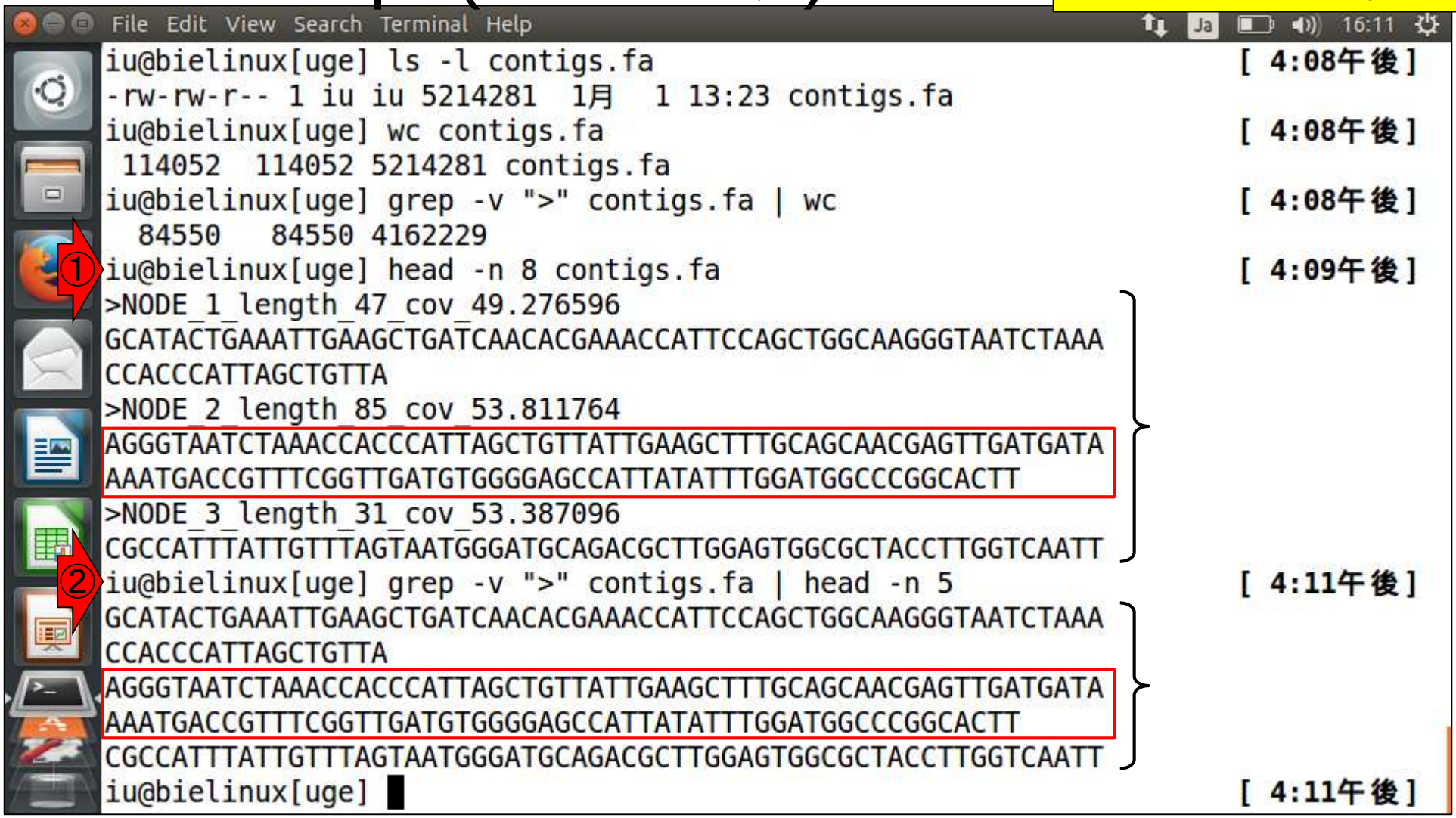


参考

W8-2：Tips(総塩基数)

状況次第で省

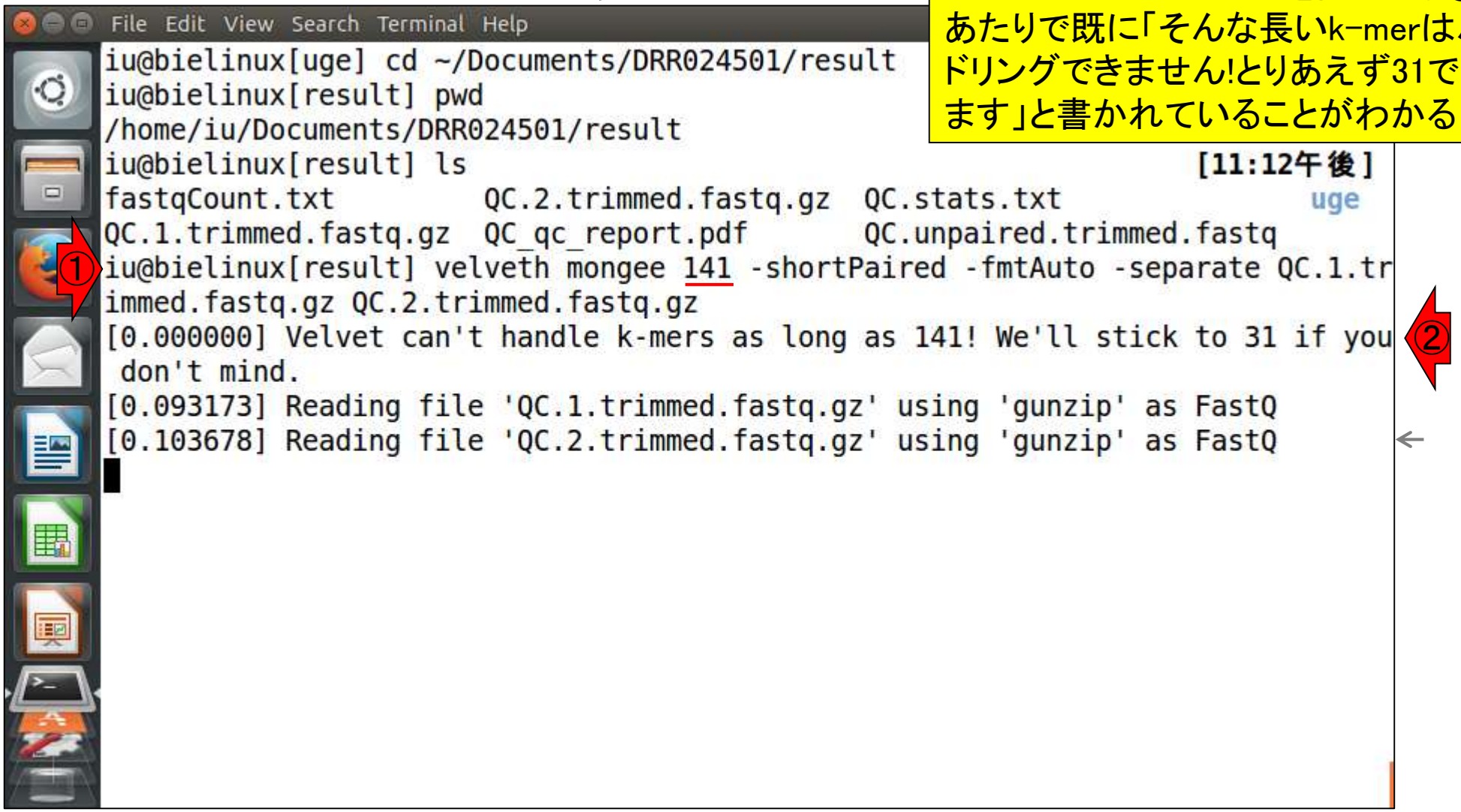
①と②の比較で、「grep -v “>”」によってdescription行をうまく除去できていることがわかる



参考

W8-3：k=141で実行

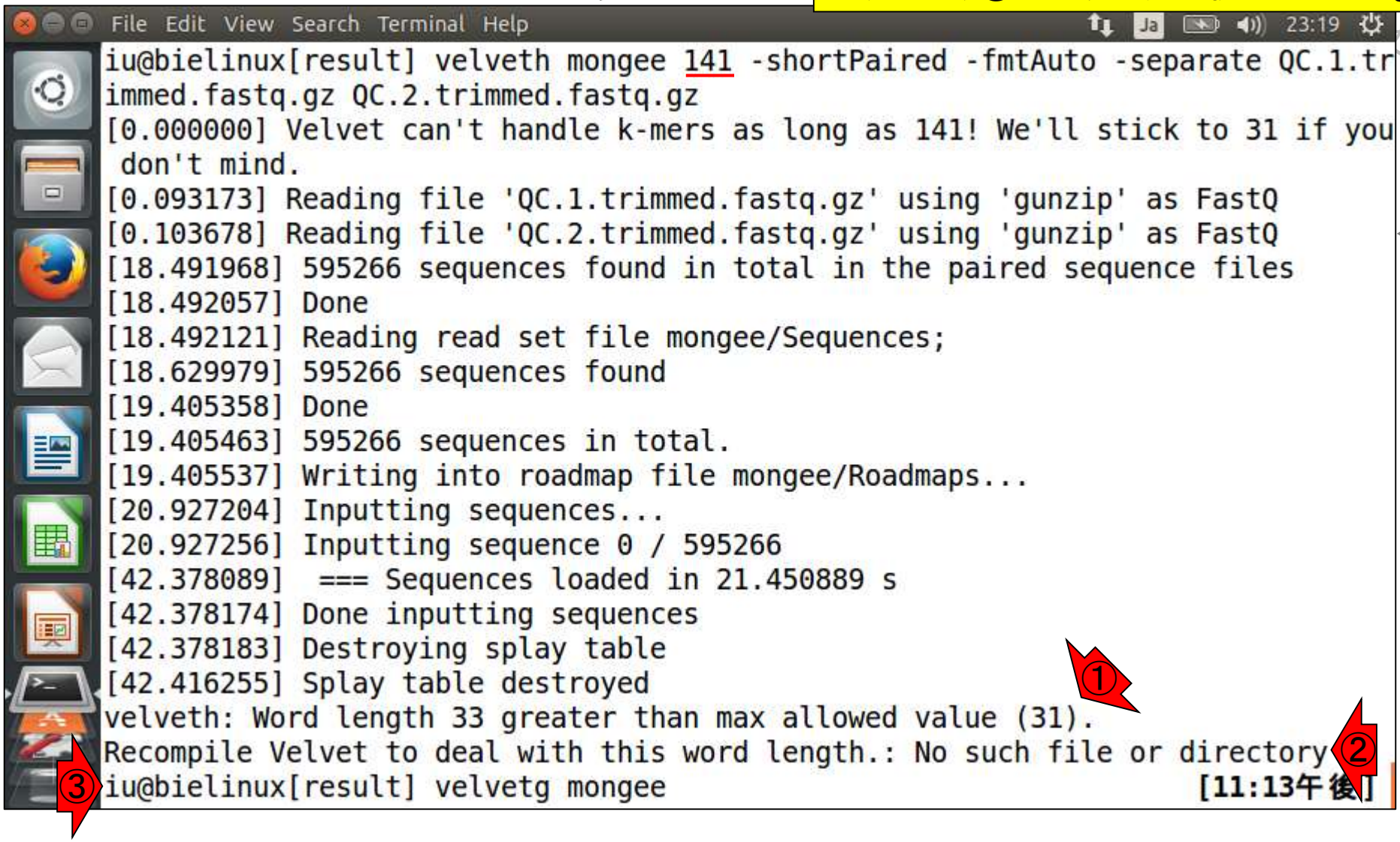
W7-3でも示したが、k=31以上の数値を指定すると31になることを一応確認すべくk=141でアセンブルを試みる。②のあたりで既に「そんな長いk-merはハンドリングできません!とりあえず31でやります」と書かれていることがわかる



参考

W8-3：k=141で実行

①のあたりでも上限は31だと書かれている。②「そんなファイルはない!」と文句を言われているようだが、③とりあえず無視してvelvetgを実行

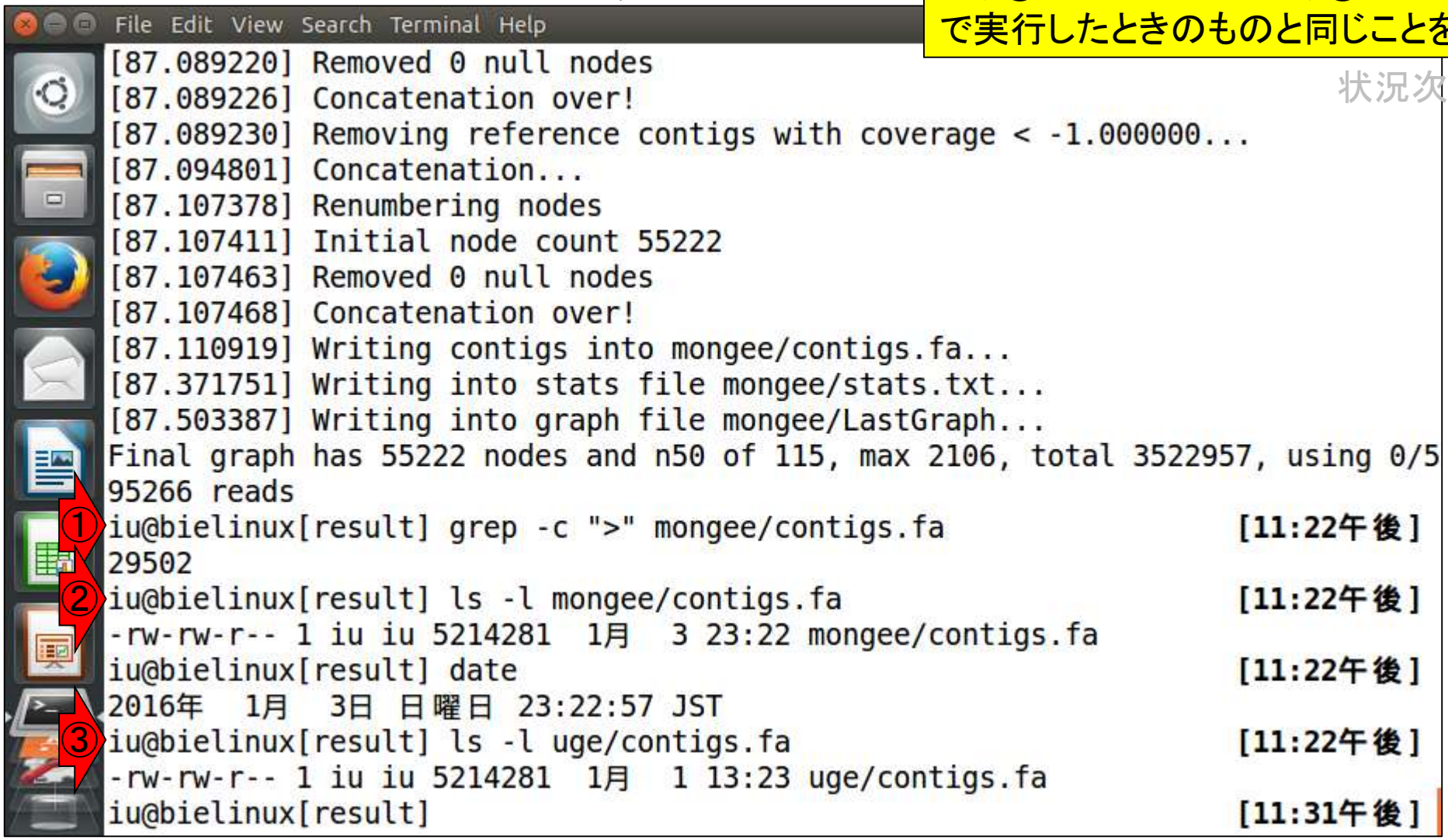


参考

W8-3：k=141で実行

無事終了したようだ。①配列数を数えた結果は、予定通りk=31のときと同じ29,502個。②ファイルサイズも、③明示的にk=31で実行したときのものと同じことを確認

状況次第で省略



Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

W9-1：Velvetマニュアル

①PDFマニュアルを開く。②Installationの、③MAXKMERLENGTHのところがk値の上限を指定するオプションのところ。スライドを見るだけ

EMBL-EBI

Velvet

Sequence assembler for very short re

- Current version: 1.2.10
- Manual and extension for Columbus in pdf forma
- ① Git URL: git clone git://github.com/dzerbi
- For up-to-date info, you can consult and/or su mailing list.
- For transcriptomic assembly Velvet is extended b

http://www.ebi.ac.uk/~zerbino/velvet/ ②

Velvet Manual - version 1.1

Daniel Zerbino

August 29, 2008

Contents

③

Zerbino and Birney, *Genome Res.*, **18**: 821-829, 2008

W9-1：Velvetマニュアル

2.3.3 MAXKMERLENGTH.

Another useful compilation parameter is the MAXKMERLENGTH. As explained in 5.2, the hash length can be crucial to getting optimal assemblies. Depending on the dataset, you might wish to use long hash lengths.

By default, hash-lengths are limited to 31bp, but you can push up this limit by adjusting the MAXKMERLENGTH parameter at compilation time:

```
make 'MAXKMERLENGTH=57'
```

①

(Note the single quotes and absence of spacing.)

By storing longer words, Velvet will be requiring more memory, so adjust this variable according to your needs and memory resources.

②

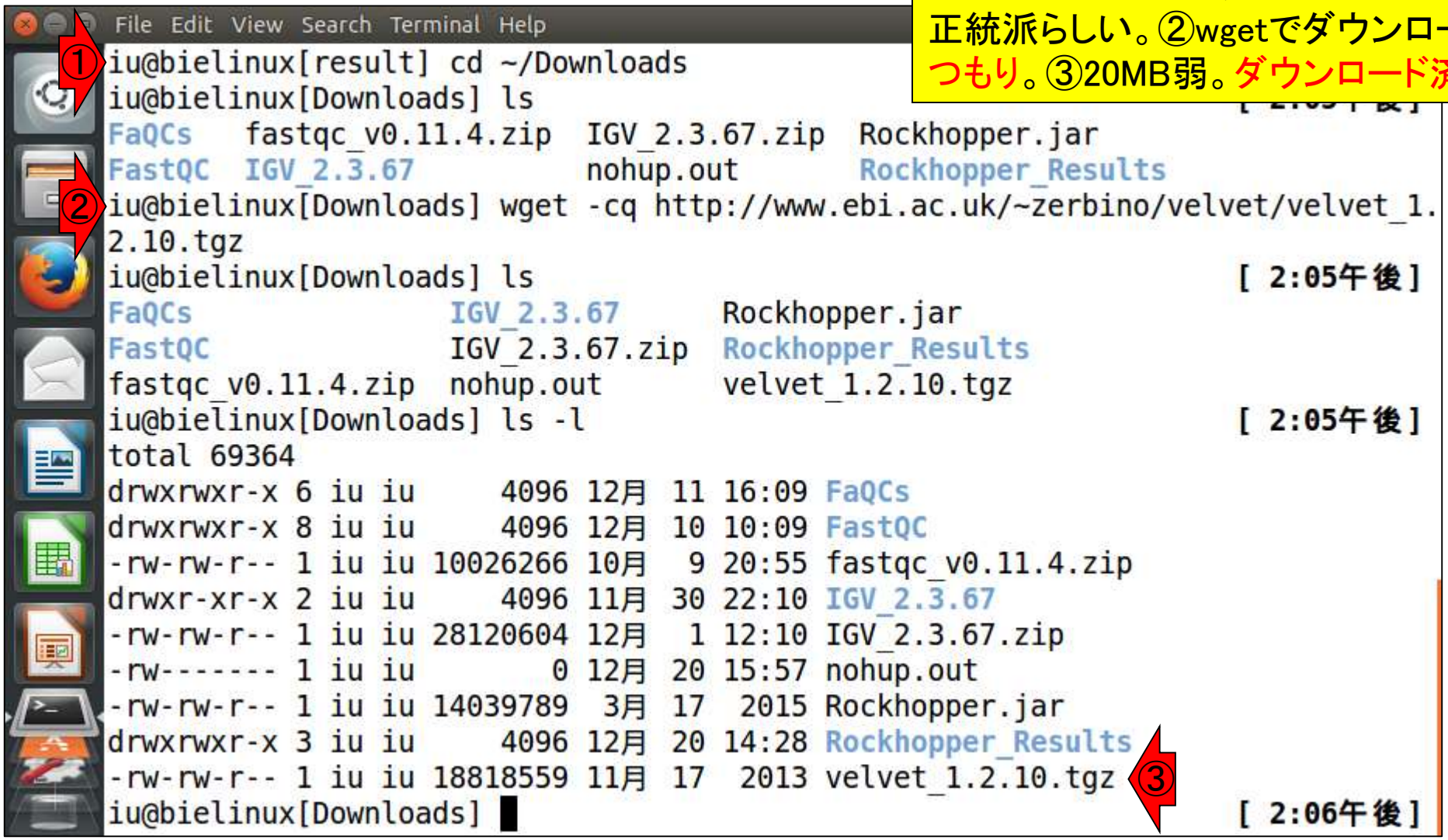
makeというコマンドでインストールを行うのが基本。①MAXKMERLENGTHオプションをつけてコンパイル（velvethやvelvetgコマンドを作成する作業に相当）を行うことで、デフォルトの31よりも大きい任意のk値を指定可能になる。ただし、②大きな値を指定するほどメモリを消費するので、実際にどこまでアセンブルできるかはやってみないとわからない

W9-2：ダウンロード

①Current versionのところで右クリックし、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。スライドを見るだけ

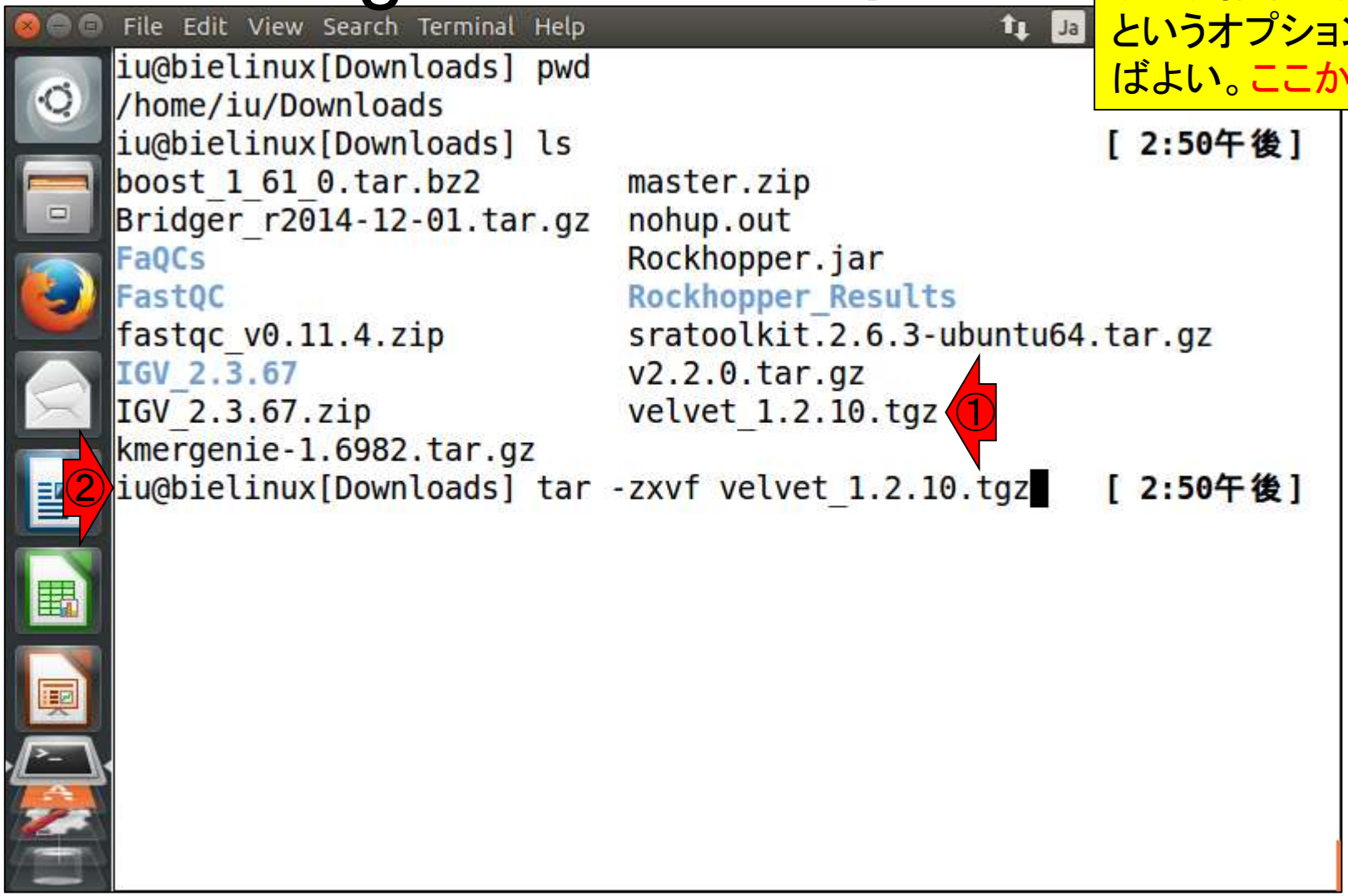
W9-2：ダウンロード

①本連載では、プログラムの大元は~/Downloadsに置くようにしているが、/usr/local/srcというディレクトリに置くのが正統派らしい。②wgetでダウンロードしたつもり。③20MB弱。ダウンロード済みです



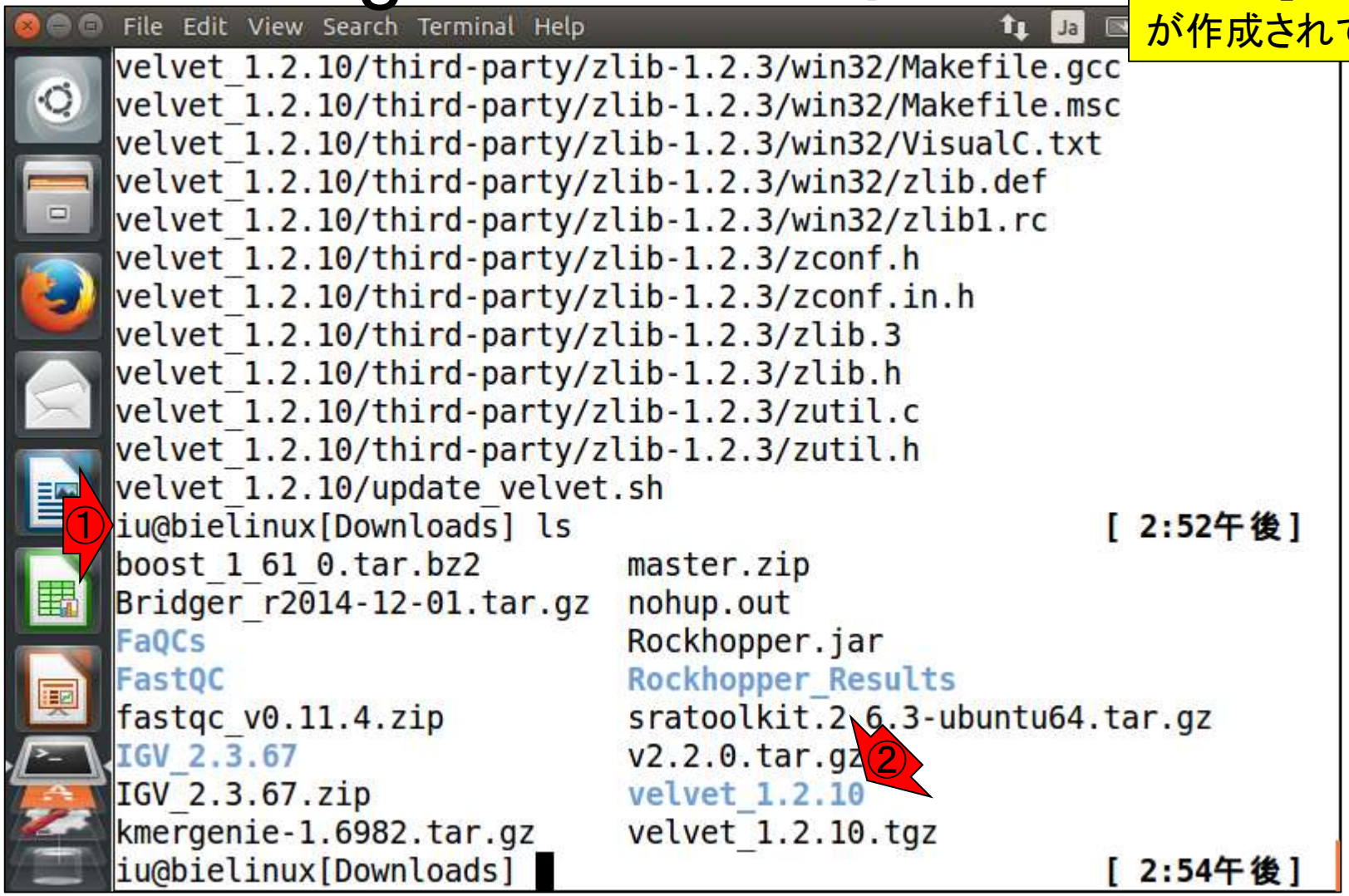
W9-3：tgzファイルの解凍

①.tgzという拡張子は、.tar.gzの略。このファイル形式も比較的よく目にする。解凍は、②tarコマンドを-zxvfというオプションをつけて実行すればよい。ここからは手を動かす



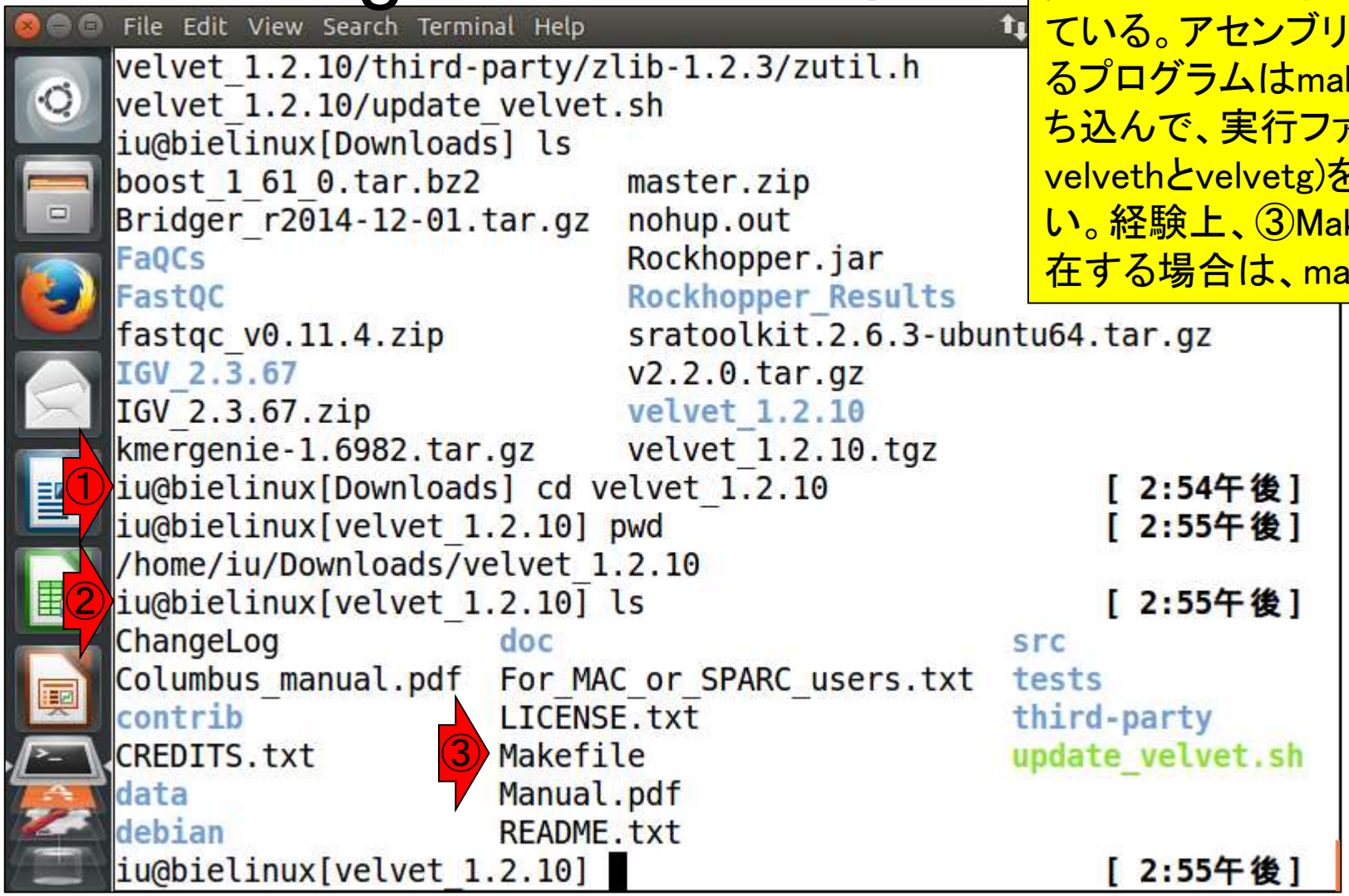
W9-3：tgzファイルの解凍

解凍は、ほぼ一瞬で終わる。①lsで確認。②確かに無事解凍されてvelvet_1.2.10というディレクトリが作成されていることがわかる



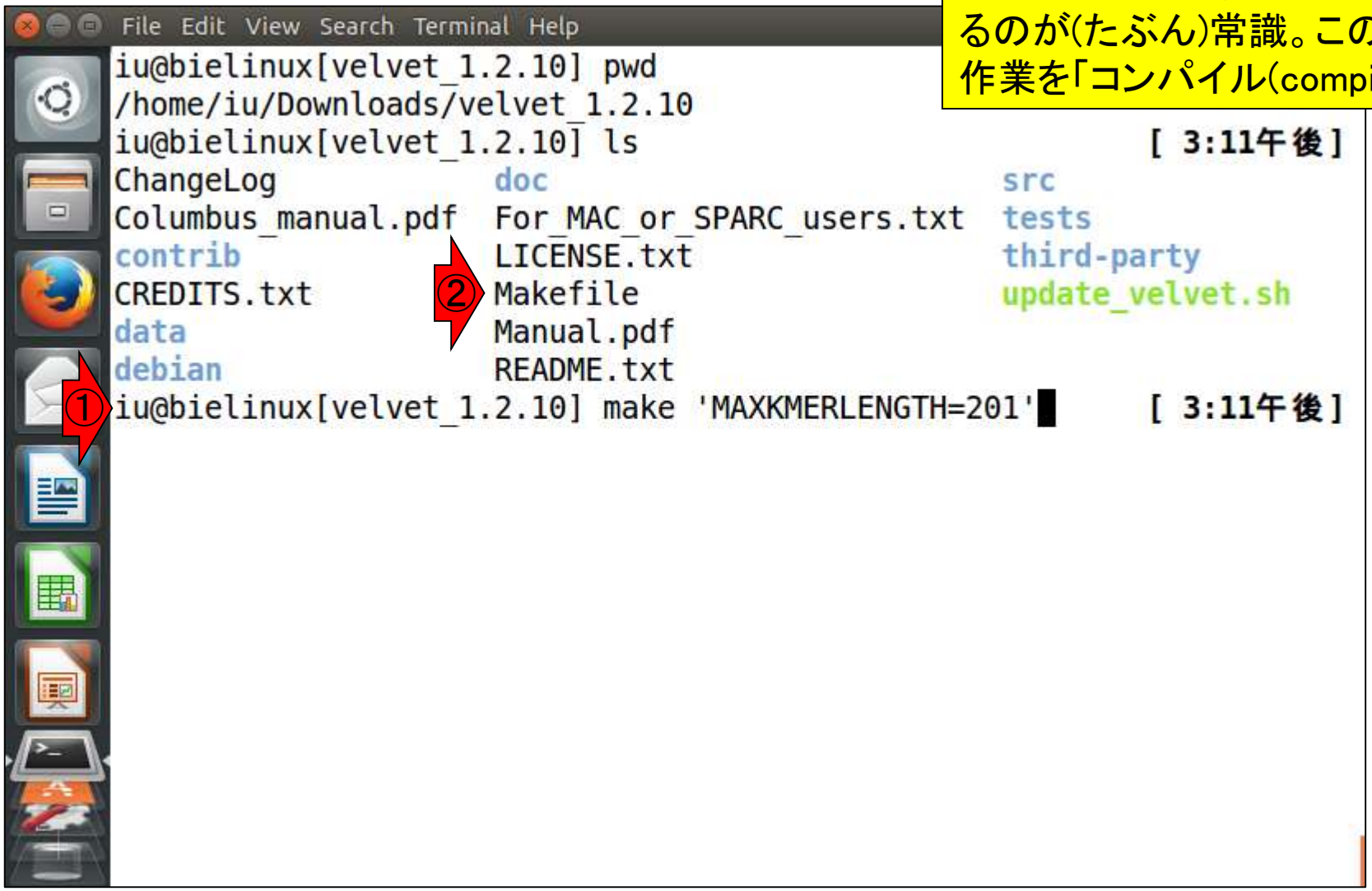
W9-3：tgzファイルの解凍

①velvet_1.2.10ディレクトリに移動し、②lsで中身を確認。マニュアルの2.2節を見ると「make(と打つのが基本)」と書いている。アセンブリなど計算時間のかかるプログラムはmakeというコマンドを打ち込んで、実行ファイル(この場合velvethとvelvetg)を作成する場合が多い。経験上、③Makefileというものが存在する場合は、makeと打てばよい



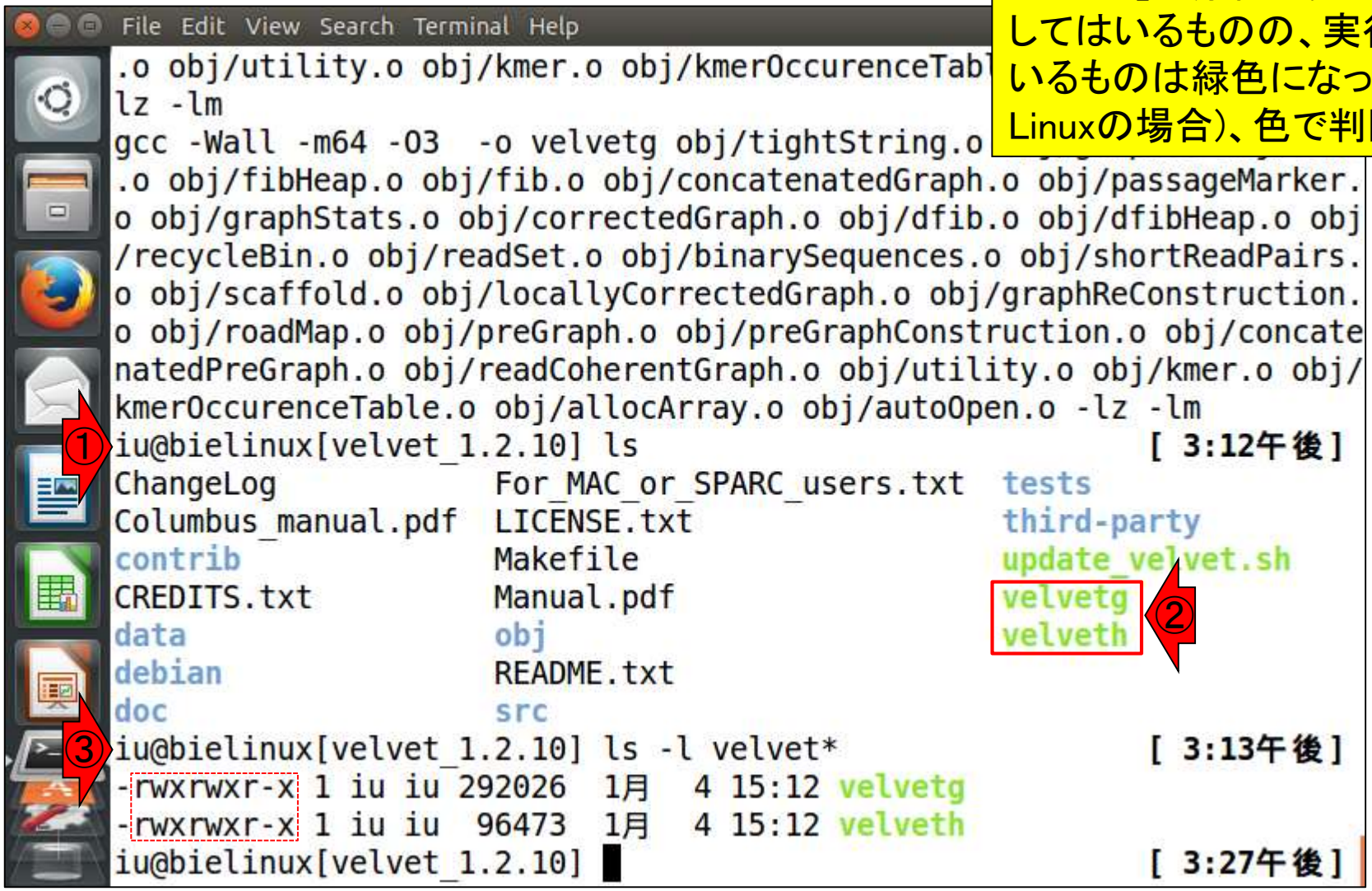
W9-4 : make

①k値の指定可能な最大値を201にしたい場合。マニュアルには明記されていないが、②Makefileが存在するディレクトリ上で実行するのが(たぶん)常識。このmakeを実行する作業を「コンパイル(compile)する」という



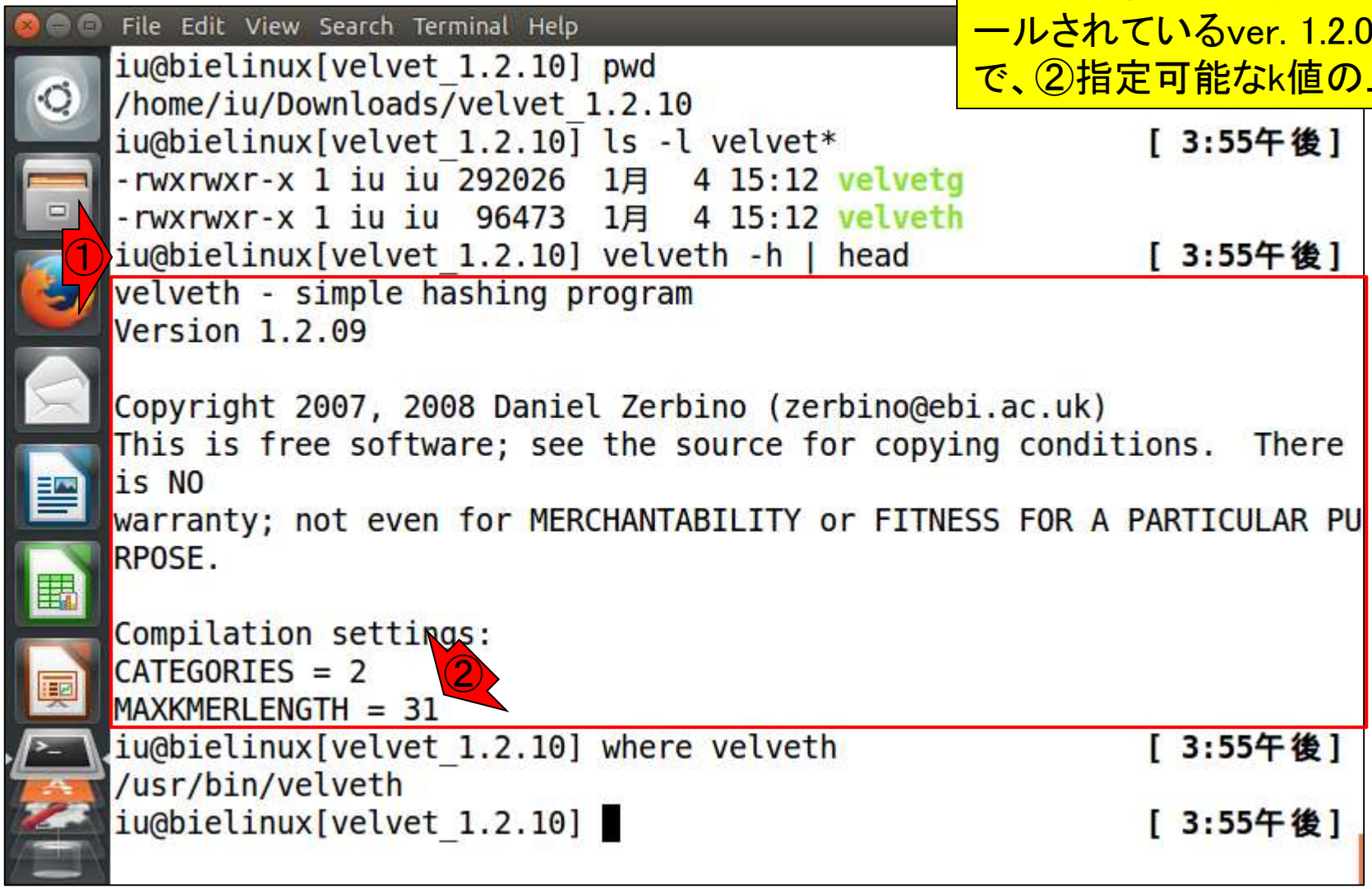
W9-4：make

make実行時間は数秒。①lsで確認。確かに②velvethとvelvetgができている。③一応「ls -l」で赤枠の実行権限部分を確認してはいるものの、実行権限(x)がついているものは緑色になっているので(Bio-Linuxの場合)、色で判断してもよい



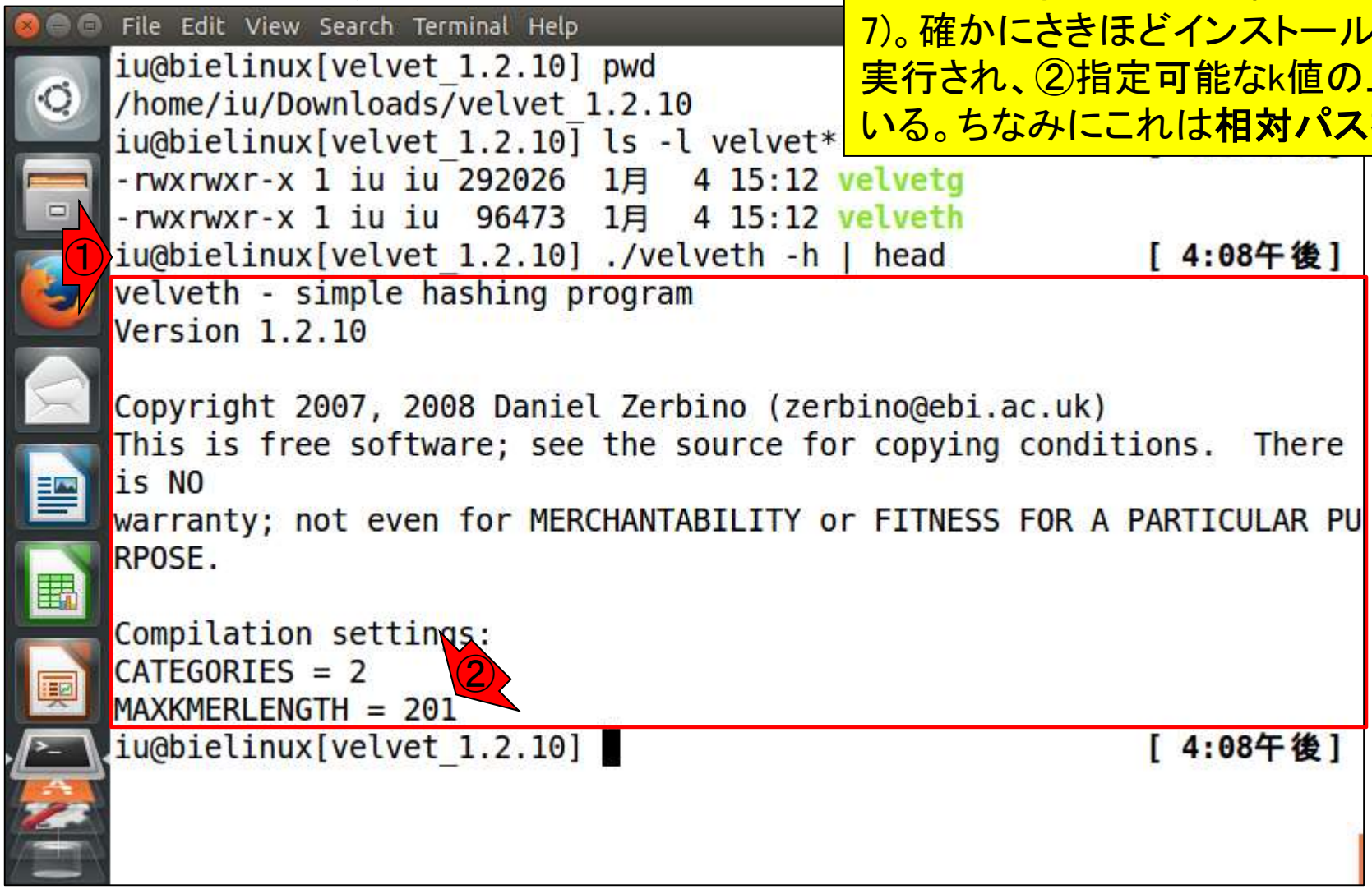
W9-5：復習と確認

第4回のW9-3でパスの概念を説明した通り、「~/Downloads/velvet_1.2.10」上で①velvethと打っても、Bio-Linuxにプレインストールされているver. 1.2.09が実行されるので、②指定可能なk値の上限は31のまま



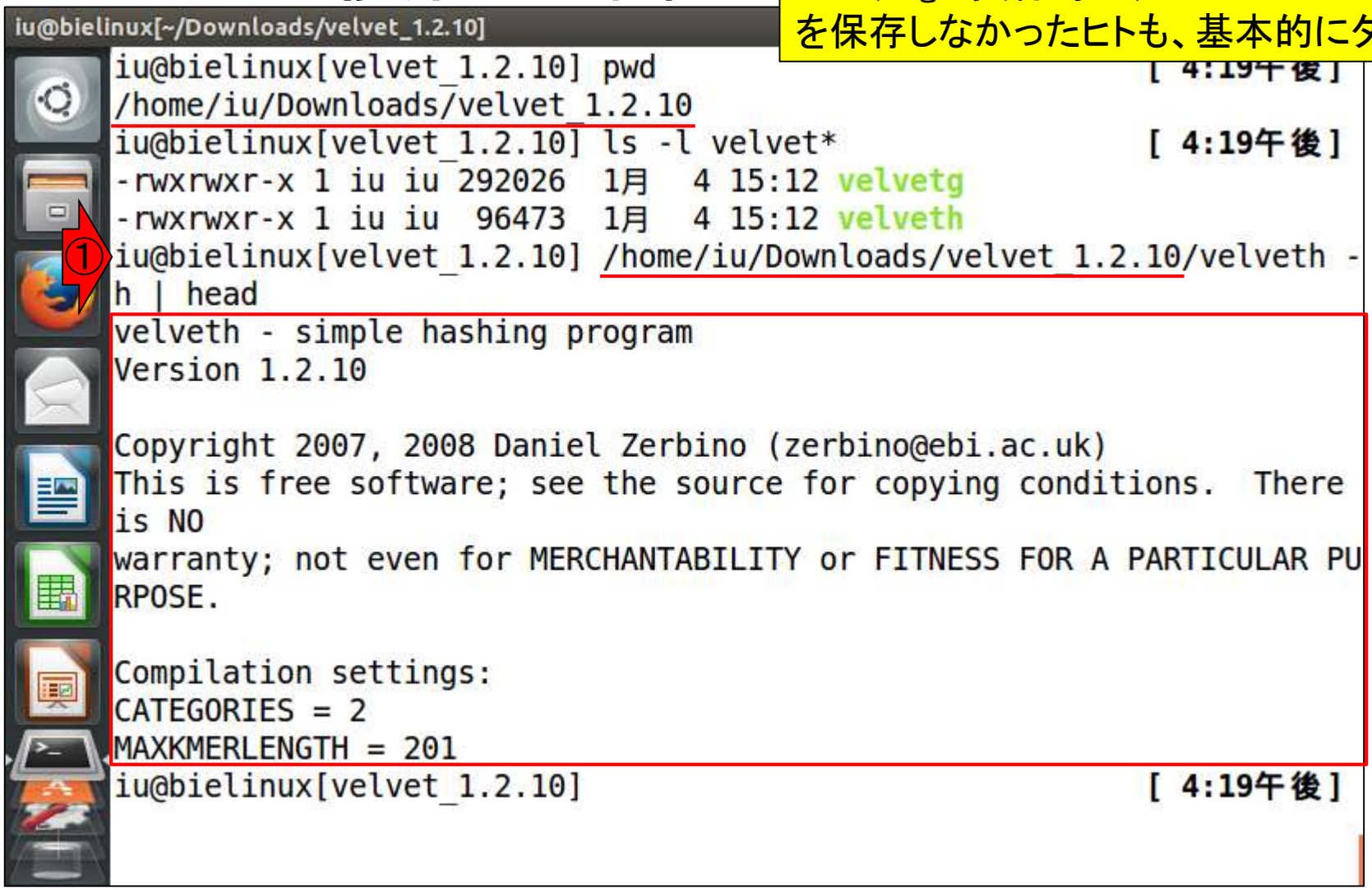
W9-5：復習と確認

カレントディレクトリの「~/Downloads/velvet_1.2.10」上に存在するvelvethを実行したい場合は、①./velvethと打てばよい(第4回のW3-1, W9-3, W15-7)。確かにさきほどインストールしたver. 1.2.10が実行され、②指定可能なk値の上限は201になっている。ちなみにこれは**相対パス**指定のやり方



W9-5：復習と確認

絶対パス指定の場合は、①こんな感じ。当然ユーザ名がiuでないヒトはこれをそのまま打ち込んでもダメ!。また、wget実行時に、~/Downloadsにvelvet_1.2.10.tgzを保存しなかったヒトも、基本的にダメ!

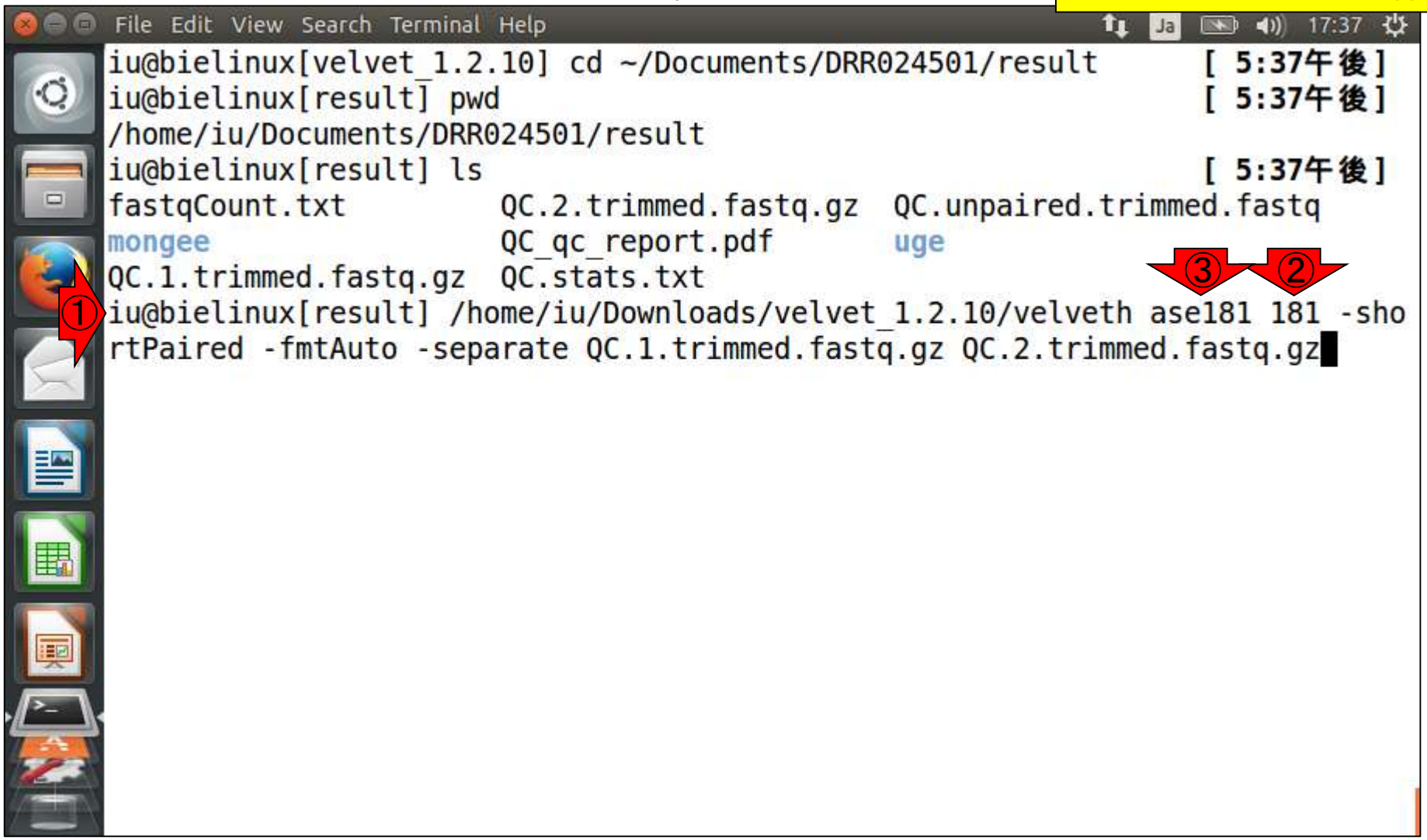


Contents

- Illumina HiSeqデータ(トランスクリプトーム)の乳酸菌ゲノムへのマッピング
  - W14:QuasRパッケージを用いたマッピングの事前準備と本番
  - W15:結果の解説、forward側の100-107bp付近に問題があることを特定
- トリミング、*de novo*トランスクリプトームアセンブリとマッピングの再実行
  - W16:問題のある領域(forward側の100-107bp)のトリミング
  - W17:トリム後のデータでアセンブリを再実行(Rockhopper2; クラスパス設定関連Tips含む)
  - W18:トリム後のデータでマッピングを再実行(QuasR)
  - W19:トリム前のデータでクオリティチェックを再実行(FastQC)
- Illumina MiSeqデータ(乳酸菌ゲノム)の特徴と前処理
  - W4:FastQC、W5:FaQCs、W6:再度FastQC
- *de novo*ゲノムアセンブリ
  - W7:Bio-Linuxにプレインストールされている Velvet (ver. 1.2.09)を上限のk=31で実行
  - W8:k=31のアセンブリ結果をRで確認。k=141で実行し、k=31の結果と同じになるのを確認
  - W9:Velvet (ver. 1.2.10)のインストール
  - W10:Velvet (ver. 1.2.10)の実行

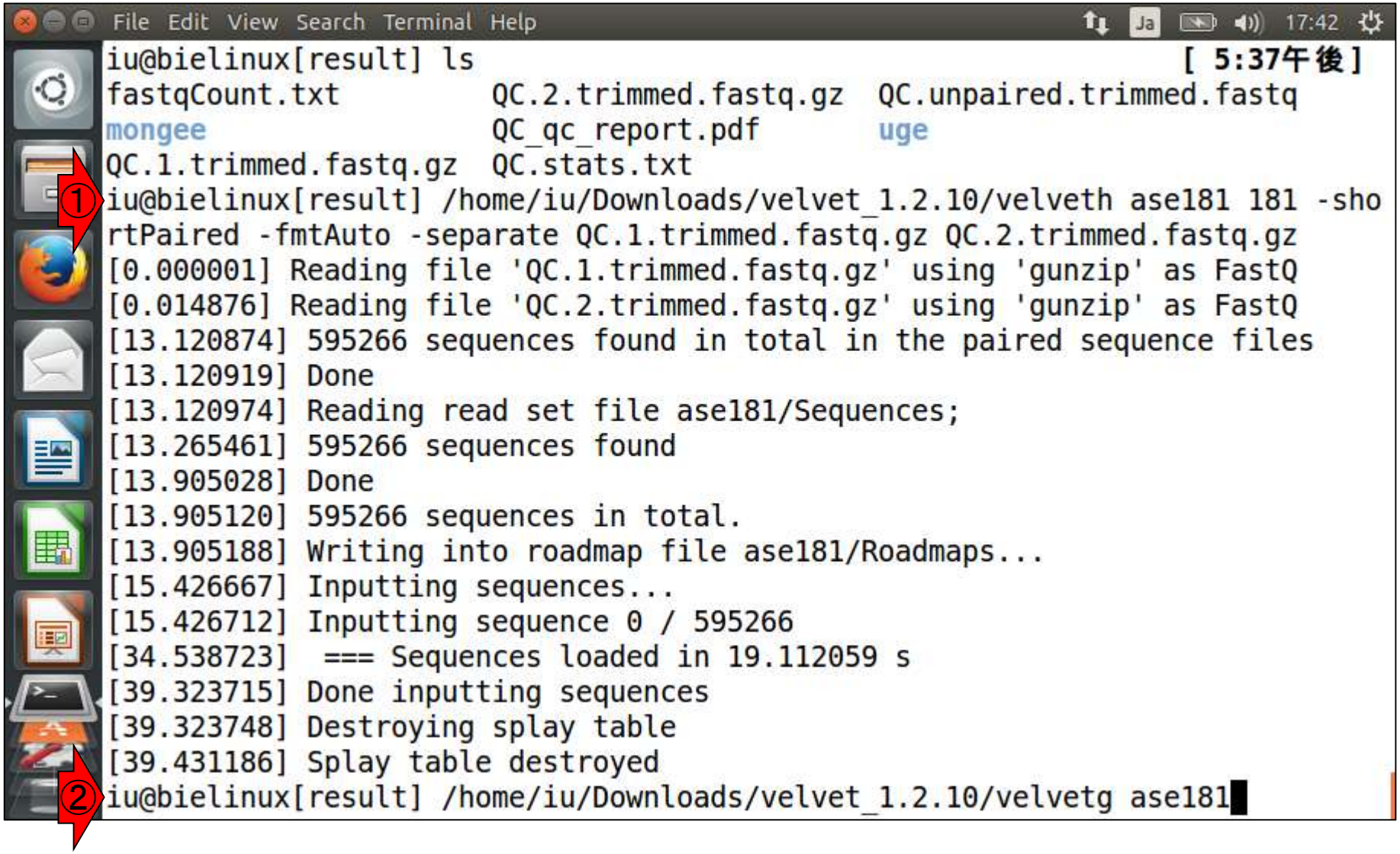
W10-1：Velvet再実行

①絶対パス指定でvelvethを実行。② k=181で実行し、結果は③ase181ディレクトリに保存。約2分



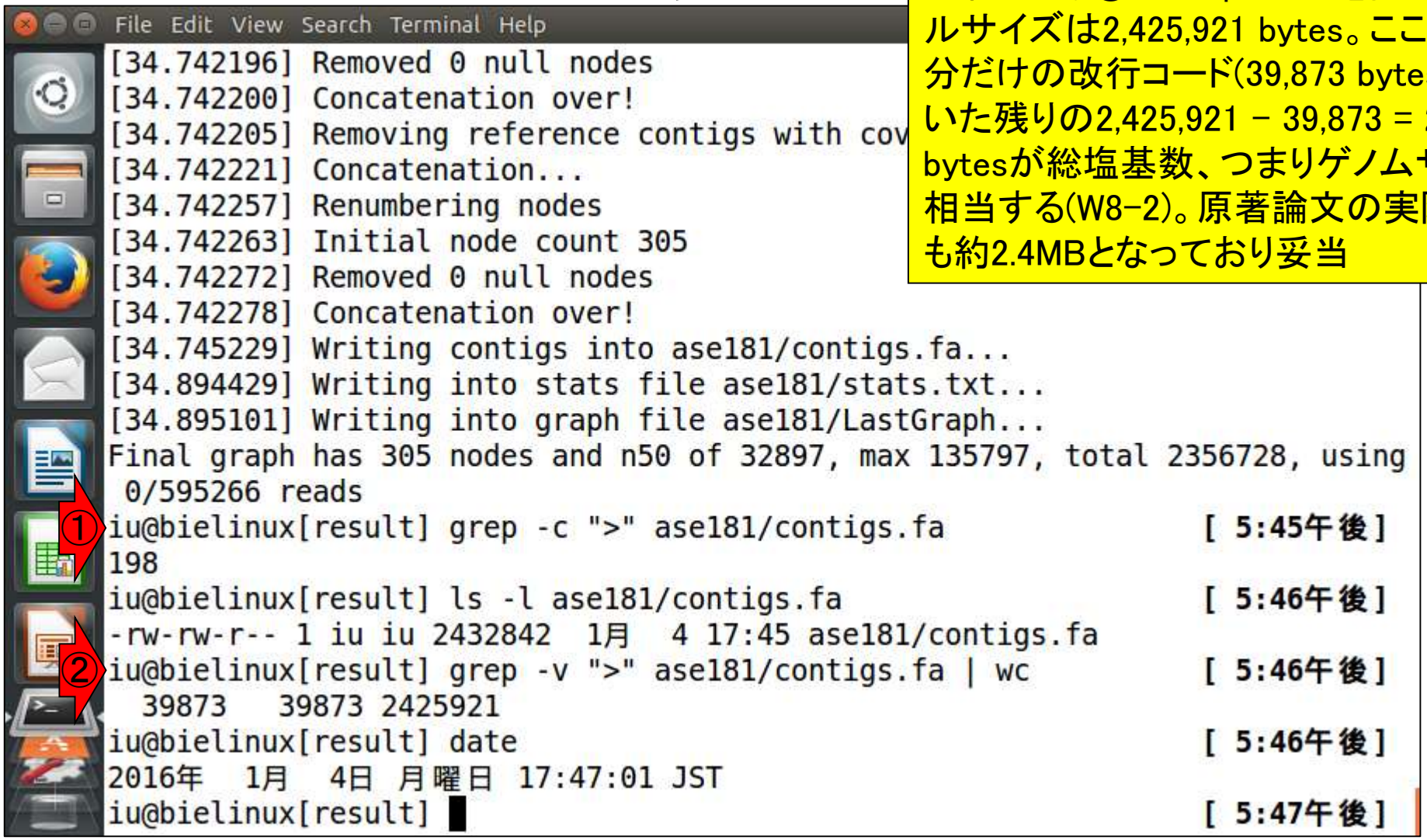
W10-1：Velvet再実行

①velvethが無事終了したので、次は②velvetgを実行。約1分



W10-1：Velvet再実行

①得られた配列数は198個。k=31のとき（29,502個; W7-8）と比べて激減していることがわかる。②description行を除いたファイルサイズは2,425,921 bytes。ここから行数分だけの改行コード(39,873 bytes)を差し引いた残りの2,425,921 - 39,873 = 2,386,048 bytesが総塩基数、つまりゲノムサイズに相当する(W8-2)。原著論文の実際の数値も約2.4MBとなっており妥当



W10-5：再挑戦

(多少話が飛躍するが)計6個のk値で一気にVelvetを実行。赤枠のような一連のコマンドからなるファイルを用意しておき、シェルスクリプト or コピペで実行すると(デフォルトの2GBメモリで)約1時間。しかし、VirtualBox上で4GBメモリに変更すると約10分で終了することが判明したので、設定変更して実行します

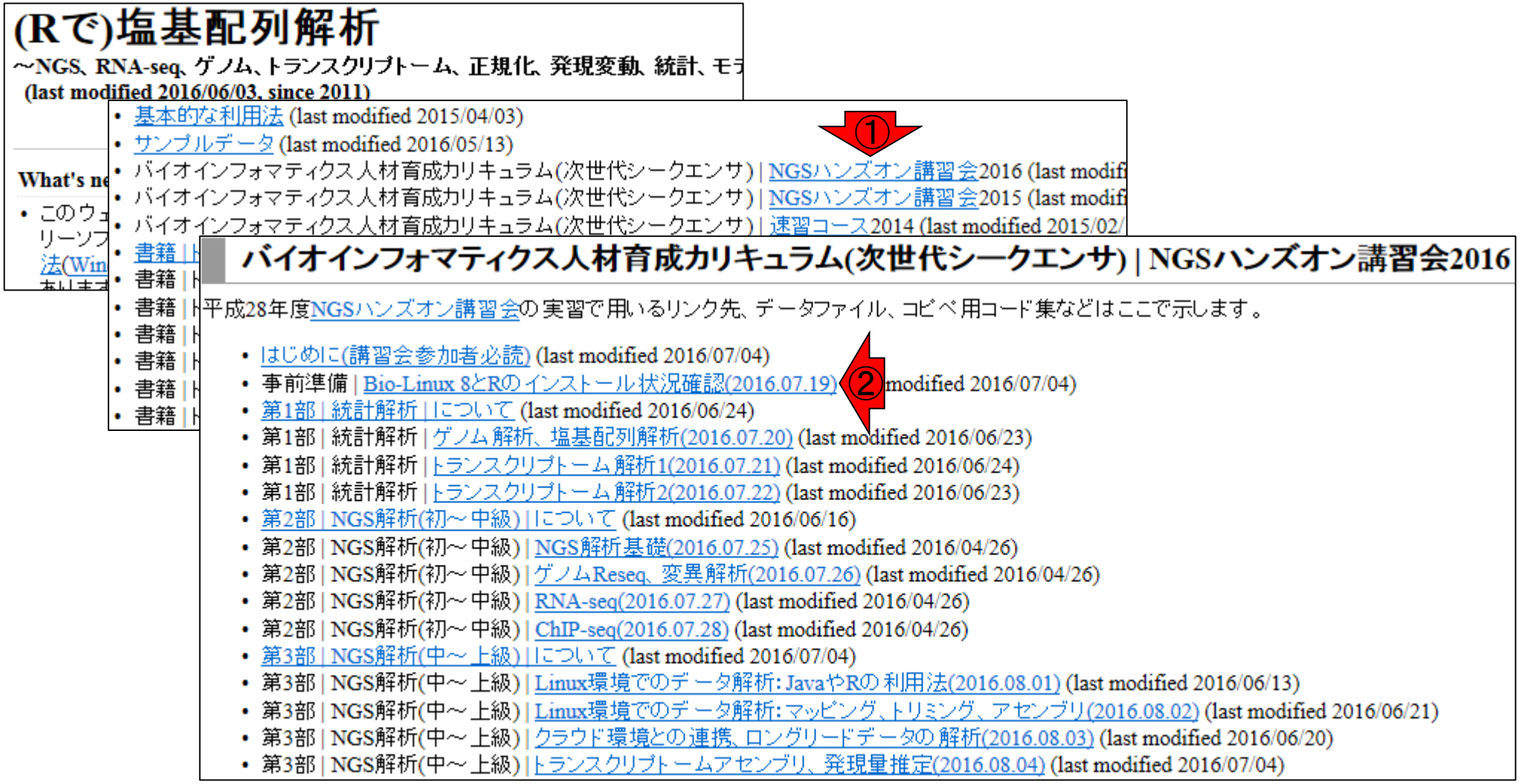
- 再挑戦[W10-5]
  一旦アセンブル結果を全て削除し、今度はk=111,121, 131, 151,

①
②

```
ls

/home/iu/Downloads/velvet_1.2.10/velveth hoge_111 111 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_121 121 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_131 131 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_151 151 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_171 171 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_191 191 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_111
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_121
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_131
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_151
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_171
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_191
grep -c ">" hoge_*/contigs.fa
grep -v ">" hoge_111/contigs.fa | wc
grep -v ">" hoge_121/contigs.fa | wc
grep -v ">" hoge_131/contigs.fa | wc
grep -v ">" hoge_151/contigs.fa | wc
grep -v ">" hoge_171/contigs.fa | wc
grep -v ">" hoge_191/contigs.fa | wc
```

使用メモリ増加

スライドを見るだけ。②事前準備(2016.07.19)の…



使用メモリ増加

①VirtualBox関連Tipsの中に、使用メモリ変更の項目があります。スライドを見るだけ

事前準備 | Bio-Linux 8とRのインストール状況確認(2016.07.19)

作成途中です。基本的に昨年度と同じく自習です。スタッフが複数人常駐予定ですので、インストールで躓いた箇所の相談など個別対応してもらってください。

1. Bio-Linux8(第2部および3部で利用するovaファイル)の導入確認
   第2部および3部で利用するovaファイルは6GBから10GB程度になります。そしてそれを使って第2部および3部を受講してもらいます。ovaファイルは独立に提供(URLをお知らせする)予定ですので、例えば全日程参加者は、第2部用ovaファイルと第3部用ovaファイルを独立にダウンロードしておいてください。なるべく早い段階でovaファイルをダウンロード可能な状態にする予定ですが…、ネットワーク環境的にダウンロードできないヒトも一定数いると思われます。当日は、それらの人々用に第2部および3部のovaファイルが入ったUSBメモリを講義室で用意する予定ですので、下記を参考にして各自のPCにコピーしてBio-Linuxを導入してください(もちろん乳酸菌NGS連載第2回最後の「1. VirtualBox、および2. Extension Pack」のインストールが完了しているという前提です)。
   - ovaファイルの導入手順: Windows用(2015.12.28版; 約2MB)
   - ovaファイルの導入手順: Macintosh用(2015.12.28版; 約2MB)

2. 共有フォルダ設定完了確認
   第2部および第3部参加者のみ。

3. 基本的なLinuxコマンドの習得状況確認
   第2部および第3部参加者のみ。

4. R本体およびパッケージのインストール確認
   基本的に第1部参加者のみでよい。

5. 講師指定の事前予習内容の再確認
   - 第1部 | 統計解析 | について (last modified 2016/06/20)
   - 第2部 | NGS解析(初～中級) | について (last modified 2016/06/16)
   - 第3部 | NGS解析(中～上級) | について (last modified 2016/07/04)

6. 講習会期間中に貸与されるノートPCを用いた各種動作確認
   基本的に貸与希望者のみではありますが、持込PCの不具合時にはアグリバイオPCを貸し出します。この際に普段利用するPC環境でないので戸惑うかもしれません。この日を利用して貸与PCに慣れておくというのもアリでしょう。

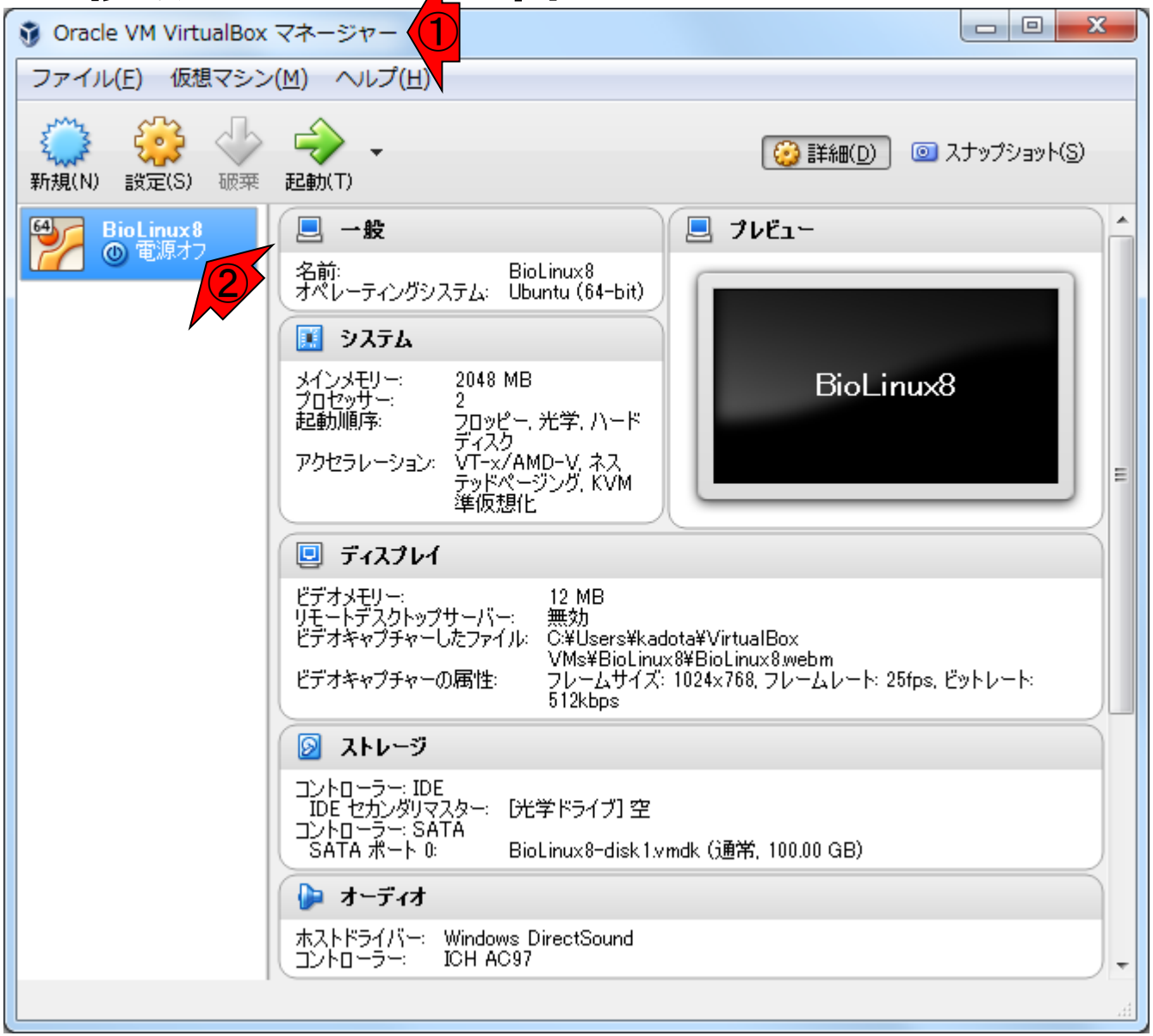
7. 無線LANの設定(持ち込みPCのみ)
   会場で利用可能なアクセスポイントの把握とパスワードの設定を行ってください(PWは当日スタッフから教えてもらってください)。

①

8. VirtualBox関連Tips(2016.07.04版)
   下記内容が含まれます。
   - BioLinux8の名前を変えたい
   - 使用メモリを変更したい
   - …

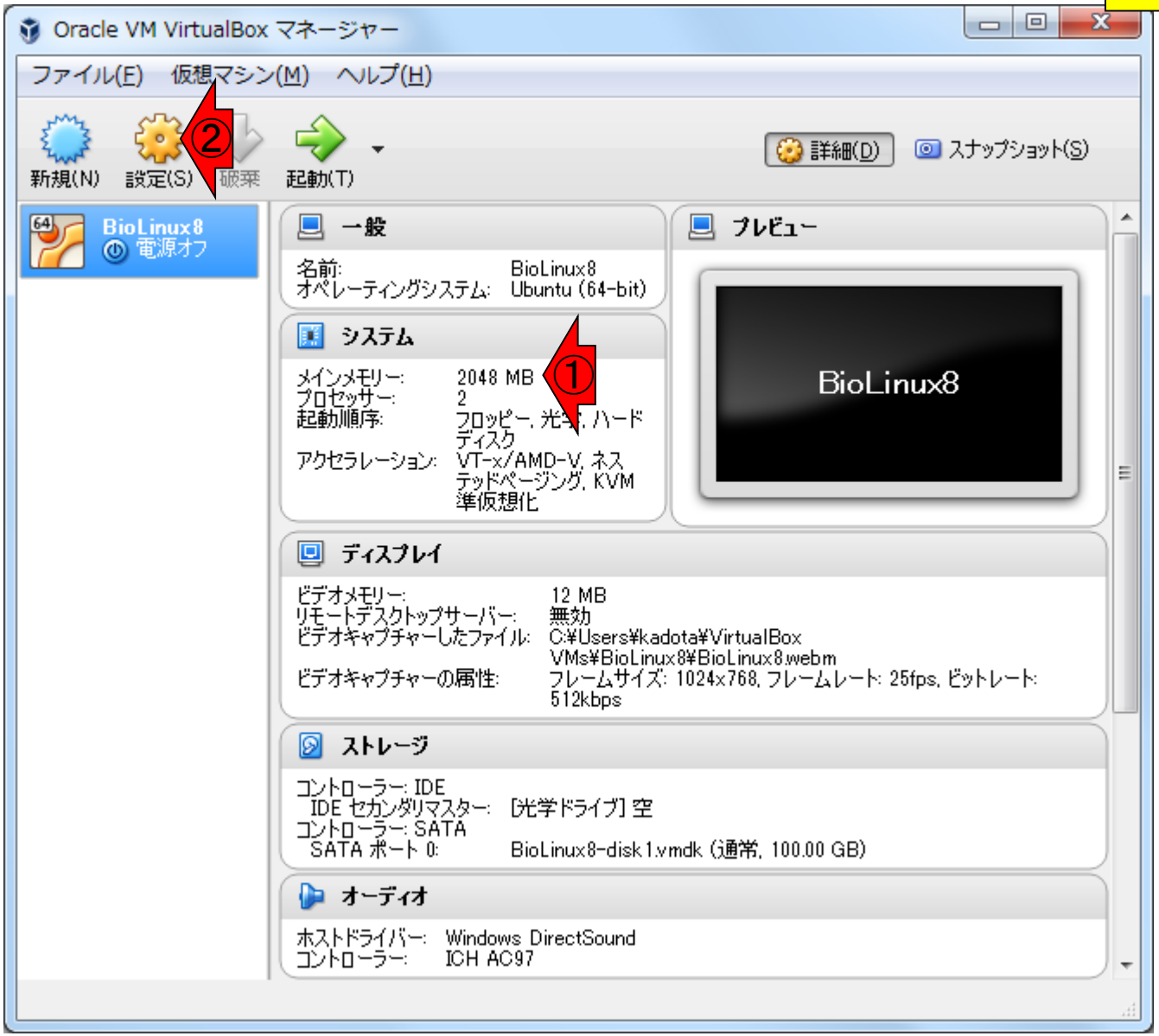
使用メモリ増加

一旦BioLinux8を保存せずに終了。すると、①VirtualBox上では、②電源オフになっているはず



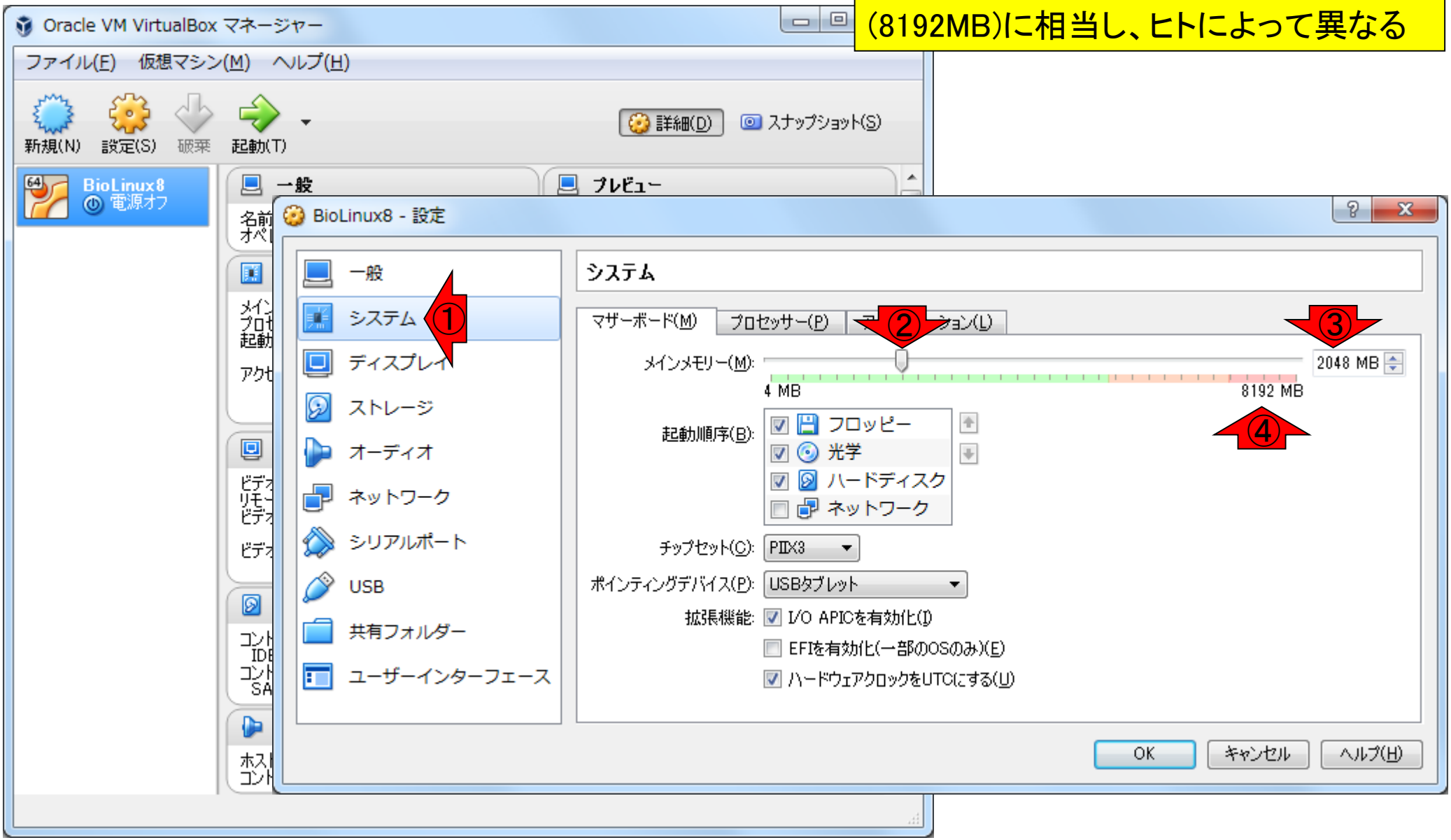
使用メモリ増加

①現在は2GB（2048MB）割り当てている状況です。これを4GB（4096MB）にするやり方を示します。②設定



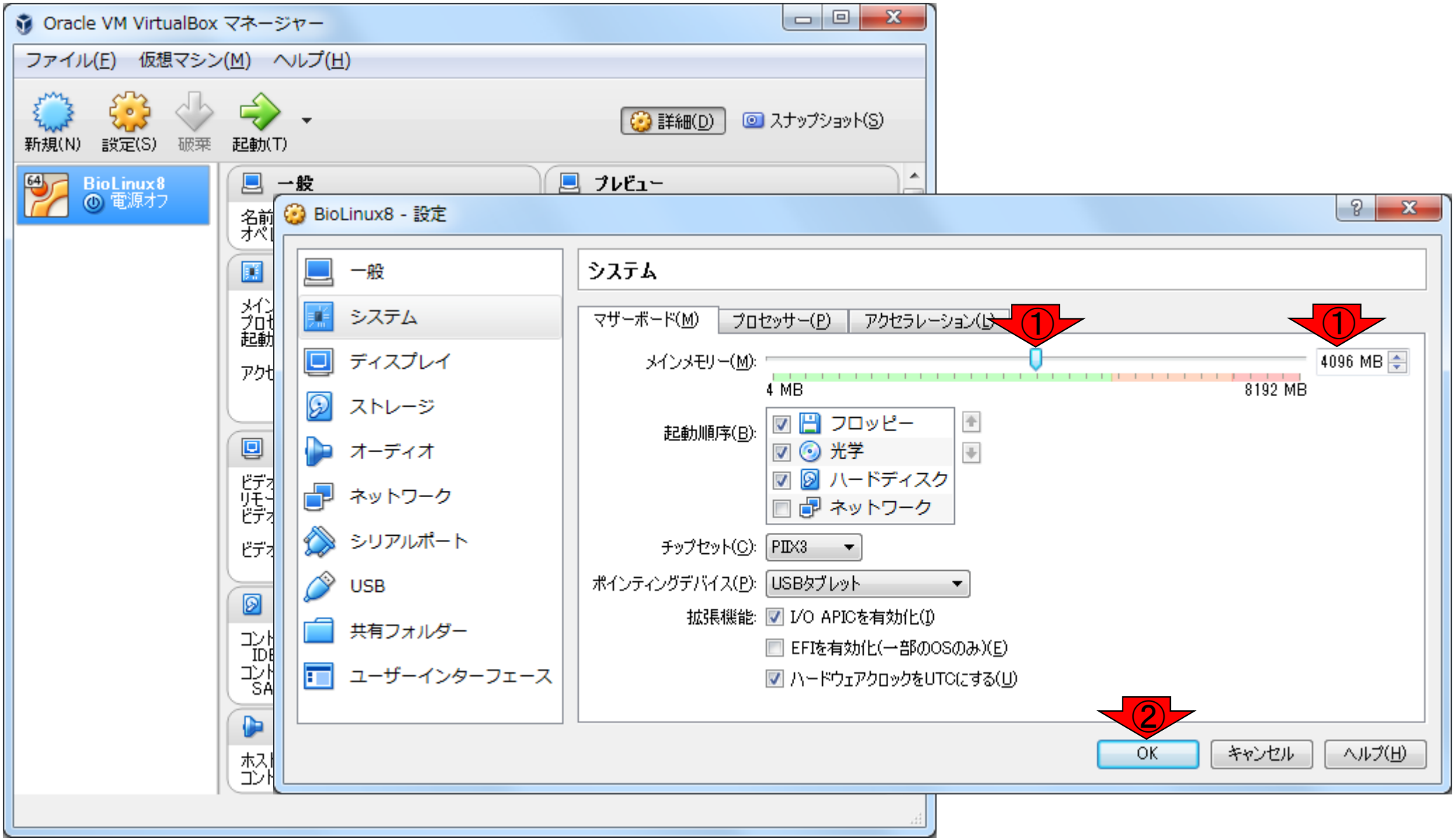
使用メモリ増加

①システム。②現在はここ。具体的な数値は③に表示されている。④ここの数値は、今使用中のノートPCのメモリ8GB(8192MB)に相当し、ヒトによって異なる



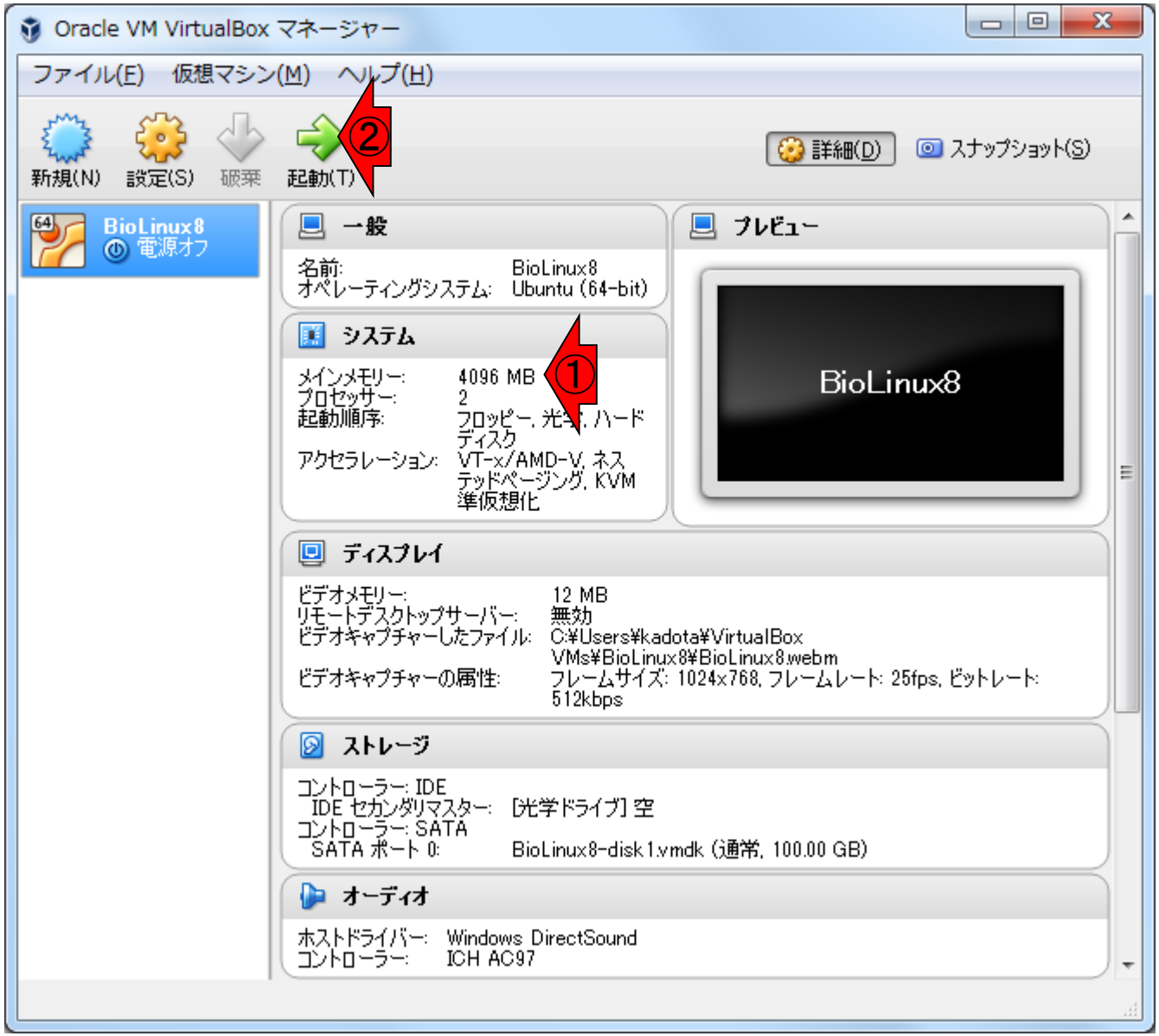
①メモリ4GBに変更して、②OK

使用メモリ増加



メモリ4GB変更後の状態

①メモリ4GBに確かに変更されたことが分かる。②起動して



W10-5：再挑戦

赤枠の前処理部分の注意。オリジナルのウェブ資料通りに赤枠部分をコピペしても、講習会では(mongeeやmulti_*というディレクトリを作成してないので)①のところでエラーが出て②を消せない。そのため、③のようにase181とugeディレクトリのみの削除にしてください

- 再挑戦[W10-5]
  一旦アセンブル結果を全て削除し、今度はk=111,121, 131, 15

```
cd ~/Documents/DRR024501/result

pwd
ls
rm -rf ase181 mongee uge multi_*
ls

/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
/home/iu/Downloads/ve
grep -c ">" hoge_*/co
grep -v ">" hoge_111/
grep -v ">" hoge_121/
```

① ② ③

```
File  Edit  View  Search  Terminal  Help                    Ja  15:17
iu@bielinux[iu] cd ~/Documents/DRR024501/result              [ 3:16午後]
iu@bielinux[result] pwd                                      [ 3:16午後]
/home/iu/Documents/DRR024501/result
iu@bielinux[result] ls                                       [ 3:16午後]
ase181                    QC_qc_report.pdf
fastqCount.txt            QC_stats.txt
QC.1.trimmed.fastq.gz     QC.unpaired.trimmed.fastq
QC.2.trimmed.fastq.gz     uge
iu@bielinux[result] rm -rf ase181 uge                        [ 3:16午後]
iu@bielinux[result] ls                                       [ 3:16午後]
fastqCount.txt            QC_qc_report.pdf
QC.1.trimmed.fastq.gz     QC.stats.txt
QC.2.trimmed.fastq.gz     QC.unpaired.trimmed.fastq
iu@bielinux[result]                                          [ 3:17午後]
```

赤枠部分をコピペで実行。4GBメモリで約10分

W10-5：再挑戦

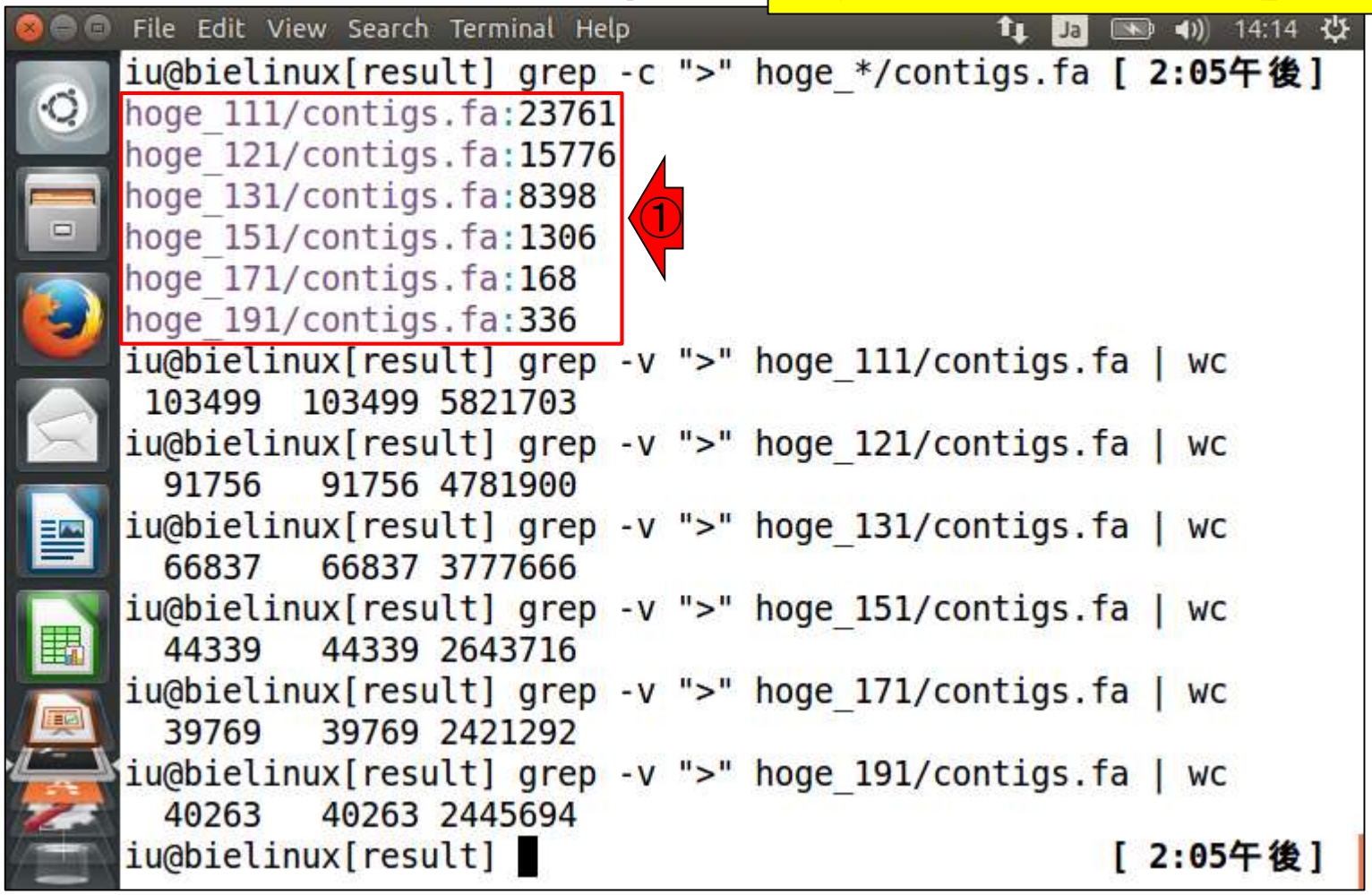
- 再挑戦[W10-5]
  一旦アセンブル結果を全て削除し、今度はk=111,121, 131, 151,171, 191でやってみる。

```
ls

/home/iu/Downloads/velvet_1.2.10/velveth hoge_111 111 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_121 121 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_131 131 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_151 151 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_171 171 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velveth hoge_191 191 -shortPaired -fmt
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_111
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_121
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_131
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_151
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_171
/home/iu/Downloads/velvet_1.2.10/velvetg hoge_191
grep -c ">" hoge_*/contigs.fa
grep -v ">" hoge_111/contigs.fa | wc
grep -v ">" hoge_121/contigs.fa | wc
grep -v ">" hoge_131/contigs.fa | wc
grep -v ">" hoge_151/contigs.fa | wc
grep -v ">" hoge_171/contigs.fa | wc
grep -v ">" hoge_191/contigs.fa | wc
```

```
Ja  15:17
/result                    [ 3:16午後]
                           [ 3:16午後]

                           [ 3:16午後]

ed.fastq

                           [ 3:16午後]
                           [ 3:16午後]

ed.fastq
iu@bielinux[result]        [ 3:17午後]
```

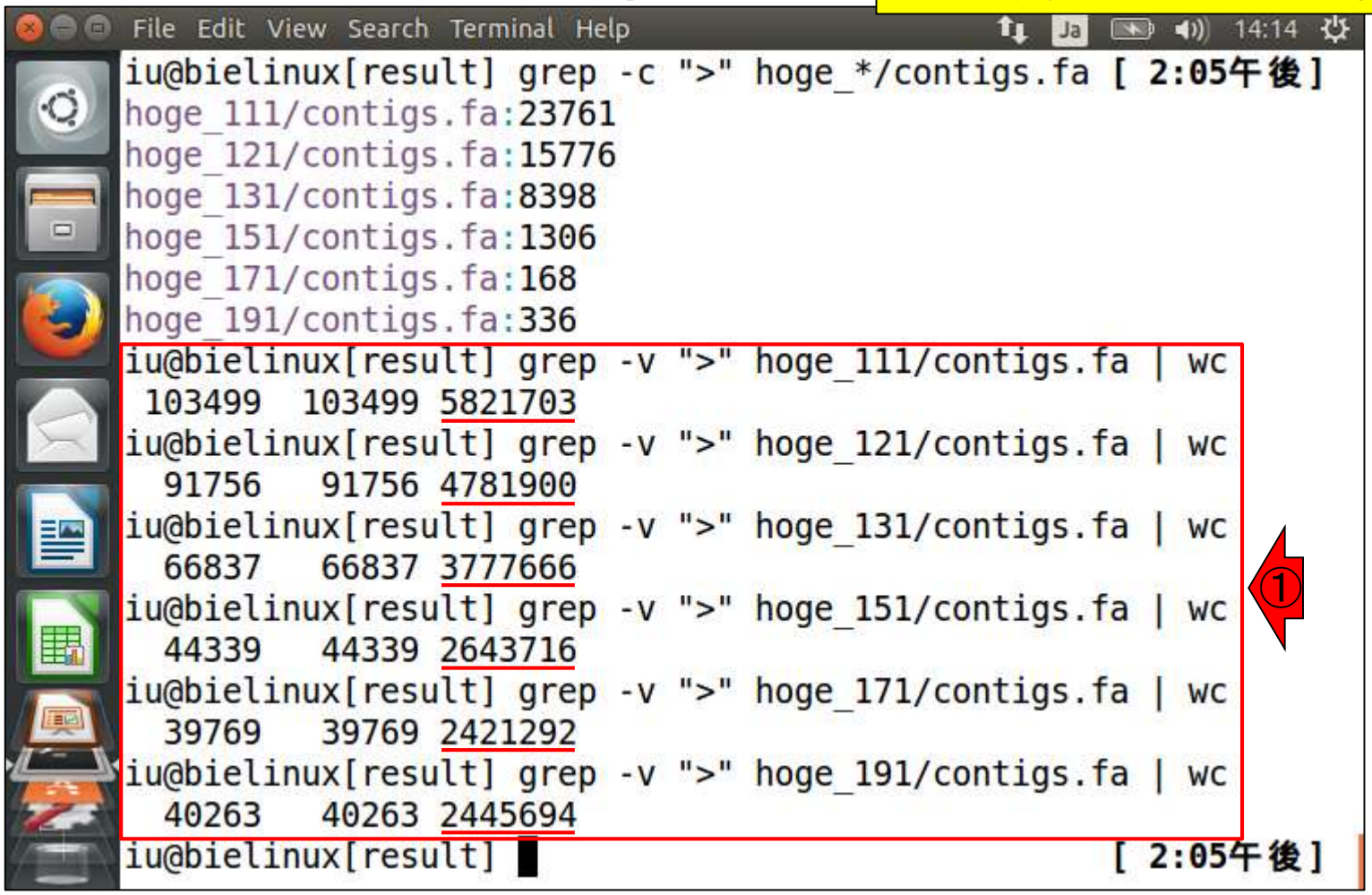
W10-5：再挑戦

①kの値を大きくすると、配列数（コンティグ数）は減少傾向となる。説明はオリジナルのウェブ資料と異なります。これは、W10-2からW10-4までを省略していることに起因します



W10-5：再挑戦

①ここは大まかなゲノムサイズ（総塩基数）を調べている（W8-2）。赤下線が該当部分。実際のゲノムサイズより、改行コード分だけわずかに大きな値になる



W10-6:これまでのまとめ

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

Velvetの場合は、①k値はリード長Lの2/3程度がよいのかもと学習する(実際には、ゲノムサイズやリード数に大きく依存する)。オリジナルのリード数(約300万×2)の1/10なので、1時間程度で概要を把握可能

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ったと仮定して「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

①

一般に、Velvetを利用する場合は、複数のk値でアセンブリを行う。主観でk=171がいいと判断したところまでが今日の内容で、原稿の①のあたり。2016年08月03日(2016.08.03)の内容の予告。もちろん客観的に最もよいと思われるk値を出力してくれるプログラムも存在する。その1つであるKmerGenieのインストールから、ゲノムサイズ推定ツールとしての利用へと展開していく。明日まで「待てっ!」