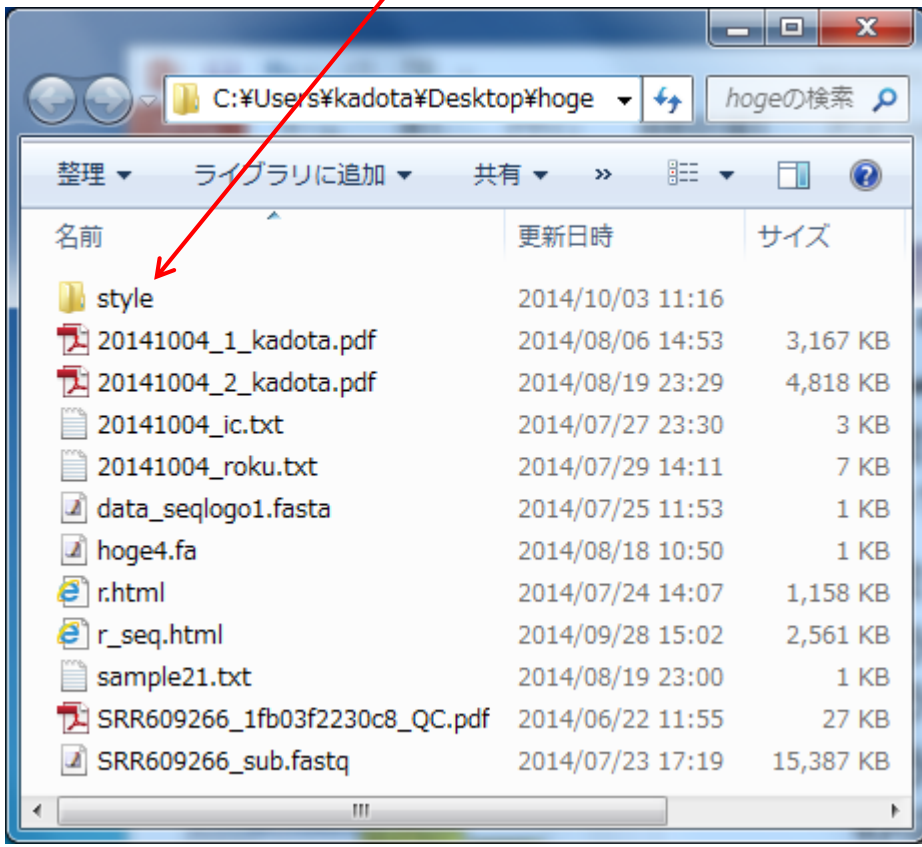
フリーソフトRを用いたビッグデータ解析:塩基配列解析を中心に

東京大学・大学院農学生命科学研究科
アグリバイオインフォマティクス教育研究ユニット
門田幸二(かどた　こうじ)
kadota@iu.a.u-tokyo.ac.jp
http://www.iu.a.u-tokyo.ac.jp/~kadota/

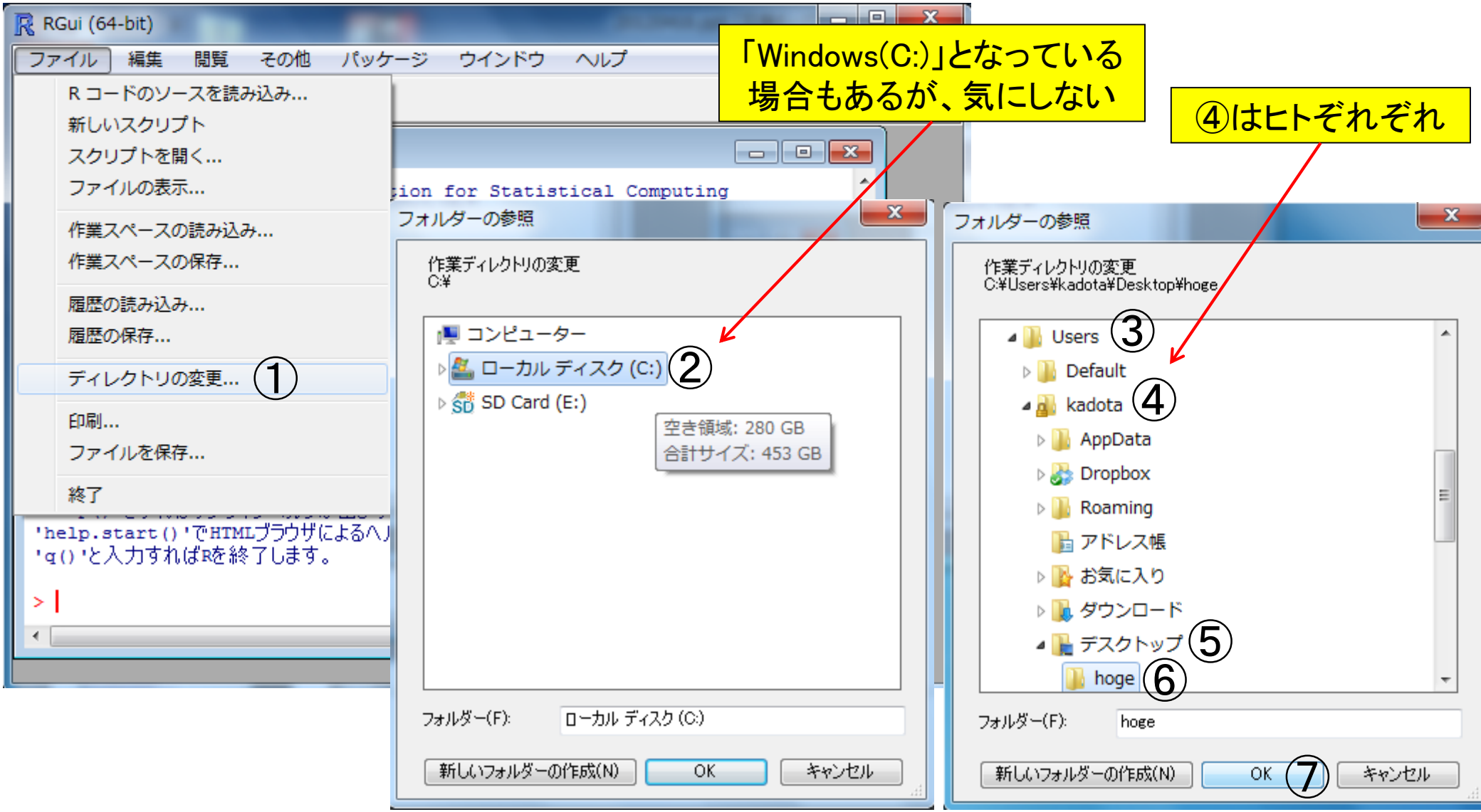
はじめに

- 講習会までにざっと目を通し、Rコードも実行しておいてください。
- 講習会当日は…
  - スライド22からスタート予定です。それ以前のスライドは自習。
  - スライド22以降でも右上に参考と書いてあるものは飛ばします。
  - 必要なパッケージはインストールされているものとします。また、作業ディレクトリの変更などの基本的な作業はできるようになっているものとします。
  - ネットワークの有無や不具合に影響されないように、hoge.zip中に2つのhtmlファイルを入れてあります。ダブルクリックすると普通に見られますのでご利用ください。
  - 右のように、デスクトップ上のhogeフォルダ中に下記ファイルが存在するという前提で行う。

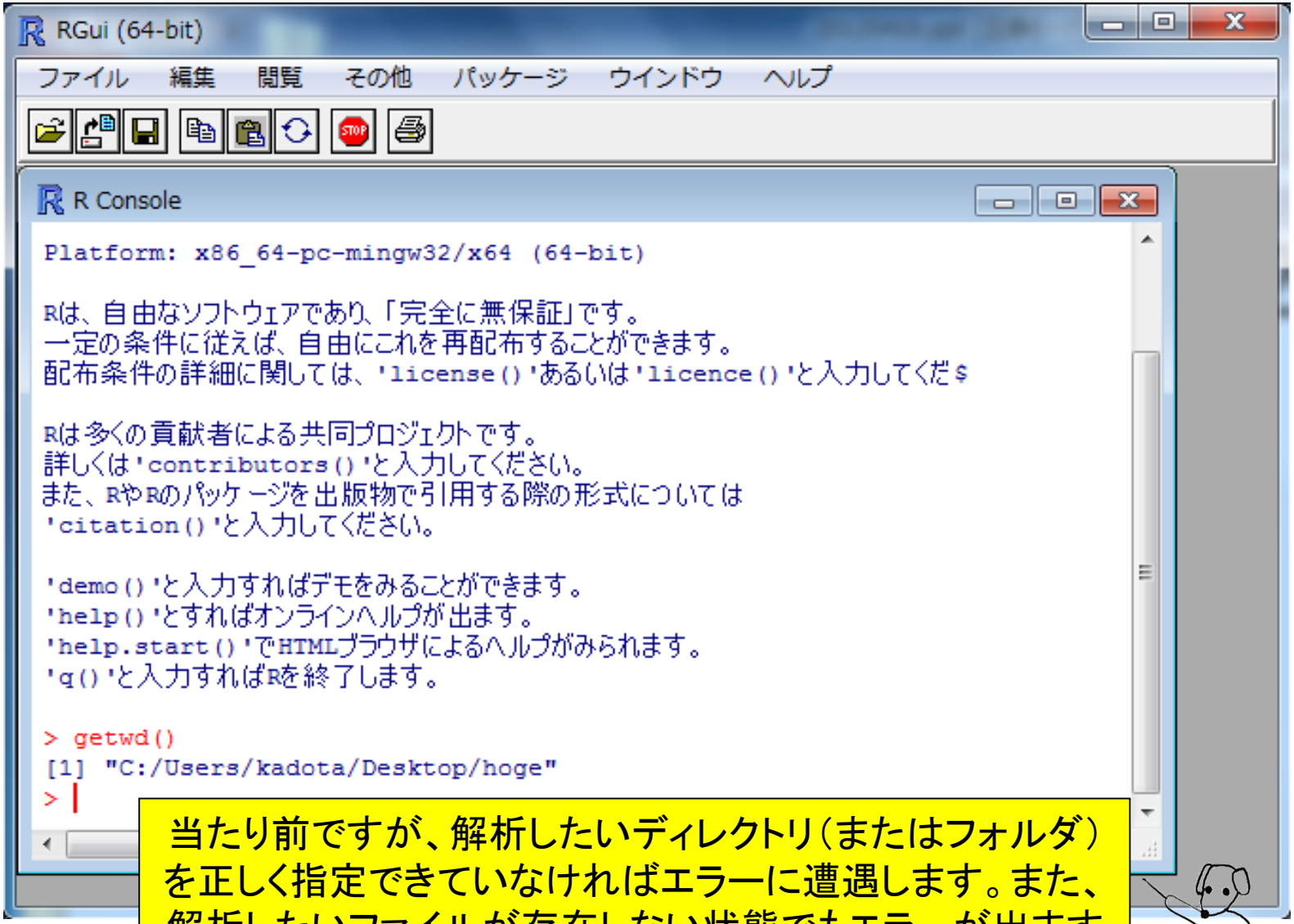
styleというフォルダをhogeフォルダ中にコピーしておくと、実際のhtmlと同じ見栄えになります。USBはスタッフから。



Rの起動と作業ディレクトリの変更



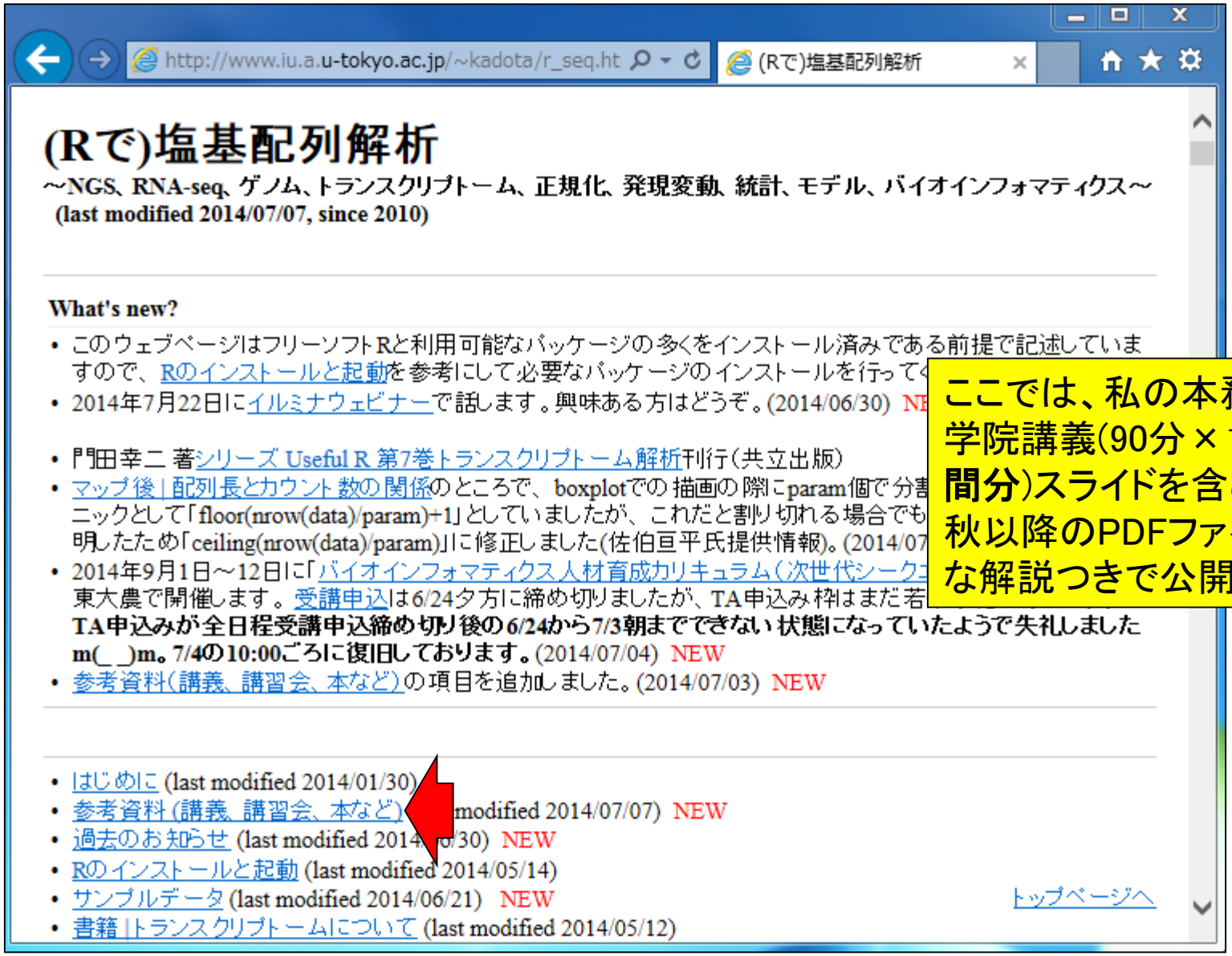
getwd()と打ち込んで確認



Contents

- **講義資料を取得(Rでできることの全体像を把握)**
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)

講義資料を取得



ここでは、私の本務である大学院講義(90分×18コマ=**27時間分**)スライドを含め、2013年秋以降のPDFファイルを簡単な解説つきで公開しています。

講義資料を取得

- はじめに (last modified 2014/01/30)
- 参考資料(講義、講習会、本など) (last modified 2014/07/07) NEW
- 過去のお知らせ (last modified 2014/06/30) NEW
- Rのインストールと起
- サンプルデータ (las
- バイオインフォマティ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ
- 書籍|トランスクリプ

参考資料(講義、講習会、本など) NEW

基本的に私門田の個人ページに記載してあるものです。かなり古い講演資料などの情報をもとに勉強されている方もいらっしゃるようですので、ここでは2013年秋以降の情報を載せておくとともに、大まかな内容についても述べておきます。講演予定のものについては、資料のアップは講演当日が基本です。50-100MB程度ありますがオリジナルのPowerPointファイルがほしい方はお気軽にリクエストしてください。講義資料としての利用などは事前連絡や謝辞も気にせずご自由にお使いください。

書籍

- **門田幸二**著(金明哲 編), シリーズ Useful R 第7巻トランスクリプトーム解析, 共立出版, 2014. ISBN: 978-4-320-12370-0
  内容:マイクロアレイとRNA-seq解析を例としてRを用いてトランスクリプトーム解析を
  が苦手なヒト向けに、重みつき平均の具体的な計算例などを挙げてオプションの意味
  構成にしてあります。書籍中のRコードは「書籍|トランスクリプトーム解析|...」をご覧
- **門田幸二**,「トランスクリプトミクスの推奨データ解析ガイドライン」, *ニュートリゲノミクス*
  シー出版, 45-52, 2013. ISBN: 978-4-7813-0820-3
  内容:マイクロアレイ解析の話がメインです。実験デザインの重要性を述べています。
  動遺伝子(DEG)検出法の組合せの重要性の話や、サンプル間クラスタリングである
  ます。MAS5データを用いる場合は特に倍率変化で議論することも無意味であること
  得られたマイクロアレイデータの場合には なぜ倍率変化でうまくいく傾向にあるかな

講習会、講義、講演資料

- **門田幸二**,「講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:農学生命情報科学特論I, 東京大学(東京), 2014.07.02
  内容:教科書の3.3節と4.3節周辺。マッピングプログラムは大きくbowtieなどのbasic aligner (unspliced aligner)とtophatなどのsplice-aware aligner (spliced aligner)に大別されること。splice-aware alignerの基本的なイメージ。ゲノム配列既知の場合の遺伝子構造推定としてTophat-Cufflinksパイプラインの基本形を紹介。既知遺伝子(または転写物)の発現解析でよい場合は、トランスクリプトーム配列へのマッピングでよい。最近はSailfishやRNA-Skimなど、k-merに基づくalignment-freeな方法が注目されていることなど。研究目的別留意点として、遺伝子間比較の場合とサンプル間比較の場合、配列長補正、総リード数補正、RPKMなど。長い転写物ほどマップされるリード数が多い傾向をRで確認。GSE42212のヒトRNA-seqデータのFASTQファイル取得以降の一通りの解析。実際に行ったのは、カウントデータ取得以降のTCCパッケージを用いたサンプル間クラスタリング、発現変動遺伝子(DEG)同定。M-A plotのおさらい。結果の解釈。FDR、分布やモデルの説明。倍率変化でDEG同定を行う場合との比較。2コマ(2×90 min)分。

ここでは、私の本務である大学院講義(90分×18コマ=**27時間分**)スライドを含め、2013年秋以降のPDFファイルを簡単な解説つきで公開しています。

R中心ですがトランスクリプトームデータ解析を一通り学びたい人は…

講義資料を取得

科目名:農学生命情報科学特論I
内容:公共DB、チェックサム、QC、前処理、k-mer、アセンブリ、マッピング、RPKM、発現変動など。
実施日:2014.06.18、2014.06.25、2014.07.02

科目名:機能ゲノム学
内容:データ取得、正規化、クラスタリング、発現変動解析、多重比較問題、機能解析など。
実施日:2014.05.14、2014.05.21、2014.05.28、2014.06.04

科目名:ゲノム情報解析基礎
内容:Rの基礎。GC含量計算やCpG解析、上流配列解析、Rのバージョンの違いなど。
実施日:2014.04.09、2014.04.23、2014.04.30

これら3科目の講義資料を順番にみていくとよい

講義資料を取得

(RobLoxBioC)の紹介および結果が変わらないことの確認までをやってもらった。2コマ(2×90 min)分。

- 門田幸二,「講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.30
  内容: Rで塩基配列解析を行うための基本的なところ。例題としてシロイヌナズナゲノムのCpG出現頻度を解析し考察。Rパッケージのインストール、エラーメッセージへの対処法、利用可能な関数の概観。sequence logosを主な講義内容とし、エントロピー計算や、なぜエントロピーをそのまま利用せずに情報量に変換するかの意義。subseq関数のオプションをうまく利用して効率的に目的のプロモーター配列領域を切り出して計算するやり方など。課題4はプログラムの一部を任意に変更する基礎的な能力を問うもの。他の例題の中に回答が存在するので、それを効率的に見つける能力を見ている。講義自体はスライド39までで、スライド40以降はうまくいかないこともあるという事例やRのバージョンの違いに気をつけろ的な話。「農学生命情報科学特論I」で改めて話す予定。1コマ(90 min)分。
- 門田幸二,「講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.23
  内容: Rで塩基配列解析を行うための基本的なところ。初心者が犯しがちなミス、プログラムの中身の説明、アノテーションファイルやmulti-FASTAファイルからの情報抽出、意図的にエラーを出させてエラーへの対処能力向上、GC含量計算やそのプログラム内部の説明、ヒトゲノムのCpG出現頻度を解析するための連続塩基出現頻度解析、BSgenomeパッケージとか。課題は、自分が解析したい入力ファイルの全体像を把握し、適切な列およびキーワードで効率よく情報収集するための練習問題レベルのものにしている。Rがいかに簡単であるかをわかってもらうことに重点を置いている。ただし、ヘッダー行でひっかけを作っており、目で見て明らかに回答がわかっている状況下でそれを正しく判断し適切なテンプレートプログラムを利用できるかを問うている。また、課題2では、ゲノム配列にもバージョンがあるということを認識してもらう。2コマ(2×90 min)分。
- 門田幸二,「ウェブページと講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.09
  内容: 初心者向けバイオインフォマティクス全般およびゲノム情報解析系のイントロダクションの話。Rのイントロダクションやこのウェブページの簡単な使い方を含む1コマ(90 min)分。
- 門田幸二,「比較トランスクリプトーム解析とその周辺」, よく分かる次世代シークエンサー解析ワークショップ
  内容: 初心者向けRNA-seqの話。主にカウントデータ

Rコード中身の詳細な説明もあります。

科目名:農学生命情報科学特論I
内容:公共DB、チェックサム、QC、前処理、k-mer、アセンブリ、マッピング、RPKM、発現変動など。
実施日:2014.06.18、2014.06.25、2014.07.02

科目名:機能ゲノム学
内容:データ取得、正規化、クラスタリング、発現変動解析、多重比較問題、機能解析など。
実施日:2014.05.14、2014.05.21、2014.05.28、2014.06.04

科目名:ゲノム情報解析基礎
内容:Rの基礎。GC含量計算やCpG解析、上流配列解析、Rのバージョンの違いなど。
実施日:2014.04.09、2014.04.23、2014.04.30

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

CpG解析(2014.04.23の講義資料)

(RobLoxBioC)の紹介および結果が変わらないことの確認までをやってもらった。2コマ(2×90 min)分。

- 門田幸二,「講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.30
  内容: Rで塩基配列解析を行うための基本的なところ。例題としてシロイヌナズナゲノムのCpG出現頻度を解析し考察。Rパッケージのインストール、エラーメッセージへの対処法、利用可能な関数の概観。sequence logosを主な講義内容とし、エントロピー計算や、なぜエントロピーをそのまま利用せずに情報量に変換するかの意義。subseq関数のオプションをうまく利用して効率的に目的のプロモーター配列領域を切り出して計算するやり方など。課題4はプログラムの一部を任意に変更する基礎的な能力を問うもの。他の例題の中に回答が存在するので、それを効率的に見つける能力を見ている。講義自体はスライド39までで、スライド40以降はうまくいかないこともあるという事例やRのバージョンの違いに気をつけろ的な話。「農学生命情報科学特論I」で話す予定。1コマ(90 min)分。
- 門田幸二,「講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.23
  内容: Rで塩基配列解析を行うための基本的なところ。初心者が犯しがちなミス、プログラムの中身の説明、アノテーションファイルやmulti-FASTAファイルからの情報抽出、意図的にエラーを出させてエラーへの対処能力向上、GC含量計算やそのプログラム内部の説明、ヒトゲノムのCpG出現頻度を解析するための連続塩基出現頻度解析、BSgenomeパッケージとか。課題は、自分が解析したい入力ファイルの全体像を把握し、適切な列およびキーワードで効率よく情報収集するための練習問題レベルのものにしている。Rがいかに簡単であるかをわかってもらうことに重点を置いている。ただし、ヘッダー行でひっかけを作っており、目で見て明らかに回答がわかっている状況下でそれを正しく判断し適切なテンプレートプログラムを利用できるかを問うている。また、課題2では、ゲノム配列にもバージョンがあるということを認識してもらう。2コマ(2×90 min)分。
- 門田幸二,「ウェブページと講義資料」, アグリバイオインフォマティクス教育研究プログラムの大学院講義科目:ゲノム情報解析基礎, 東京大学(東京), 2014.04.09
  内容: 初心者向けバイオインフォマティクス全般およびゲノム情報解析系のイントロダクションの話。Rのイントロダクションやこのウェブページの簡
- 門田幸二,「比較トランスクリプトーム解析とその周辺
  よく分かる次世代シークエンサー解析ワークショップ
  内容: 初心者向けRNA-seqの話。主にカウントデータ

Rコード中身の詳細な説明もあります。

科目名:農学生命情報科学特論I
内容:公共DB、チェックサム、QC、前処理、k-mer、アセンブリ、マッピング、RPKM、発現変動など。
実施日:2014.06.18、2014.06.25、2014.07.02

科目名:機能ゲノム学
内容:データ取得、正規化、クラスタリング、発現変動解析、多重比較問題、機能解析など。
実施日:2014.05.14、2014.05.21、2014.05.28、2014.06.04

科目名:ゲノム情報解析基礎
内容:Rの基礎。GC含量計算やCpG解析、上流配列解析、Rのバージョンの違いなど。
実施日:2014.04.09、2014.04.23、2014.04.30

ヒトゲノム中のCpG出現確率は低い

- 全部で16通りの2連続塩基の出現頻度分布を調べると、CGとなる確率の実測値(0.986%)は期待値(4.2%)よりもかなり低い
- 期待値
  - ゲノム中のGC含量を考慮した場合：約41%(A:0.295, C:0.205, G: 0.205, T:0.295)なので、0.205 × 0.205= 4.2%
  - ゲノム中のGC含量を考慮しない場合： 50%(A:0.25, C:0.25, G: 0.25, T:0.25)なので、0.25 × 0.25= 6.25%
- *k*連続塩基の組合せは4^k通り
  - 2連続塩基の場合は4^2 = 16通り
    - AA, AC, AG, AT, CA, CC, **CG**, CT, GA, GC, GG, GT, TA, TC, TG, TT
  - 3連続塩基の場合は4^3 = 64通り
    - AAA, AAC, AAG, AAT, ACA, ACC, …, TGG, TGT, TTA, TTC, TTG, TTT
- CpG解析（CGの結果を他と比較）
  - 入力：ヒトゲノム配列のmulti-FASTA形式ファイル（またはRパッケージ）
  - 出力：16種類の連続塩基の染色体ごとの出現頻度（または出現確率）

BSgenome.Hsapiens.NCBI.GRCh38というヒトゲノム情報を含むRパッケージを入力としてCpG解析を行う

(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、統計、モデル、バイオインフォマティクス～

(last m

What's

- この
- 2014
- 2014
- 2014
- 門田
- はじめ

- イントロ | 一般 | 2連続塩基の出現頻度情報を取得

- イントロ | 一般 | 逆相補鎖(reverse complement)を取得(last modified 2013/06/14)
- イントロ | 一般 | 逆鎖(reverse)を取得(last modified 2013/06/14)
- イントロ | 一般 | 2連続塩基の出現頻度情報を取得(last modified 2014/04/14)
- イントロ | 一般 | 3連続塩基の出現頻度情報を取得(last modified 2013/06/14)
- イントロ | 一般 | 任意の長さの連続塩基の出現頻度情報を取得(last modified 2013/06/14)

イントロ | 一般 | 2連続塩基の出現頻度情報を取得

multi-FASTA形式ファイルを読み込んで、"AA", "AC", "AG", "AT", "CA", "CC", "CG", "CT", "GA", "GC", "GG", "GT", "TA", "TC", "TG", "TT"の計4^2 = 16通りの2連続塩基の出現頻度を調べるやり方を示します。

ヒトゲノムで"CG"の割合が期待値よりも低い(Lander et al., 2001; Saxonov et al., 2006)ですが、それを簡単に検証できる

「ファ

1. イント

タイトル

7. BSgenomeパッケージ中のヒトゲノム配列("BSgenome.Hsapiens.NCBI.GRCh38")の場合:

2013年12月にリリースされたGenome Reference Consortium GRCh38です。出力は出現確率です。

```
out_f <- "hoge7.txt"                          #出力ファイル名を指定してout_fに格納
param <- "BSgenome.Hsapiens.NCBI.GRCh38"#パッケージ名を指定

#必要なパッケージをロード
library(Biostrings)                           #パッケージの読み込み
library(param, character.only=T)              #paramで指定したパッケージの読み込み

#前処理(paramで指定したパッケージ中のオブジェクト名をgenomeに統一)
tmp <- ls(paste("package", param, sep=":"))#paramで指定したパッケージで利用可能なオ
genome <- eval(parse(text=tmp))               #文字列tmpをRオブジェクトとしてgenomeに格納(
fasta <- getSeq(genome)                       #ゲノム塩基配列情報を抽出した結果をfastaに格納
names(fasta) <- seqnames(genome)              #description情報を追加している
fasta                                         #確認してるだけです

#本番
out <- dinucleotideFrequency(fasta, as.prob=T)#2連続塩基の出現確率情報をoutに格納

#ファイ
tmp <-                                        に出現頻度情報のout
write.                                        #tmpの中身を指定し
```

BSgenome.Hsapiens.NCBI.GRCh38というヒトゲノム情報を含むRパッケージを入力としてCpG解析を行う

(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、統計、モデル、バイオインフォマティクス~

(last m

- イントロ | 一般 | 2連続塩基の出現頻度情報を取得

What's

- この
- 2014
- 2014
- 門田
- 参考
- はじめ

- イントロ | 一般 | 逆相補鎖(reverse complement)を取得(last modified 2013/06/14)
- イントロ | 一般 | 逆鎖(reverse)を取得(last modified 2013/06/14)
- イントロ | 一般 | 2連続塩基の出現頻度情報を取得(last modified 2014/04/14)
- イントロ | 一般 | 3連続塩基の出現頻度情報を取得(last modified 2013/06/14)
- イントロ | 一般 | 任意の長さの連続塩基の出現頻度情報を取得(last modified 2013/06/14)

イントロ | 一般 | 2連続塩基の出現頻度情報を取得

multi-FASTA形式ファイルを読み込んで、"AA", "AC", "AG", "AT", "CA", "CC", "CG", "CT", "GA", "GC", "GG", "GT", "TA", "TC", "TG", "TT"の計4^2 = 16通りの2連続塩基の出現頻度を調べるやり方を示します。

ヒトゲノムで"CG"の割合が期待値よりも低い(Lander et al., 2001; Saxonov et al., 2006)ですが、それを簡単に検証できる

「ファ

1. イント

タイトル

7. BSgenomeパッケージ中のヒトゲノム配列("BSgenome.Hsapiens.NCBI.GRCh38")の場合:

2013年12月にリリースされたGenome Reference Consortium GRCh38です。出力は出現確率です。

```
out_f <- "hoge7.txt"                          #出力ファイル名を指定してout_fに格納
param <- "BSgenome.Hsapiens.NCBI.GRCh38"#パッケージ名を指定

#必要なパッケージをロード
library(Biostrings)                           #パッケ
library(param, character.only=T)              #param

#前処理(paramで指定したパッケージ中のオブジェクト
tmp <- ls(paste("package", param, sep=":"))#p
genome <- eval(parse(text=tmp))               #文字列
fasta <- getSeq(genome)                       #ゲノム
names(fasta) <- seqnames(genome)              #description情報を追加している
fasta                                         #確認してるだけです

#本番
out <- dinucleotideFrequency(fasta, as.prob=T)#2連続塩基の出現確率情報をoutに格納

#ファイルに保存
tmp <- cbind(names(fasta), out)               #最初の列にID情報、そのあとに出現頻度情報のou
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定
```

入力:BSgenome.Hsapiens.NCBI.GRCh38というヒトゲノム情報を含むRパッケージ

出力:16種類の連続塩基の染色体ごとの出現確率情報を含むhoge7.txtというタブ区切りテキストファイル

基本はコピペ

WindowsのヒトはCTRLとALTキーを押しながらコードの枠内で左クリックすると全選択できます

7. BSgenomeパッケージ中のヒトゲノム配列("BSgenome.Hsapiens.NCBI.GRCh38")の場合:

2013年12月にリリースされたGenome Reference Consortium GRCh38です。出力は出現確率です。

```
out_f <- "hoge7.txt"                                    #出力ファイル名を指定してout_fに格納
param <- "BSgenome.Hsap                                 定

#必要なパッケージをロード
library(Biostrings)
library(param, characte

#前処理(paramで指定した
tmp <- ls(paste("packag
genome <- eval(parse(te
fasta <- getSeq(genome)
names(fasta) <- seqname
fasta

#本番
out <- dinucleotideFrec

#ファイルに保存
tmp <- cbind(names(fast
write.table(tmp, out_f, sep="\t", ap
```

① 切り取り(T) / コピー(C) / 貼り付け / すべて選択(A) / 印刷(I)... / 印刷プレビュー / Bing でマップ / Bing で翻訳 / Google で検索 / 電子メール (W / すべてのアク / Send to One

RGui (64-bit)

ファイル 編集 閲覧 その他 パッケージ ウインドウ ヘルプ

R Console

```
Platform: x86_64-pc-mingw32/x64 (64-bit)

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

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

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

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

② コピー Ctrl+C / ペースト Ctrl+V / コマンドのみペースト / コピー&ペースト Ctrl+X / ウインドウの消去 Ctrl+L / 全て選択 / バッファに出力 Ctrl+W / ウィンドウを常にトップに置く

①一連のコマンド群をコピーして
②R Console画面上でペースト

実行結果

エラーなく実行できると、出力ファイルとして指定したhoge7.txtが作業ディレクトリ中に生成される。

7. BSgenomeパッケージ中のヒトゲノム配列("BSgenome.Hsapiens.NCBI.GRCh38")の場合:

2013年12月にリリースされたGenome Reference Consortium GRCh38です。出力は出現確率で

```
out_f <- "hoge7.txt"
param <- "BSgenome.Hsapiens.NCBI.GR

#必要なパッケージをロード
library(Biostrings)
library(param, character.only=T)

#前処理(paramで指定したパッケージ中の
tmp <- ls(paste("package", param, s
genome <- eval(parse(text=tmp))
fasta <- getSeq(genome)
names(fasta) <- seqnames(genome)
fasta

#本番
out <- dinucleotideFrequency(fasta,

#ファイルに保存
tmp <- cbind(names(fasta), out)
write.table(tmp, out_f, sep="\t", a
```

RGui (64-bit)

ファイル 編集 閲覧 その他 パッケージ ウインドウ ヘルプ

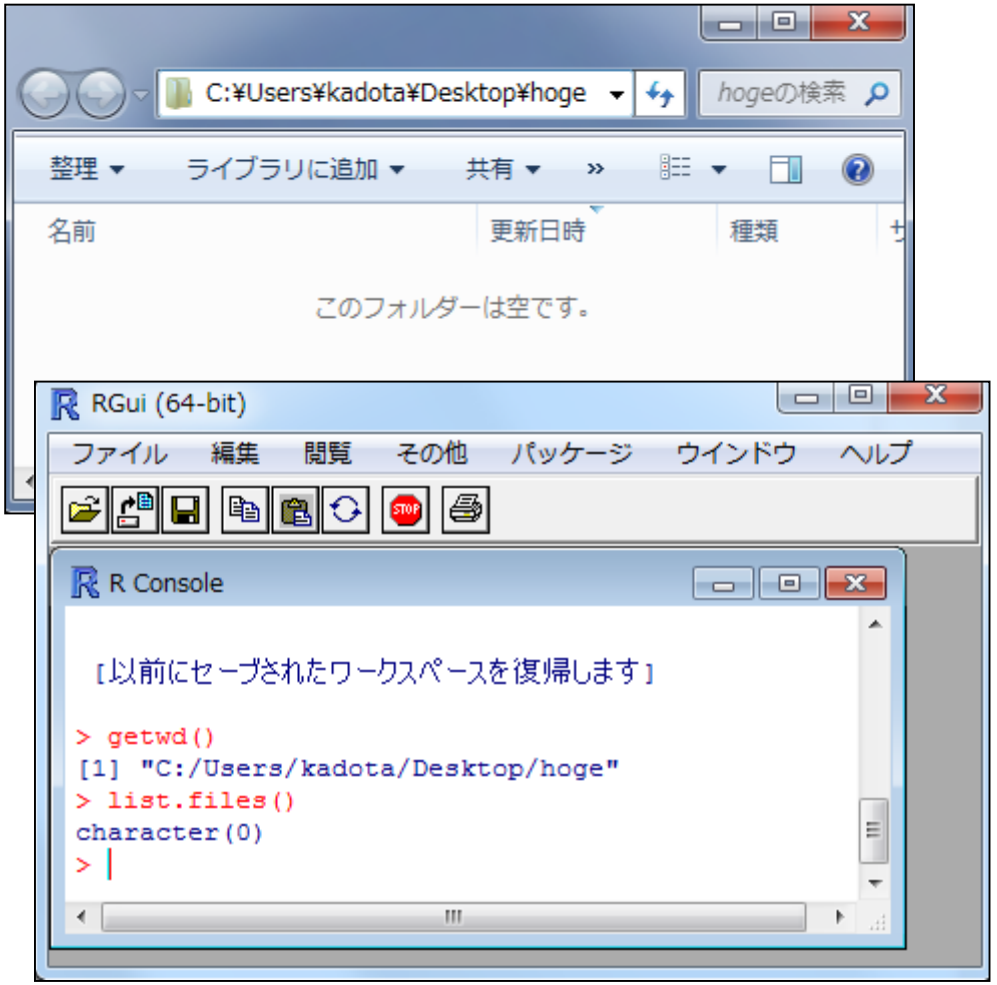
R Console

```
> names(fasta) <- seqnames(genome)        #description情報を追加している
> fasta                                   #確認してるだけです
  A DNAStringSet instance of length 455
          width seq                                                 names
  [1] 248956422 NNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNN 1
  [2] 242193529 NNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNN 2
  [3] 198295559 NNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNN 3
  [4] 190214555 NNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNN 4
  [5] 181538259 NNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNN 5
  ...       ... ...
[451]    200773 TCTACTCTCCCATGCTTGC...AGGACTCATGGGGAATTC HSCHR19KIR_FH08_B...
[452]    170148 TTTCTTTCTTTTTTTTTTT...AGGACTCATGGGGAATTC HSCHR19KIR_FH13_A...
[453]    215732 TGTGGTGAGGACCCTTAAG...AGGACTCATGGGGAATTC HSCHR19KIR_FH13_B...
[454]    170537 TCTACTCTCCCATGCTTGC...AGGACTCATGGGGAATTC HSCHR19KIR_FH15_A...
[455]    177381 GATCTATCTGTATCTCCAC...AGGACTCATGGGGAATTC HSCHR19KIR_RP5_B_...
>
> #本番
> out <- dinucleotideFrequency(fasta, as.prob=T)#2連続塩基の出現確率情報をou$
>
> #ファイルに保存
> tmp <- cbind(names(fasta), out)         #最初の列にID情報、そのあとに出現頻$
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中$
>
```

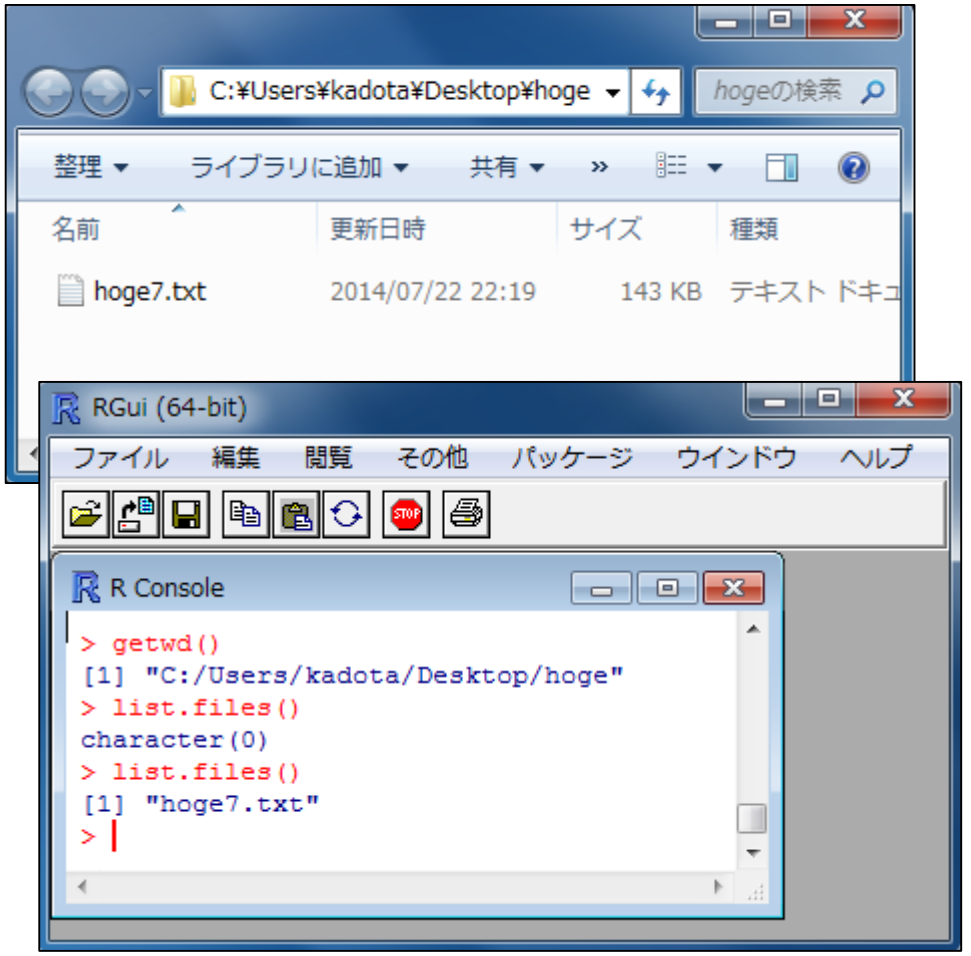
2分強かかります

実際のhogeフォルダとR操作画面の関係

ファイル保存前



ファイル保存後



list.files関数は作業ディレクトリ中のファイル名を表示

2連続塩基の出現確率:ヒトゲノム

7. BSgenomeパッケージ中のヒトゲノム配列("BSgenome.Hsapiens.NCBI.GRCh38")の場合:

2013年12月にリリースされたGenom

```
out_f <- "hoge7.txt"
param <- "BSgenome.Hsapien

#必要なパッケージをロード
library(Biostrings)
library(param, character.c

#前処理(paramで指定したパッ
tmp <- ls(paste("package",
genome <- eval(parse(text=
fasta <- getSeq(genome)
names(fasta) <- seqnames(g
fasta

#本番
out <- dinucleotideFrequen

#ファイルに保存
tmp <- cbind(names(fasta),
write.table(tmp, out_f, se
```

出力: hoge7.txt

	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
1	9.5%	5.0%	7.1%	7.4%	7.3%	5.4%	1.0%	7.1%	6.0%	4.4%	5.4%	5.0%	6.3%	6.0%	7.3%	9.6%
2	10.0%	5.0%	7.0%	7.9%	7.2%	5.0%	0.9%	7.0%	5.9%	4.1%	5.0%	5.0%	6.7%	5.9%	7.2%	10.0%
3	10.1%	5.0%	6.9%	8.0%	7.2%	4.9%	0.8%	6.9%	5.9%	4.0%	4.9%	5.0%	6.9%	5.9%	7.2%	10.2%
4	10.6%	5.0%	6.7%	8.5%	7.1%	4.5%	0.8%	6.7%	5.9%	3.8%	4.5%	5.0%	7.3%	5.8%	7.1%	10.6%
5	10.2%	5.0%	6.9%	8.1%	7.2%	4.8%	0.9%	6.9%	5.9%	4.0%	4.8%	5.1%	6.9%	5.9%	7.2%	10.3%
6	10.2%	5.0%	6.9%	8.1%	7.2%	4.8%	0.9%	6.9%	5.9%	4.0%	4.9%	5.0%	6.9%	5.9%	7.2%	10.2%
7	9.8%	5.0%	7.0%	7.7%	7.3%	5.1%	1.0%	7.0%	6.0%	4.2%	5.1%	5.1%	6.5%	5.9%	7.3%	10.0%
8	10.0%	5.1%	6.9%	7.9%	7.2%	5.0%	0.9%	6.9%	6.0%	4.1%	5.0%	5.0%	6.7%	5.9%	7.2%	10.0%
9	9.7%	5.1%	7.0%	7.6%	7.3%	5.3%	1.0%	7.0%	6.0%	4.3%	5.3%	5.0%	6.4%	6.0%	7.3%	9.7%
10	9.6%	5.0%	7.1%	7.5%	7.3%	5.3%	1.0%	7.1%	6.0%	4.4%	5.3%	5.1%	6.3%	6.0%	7.4%	9.7%
11	9.5%	5.1%	7.1%	7.5%	7.3%	5.3%	1.0%	7.1%								9.6%
12	9.8%	5.0%	7.0%	7.7%	7.2%	5.1%	1.0%	7.0%								9.9%
13	10.5%	5.0%	6.8%	8.4%	7.1%	4.5%	0.9%	6.7%								10.6%
14	9.7%	5.0%	7.0%	7.7%	7.2%	5.1%	1.0%	7.0%								9.9%
15	9.4%	5.1%	7.1%	7.3%	7.3%	5.4%	1.1%	7.1%	6.0%	4.5%	5.5%	5.1%	6.1%	6.0%	7.4%	9.5%
16	8.6%	5.1%	7.3%	6.7%	7.5%	6.1%	1.4%	7.2%	6.1%	5.0%	6.1%	5.1%	5.4%	6.1%	7.6%	8.8%
17	8.5%	5.1%	7.3%	6.4%	7.4%	6.3%	1.5%	7.4%	6.2%	5.1%	6.4%	5.0%	5.2%	6.1%	7.5%	8.6%
18	10.1%	5.1%	7.0%	7.9%	7.2%	4.7%	0.9%	6.9%	6.1%	4.0%	4.9%	5.1%	6.7%	5.9%	7.3%	10.3%
19	7.7%	5.1%	7.5%	5.7%	7.5%	7.0%	1.9%	7.4%	6.2%	5.6%	7.1%	5.2%	4.5%	6.2%	7.6%	7.9%
20	8.8%	5.0%	7.3%	6.8%	7.5%	5.8%	1.2%	7.3%	6.2%	4.8%	6.0%	5.1%	5.5%	6.1%	7.6%	9.1%
21	9.8%	5.1%	6.9%	7.7%	7.3%	5.1%	1.2%	6.9%	6.0%	4.3%	5.1%	5.1%	6.4%	6.0%	7.3%	9.9%
22	7.9%	5.1%	7.5%	6.0%	7.6%	6.7%	1.6%	7.4%	6.2%	5.5%	6.8%	5.1%	4.7%	6.1%	7.7%	8.0%
X	10.1%	5.1%	6.8%	8.2%	7.2%	4.8%	0.9%	6.8%	6.0%	3.9%	4.9%	5.1%	6.9%	5.9%	7.3%	10.2%
Y	9.8%	5.1%	6.8%	8.1%	7.4%	4.9%	0.9%	6.8%	6.0%	3.9%	4.9%	5.2%	6.6%	6.1%	7.5%	10.0%
…																

確かにCGが期待値(4.2%)よりも低いことがわかります

2連続塩基の出現頻度：基本形

イントロ | 一般 | 2連続塩基の出現頻度情報を取得 NEW

multi-fasta形式ファイルを読み込んで、"AA", "AC", "AG", "AT", "CA", "CC", "CG", "CT", "GA", "GC", "GG", "GT", "TA", "TC", "TG", "TT"の計$4^2 = 16$通りの2連続塩基の出現頻度を調べるやり方を示します。
ヒトゲノムで"CG"の割合が期待値よりも低い(Lander et al., 2001; Saxonov et al., 2006)ですが、それを簡単に検証できます。
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し以下をコピペ。

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:

タイトル通りの出現頻度です。

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

#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

#本番
out <- dinucleotideFrequency(fasta)         #2連続塩基の出現頻度情報をoutに格

#ファイルに保存
tmp <- cbind(names(fasta), out)             #最初の列にID情報、そのあとに出現
write.table(tmp
```

hoge4.fa - メモ帳

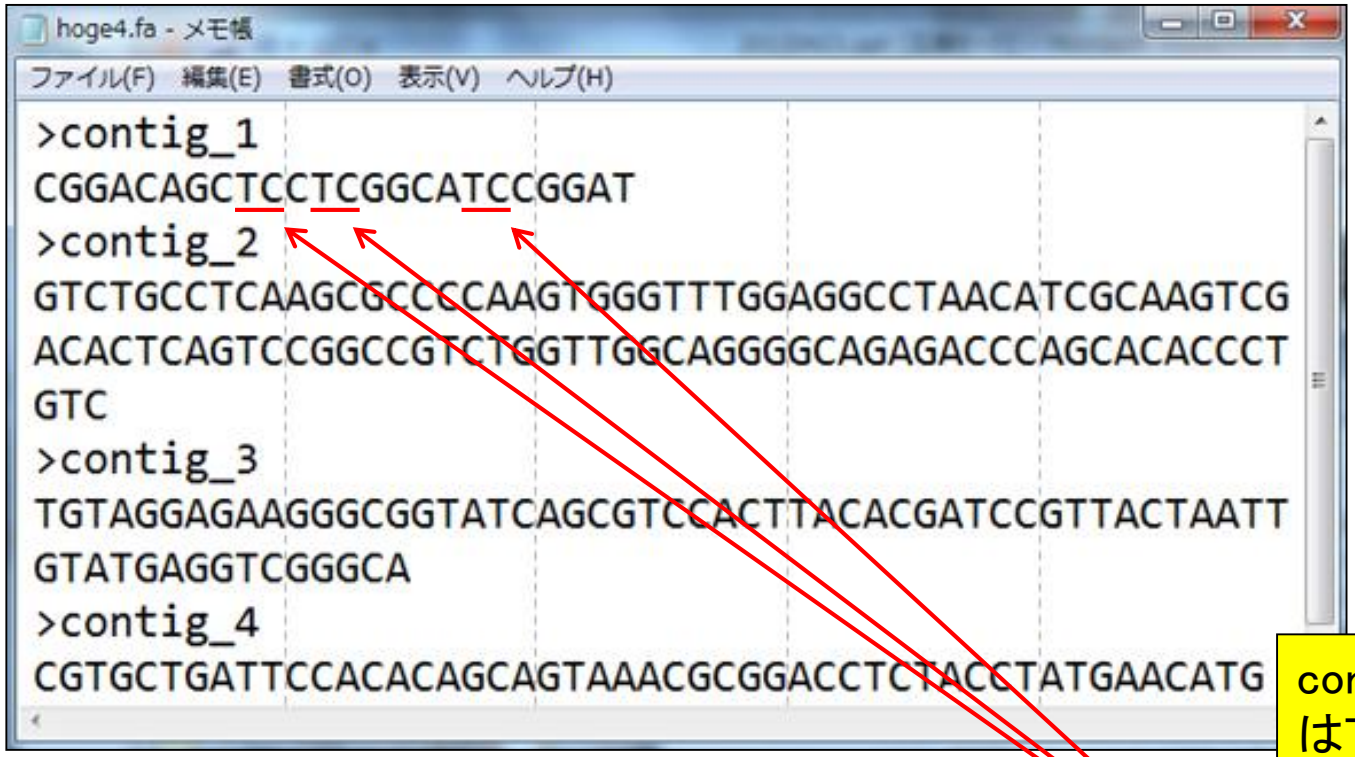
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
>contig_1
CGGACAGCTCCTCGGCATCCGGAT
>contig_2
GTCTGCCTCAAGCGCCCCAAGTGGGTTTGGAGGCCTAACATCGCAAGTCG
ACACTCAGTCCGGCCGTCTGGTTGGCAGGGGCAGAGACCCAGCACACCCT
GTC
>contig_3
TGTAGGAGAAGGGCGGTATCAGCGTCCACTTACACGATCCGTTACTAATT
GTATGAGGTCGGGCA
>contig_4
CGTGCTGATTCCACACAGCAGTAAACGCGGACCTCTACCTATGAACATG
```

出力：hoge1.txt

	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
contig_1	0	1	1	2	2	2	3	2	2	2	3	0	0	3	0	0
contig_2	4	6	9	1	11	11	5	6	4	9	10	8	1	8	6	3
contig_3	2	4	5	4	4	2	5	2	4	3	7	6	6	4	3	3
contig_4	3	6	2	3	5	3	3	4	3	3	1	2	3	2	4	1

2連続塩基の出現頻度：基本形



contig_1の塩基配列中にはTCという2連続塩基が3つ存在するということ

出力：hoge1.txt

	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
contig_1	0	1	1	2	2	2	3	2	2	2	3	0	0	3	0	0
contig_2	4	6	9	1	11	11	5	6	4	9	10	8	1	8	6	3
contig_3	2	4	5	4	4	2	5	2	4	3	7	6	6	4	3	3
contig_4	3	6	2	3	5	3	3	4	3	3	1	2	3	2	4	1

2連続塩基の出現確率：基本形

2. 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:

出現頻度ではなく、出現確率を得るやり方です。

```
in_f <- "hoge4.fa"                          #入力ファイル名を指定してin_fに格納
out_f <- "hoge2.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

#本番
out <- dinucleotideFrequency(fasta, as.prob=T)#2連続塩基の出現確率情報を

#ファイルに保存
tmp <- cbind(names(fasta), out)             #最初の列にID情報、そのあとに出現
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmp
```

hoge4.fa - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
>contig_1
CGGACAGCTCCTCGGCATCCGGAT
>contig_2
GTCTGCCTCAAGCGCCCCAAGTGGGTTTGGAGGCCTAACATCGCAAGTCG
ACACTCAGTCCGGCCGTCTGGTTGGCAGGGGCAGAGACCCAGCACACCCT
GTC
>contig_3
TGTAGGAGAAGGGCGGTATCAGCGTCCACTTACACGATCCGTTACTAATT
GTATGAGGTCGGGCA
>contig_4
CGTGCTGATTCCACACAGCAGTAAACGCGGACCTCTACCTATGAACATG
```

出力：hoge2.txt

	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
contig_1	0.0%	4.3%	4.3%	8.7%	8.7%	8.7%	13.0%	8.7%	8.7%	8.7%	13.0%	0.0%	0.0%	13.0%	0.0%	0.0%
contig_2	3.9%	5.9%	8.8%	1.0%	10.8%	10.8%	4.9%	5.9%	3.9%	8.8%	9.8%	7.8%	1.0%	7.8%	5.9%	2.9%
contig_3	3.1%	6.3%	7.8%	6.3%	6.3%	3.1%	7.8%	3.1%	6.3%	4.7%	10.9%	9.4%	9.4%	6.3%	4.7%	4.7%
contig_4	6.3%	12.5%	4.2%	6.3%	10.4%	6.3%	6.3%	8.3%	6.3%	6.3%	2.1%	4.2%	6.3%	4.2%	8.3%	2.1%

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- **Sequence logos(ポジションごとに特徴的な塩基を強調表示)**
  - **イントロダクション**
    - **small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較**
    - **マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握**
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

NGSデータ解析とR

塩基配列データ取得
↓
QCやpreprocessing
↓
マッピング
↓
カウント情報取得
↓
発現変動解析

(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 2014/07/14, since 2010)

- アセンブル | について (last modified 2014/06/20) NEW
- アセンブル | ゲノム用 (last modified 2014/07/08) NEW
- アセンブル | トランスクリプトーム(転写物)用 (last modified 2014/07/08) NEW
- マッピング | について (last modified 2014/07/09) NEW
- マッピング | basic aligner (last modified 2014/07/09) NEW
- マッピング | splice-aware aligner (last modified 2014/07/09) NEW
- マッピング | Bisulfite sequencing用 (last modified 2014/07/09) NEW
- マッピング | (ESTレベルの長さの)contig (last modified 2014/06/24) NEW
- 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(Lerch XXX) (last modified 201
- マッピング | single-end | ゲノム | basic aligner(応用) | QuasR(Lerch XXX) (last modified 201
- マッピング | single-end | ゲノム | splice-aware aligner | QuasR(Lerch XXX) (last modified 20
- 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) NEW
- 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 2014/03/12)
- マップ後 | カウント情報取得 | ゲノム | アノテーション有 | QuasR(Lerch XXX) (last modified 2013/
- マップ後 | カウント情報取得 | ゲノム | アノテーション無 | QuasR(Lerch XXX) (last modified 2014/
- マップ後 | カウント情報取得 | トランスクリプトーム | BEDファイルから (last modified 2014/06/21)
- マップ後 | 配列長とカウント数の関係 (last modified 2014/07/03) NEW

クオリティの低いリードの除去（フィルタリング）やアダプター配列の除去もできます。特にアダプター配列除去はsmall RNA-seqマッピング結果に大きな影響を及ぼす。

small RNA-seqデータのマッピング

- マッピング | single-end | ゲノム | basic aligner(基礎) | QuasR(Lerch_XXX)(last modified 2013/10/25)
- マッピング | single-end | ゲノム | basic aligner(応用) | QuasR(Lerch_XXX)(last modified 2013/10/25)
- マッピング | single-end | ゲノム | splice-aware aligner | QuasR(Lerch_XXX)(last modified 2013/10/25)

マッピング | single-end | ゲノム | basic aligner(応用) | QuasR(Lerch_XXX) NEW

QuasRパッケージを用いてsingle-end RNA-seqデータのリファレンスゲノム配列へのマッピングを行うやり方を示します。basic alignerの一つ

6. 2つのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gzとhoge4.fastq.gz)のカイコゲノム(integretedseq.fa)へのマッピングの場合(mapping_single_genome8.txt):

small RNA-seqデータ(400Mb弱; 11928428リード; Nie et al., BMC Genomics, 2013)です。イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたものがSRR609266.fastq.gzです。また、前処理 | トリミング | アダプター配列除去(基礎) | ShortRead(Morgan 2009)の4を実行して得られたものがhoge4.fastq.gzです。カイコゲノム配列は、農業生物資源研究所(NIAS)が提供しているカイコゲノム配列のウェブページからIntegrated sequences (integretedseq.txt.gz)をダウンロードしました。ファイル名は"integretedseq.txt"となりますが、拡張子を".txt"から".fa"に変更して、"integrete
かかります。

```
in_f1 <- "mapping_single_genome8.txt"  #入力ファイル名を指定してin_f1に
in_f2 <- "integretedseq.fa"            #入力ファイル名を指定してin_f2に
param_mapping <- "-m 1 --best --strata -v 2"#マッピング時のオプション

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

#本番(マッピング)
time_s <- proc.time()                  #計算時間を計測するため
out <- qAlign(in_f1, in_f2, alignmentParameter=param_mapping)#マッピングを行うqAlign関数を実行した結果を
time_e <- proc.time()                  #計算時間を計測するため
time_e - time_s                        #計算時間を表示(一番右側の数字。単位はsecond)
out                                    #マッピングに用いたパラメータや入力ファイルの情報などを表示
alignmentStats(out)                    #マッピング結果(alignment statistics)の表示。seqlength：リファレ
```

リファレンス(カイコゲノム配列)とアダプター配列除去前後のsmall RNA-seqファイルを入力として、Rでマッピング。**実習ではやりません。**マッピング後に得られる*_QC.pdfファイルのみhogeフォルダにあります。

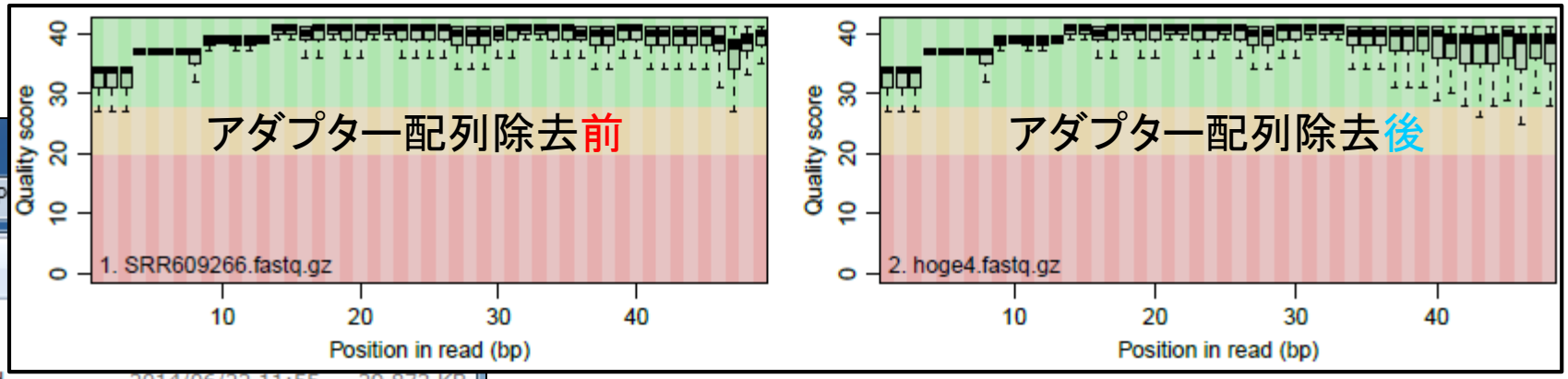
small RNA-seqデータのマッピング

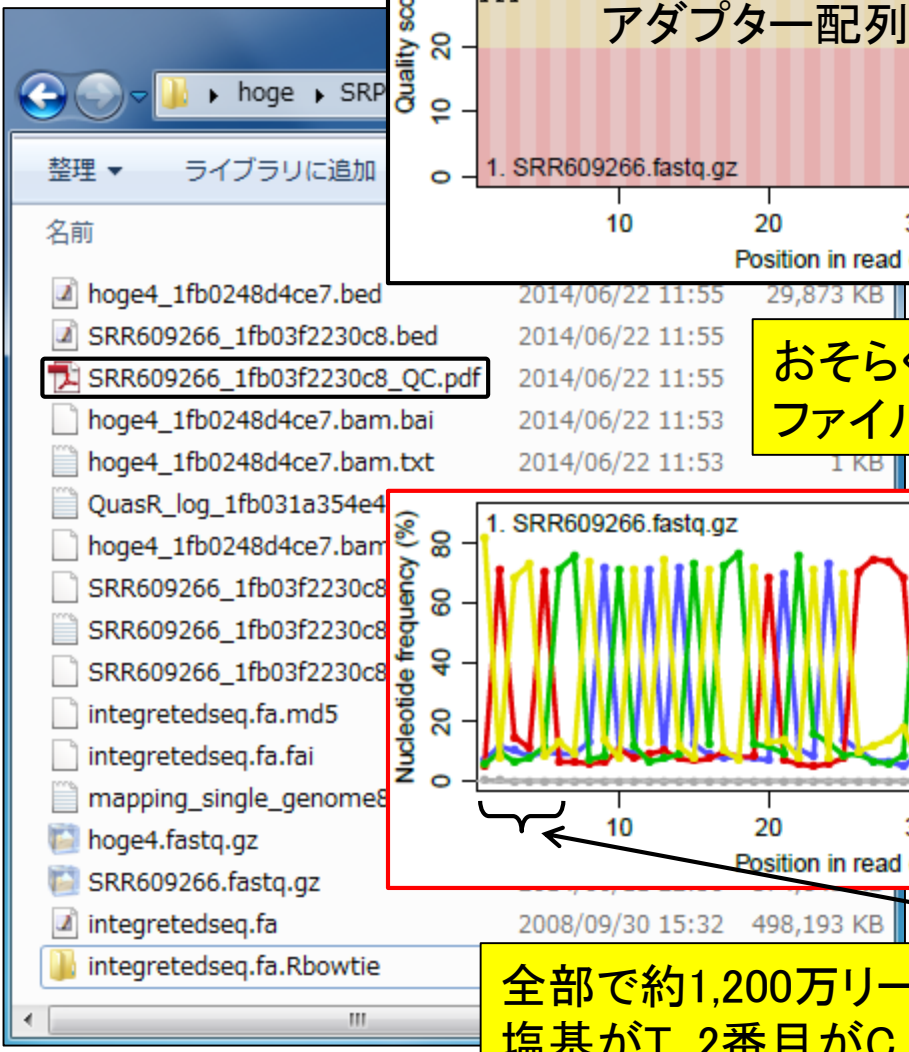
R Console

```
> alignmentStats(out)                          #マッピング結果(alignment statist$
                          seqlength  mapped unmapped
pre_adapter_trim:genome   502962917    2257 11926171
post_adapter_trim:genome 502962917 1308126 10620302
>
> #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 Co
> qQCReport(out, pdfFilename=out_f)            #QCレポート結果をファイルに保存
collecting quality control data
creating QC plots
> out_f                                        #ファイル名を表示してるだけです
[1] "C:/Users/kadota/Desktop/hoge/SRP016842\\SRR609266_1fb03f2230c8_QC.pdf"
[2] "C:/Users/kadota/Desktop/hoge/SRP016842\\hoge4_1fb0248d4ce7_QC.pdf"
>
> #ファイルに保存(BED形式ファイル)
> tmpfname <- out@alignments[,1]               #ファイル名(in_f1の1列目に相当)を$
> for(i in 1:length(tmpfname)){                #サンプル数(ファイル数)分だけルー$
+   hoge <- readGAlignments(tmpfname[i]) #BAM形式ファイルを読み込んだ結果$
+   hoge <- as.data.frame(hoge)                #データフレーム形式に変換
+   tmp <- hoge[, c("seqnames","start","end")]#必要な列の情報のみ抽出した$
+   out_f <- sub(".bam", ".bed", tmpfname[i])#BED形式ファイル名を作成した$
+   out_f                                      #ファイル名を表示してるだけです
+   write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F, col.$
+ }
```

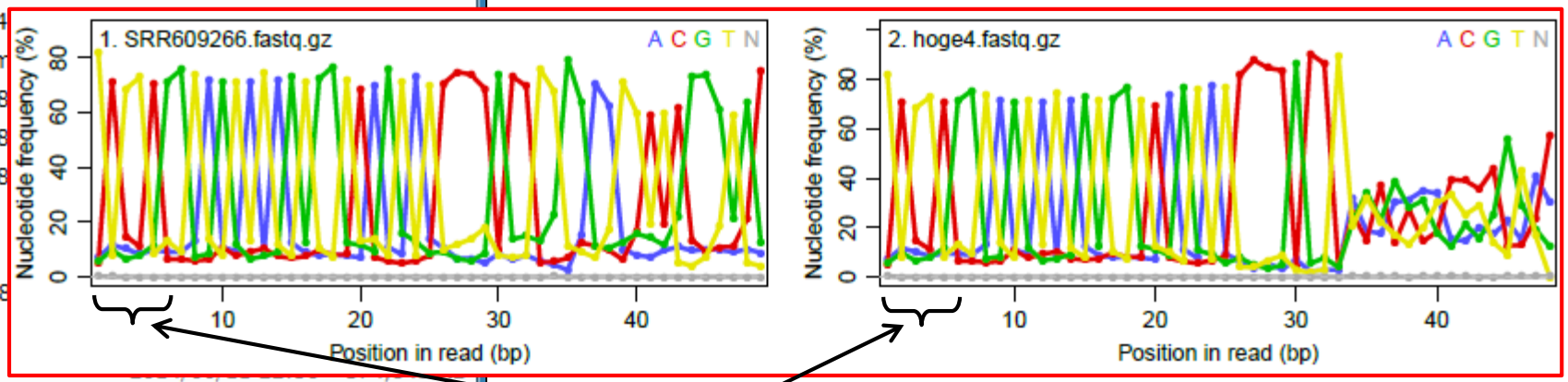
マッピング結果。除去後のマップされたリード数(1,308,126リード)が除去前(2,257リード)に比べて桁違いに多いことが分かる。

small RNA-seqデータのマッピング結果



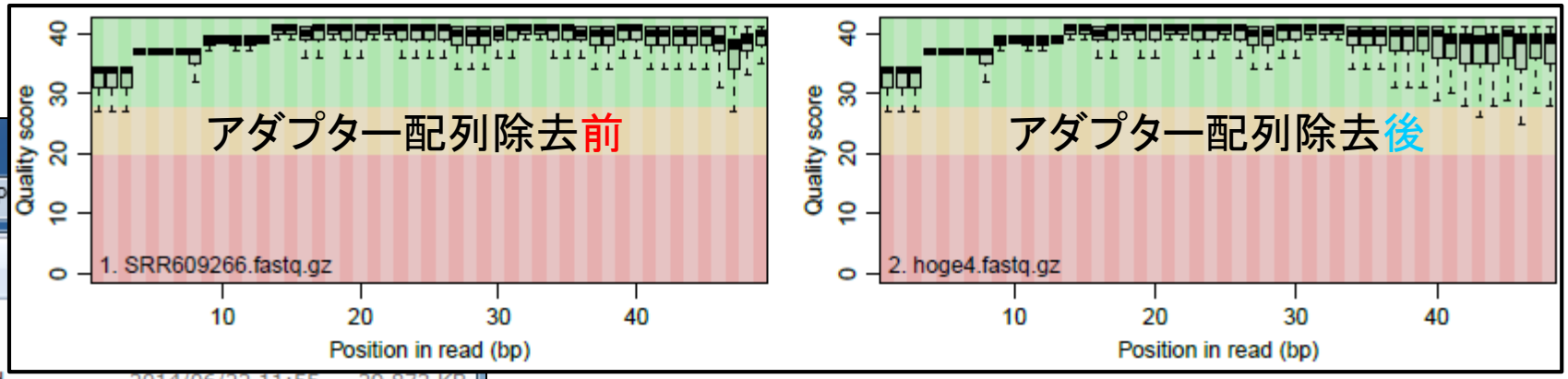


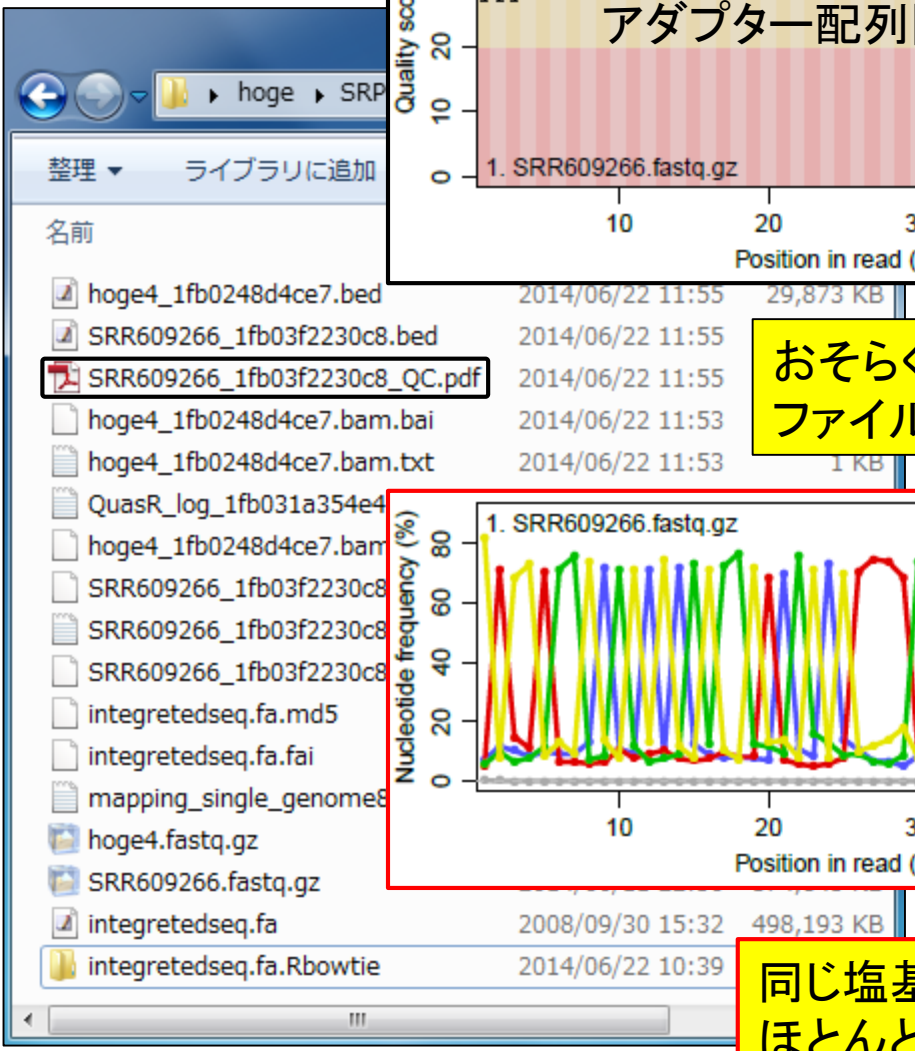
おそらくどのマッピングプログラムもこのようなサマリーレポートファイルを出力する。上：クオリティ分布、下：塩基組成



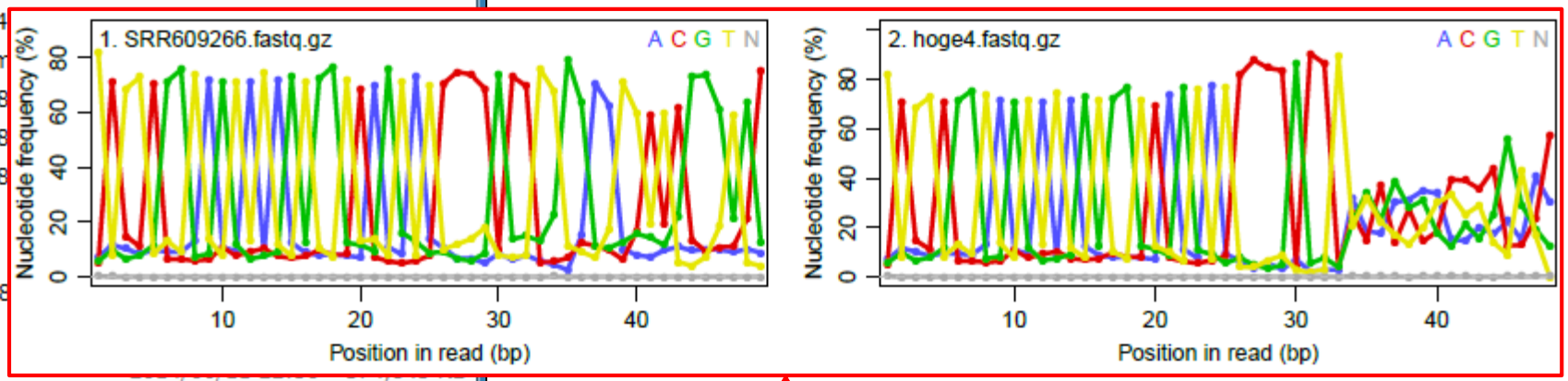
全部で約1,200万リードのポジションごとの塩基組成。「1番目の塩基がT, 2番目がC, 3-4番目がT, …」が多いことを表している。

small RNA-seqデータのマッピング結果

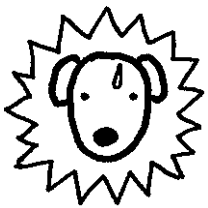


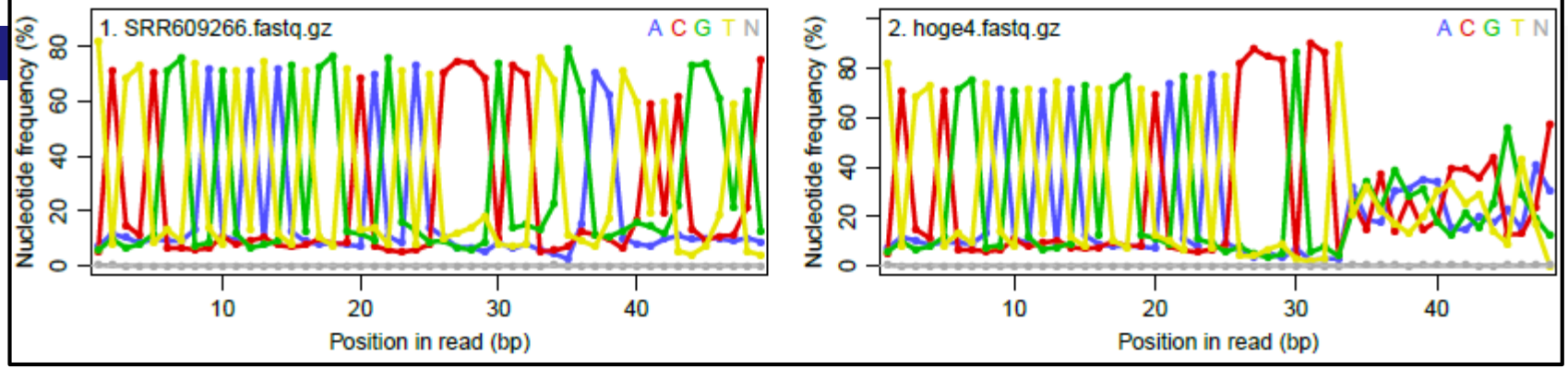


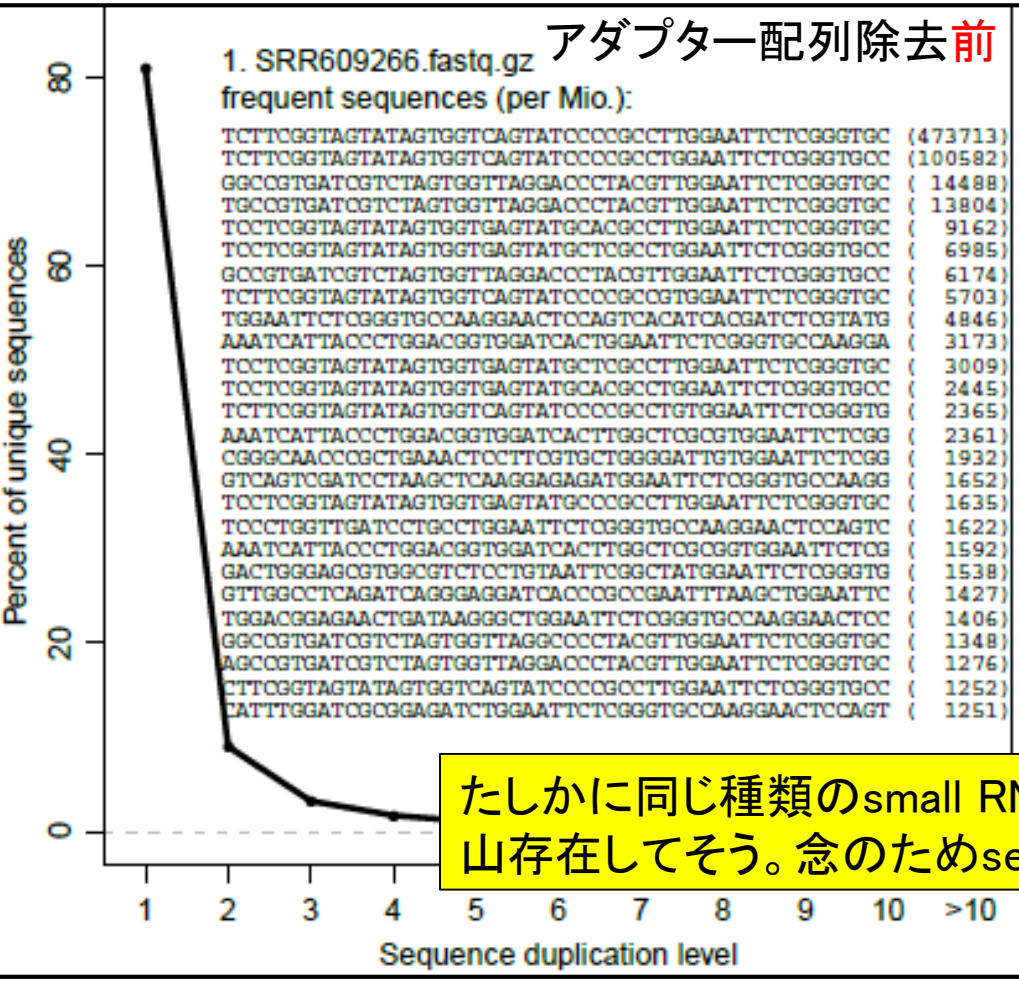
おそらくどのマッピングプログラムもこのようなサマリーレポートファイルを出力する。上：クオリティ分布、下：塩基組成



同じ塩基配列からなるリード(TCTTCGGT…)がほとんどを占めているようにも見える。バグ?!




1. SRR609266.fastq.gz
A C G T N
Nucleotide frequency (%)
0
20
40
60
80
10
20
30
40
Position in read (bp)
2. hoge4.fastq.gz
A C G T N
Nucleotide frequency (%)
0
20
40
60
80
10
20
30
40
Position in read (bp)


アダプター配列除去前
1. SRR609266.fastq.gz
frequent sequences (per Mio.):
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC (473713)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTGGAATTCTCGGGTGCC (100582)
GGCCGTGATCGTCTAGTGGTTAGGACCCTACGTTGGAATTCTCGGGTGC (14488)
TGCCGTGATCGTCTAGTGGTTAGGACCCTACGTTGGAATTCTCGGGTGC (13804)
TCCTCGGTAGTATAGTGGTGAGTATGCACGCCTTGGAATTCTCGGGTGC (9162)
TCCTCGGTAGTATAGTGGTGAGTATGCTCGCCTGGAATTCTCGGGTGCC (6985)
GCCGTGATCGTCTAGTGGTTAGGACCCTACGTTGGAATTCTCGGGTGCC (6174)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCGTGGAATTCTCGGGTGC (5703)
TGGAATTCTCGGGTGCCAAGGAACTCCAGTCACATCACGATCTCGTATG (4846)
AAATCATTACCCTGGACGGTGGATCACTGGAATTCTCGGGTGCCAAGGA (3173)
TCCTCGGTAGTATAGTGGTGAGTATGCTCGCCTTGGAATTCTCGGGTGC (3009)
TCCTCGGTAGTATAGTGGTGAGTATGCACGCCTGGAATTCTCGGGTGCC (2445)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTGTGGAATTCTCGGGTG (2365)
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGTGGAATTCTCGG (2361)
CGGGCAACCCGCTGAAACTCCTTCGTGCTGGGGATTGTGGAATTCTCGG (1932)
GTCAGTCGATCCTAAGCTCAAGGAGAGATGGAATTCTCGGGTGCCAAGG (1652)
TCCTCGGTAGTATAGTGGTGAGTATGCCCGCCTTGGAATTCTCGGGTGC (1635)
TCCCTGGTTGATCCTGCCTGGAATTCTCGGGTGCCAAGGAACTCCAGTC (1622)
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGTGGAATTCTCG (1592)
GACTGGGAGCGTGGCGTCTCCTGTAATTCGGCTATGGAATTCTCGGGTG (1538)
GTTGGCCTCAGATCAGGGAGGATCACCCGCCGAATTTAAGCTGGAATTC (1427)
TGGACGGAGAACTGATAAGGGCTGGAATTCTCGGGTGCCAAGGAACTCC (1406)
GGCCGTGATCGTCTAGTGGTTAGGCCCCTACGTTGGAATTCTCGGGTGC (1348)
AGCCGTGATCGTCTAGTGGTTAGGACCCTACGTTGGAATTCTCGGGTGC (1276)
CTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGCC (1252)
CATTTGGATCGCGGAGATCTGGAATTCTCGGGTGCCAAGGAACTCCAGT (1251)
Percent of unique sequences
0
20
40
60
80
1
2
3
4
5
6
7
8
9
10
>10
Sequence duplication level
アダプター配列除去後
2. hoge4.fastq.gz
frequent sequences (per Mio.):
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCT (477603)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCC (102080)
GGCCGTGATCGTCTAGTGGTTAGGACCCTACGT (14594)
TGCCGTGATCGTCTAGTGGTTAGGACCCTACGT (13912)
TCCTCGGTAGTATAGTGGTGAGTATGCACGCCT (9344)
TCCTCGGTAGTATAGTGGTGAGTATGCTCGCC (6968)
GCCGTGATCGTCTAGTGGTTAGGACCCTACGT (6370)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCG (5864)
TGGAATTCTCGGGTGCCAAGGAACTCCAGTCACATCACGATCTCGTA (4857)
AAATCATTACCCTGGACGGTGGATCAC (3176)
TCCTCGGTAGTATAGTGGTGAGTATGCTCGCCT (3065)
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTG (2475)
TCCTCGGTAGTATAGTGGTGAGTATGCACGCC (2460)
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCG (2354)
CGGGCAACCCGCTGAAACTCCTTCGTGCTGGGGATTG (1975)
TCCTCGGTAGTATAGTGGTGAGTATGCCCGCCT (1745)
TCCCTGGTTGATCCTGCC (1686)
GTCAGTCGATCCTAAGCTCAAGGAGAGA (1670)
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGG (1645)
GACTGGGAGCGTGGCGTCTCCTGTAATTCGGCTA (1553)
GTTGGCCTCAGATCAGGGAGGATCACCCGCCGAATTTAAGC (1470)
TGGACGGAGAACTGATAAGGGC (1451)
CTTCGGTAGTATAGTGGTCAGTATCCCCGCCT (1367)
GGCCGTGATCGTCTAGTGGTTAGGCCCCTACGT (1355)
TGCCGTGATCGTCTAGTGGTTAGGCCCCTACGT (1278)
TGAGATCATTGTGAAAGCTAAT (1274)
CATTTGGATCGCGGAGATC (1268)
CCTACGT (1237)
GAACCACTGATTGCAT (1144)
Percent of unique sequences
0
20
40
60
80
1
2
3
4
5
6
7
8
9
10
>10
Sequence duplication level
たしかに同じ種類のsmall RNA配列(TCTTCGGT…)が沢山存在してそう。念のためsequence logosで確認してみる。

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - **Sequence logosの実行**
    - **実データのgzip圧縮FASTQファイル(実習なし)**
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

- 解析 | 一般 | GC含量 (GC contents)(last modified 2014/05/01)
- 解析 | 一般 | Sequence logos(Schneider_1990) modified 2014/06/21) NEW
- 解析 | 一般 | 上流配列解析 | LDSS(Yamamoto 07)(last modified 2012/07/17)
- 解析 | 一般 | 上流配列解析 | Relative Appearance Ratio(Yamamoto_2011)(last modified

• 解析 | 一般 | Sequence logos(Schneider_1990)

解析 | 一般 | Sequence logos(Schneider_1990) NEW

seqLogoパッケージを用いてsequence logos (Schneider and Stephens, 1990)を実行するやり方を示します。ここでは

mu
があ
「ファイ

8. FASTQ形式ファイル(SRR609266.fastq.gz)の場合:

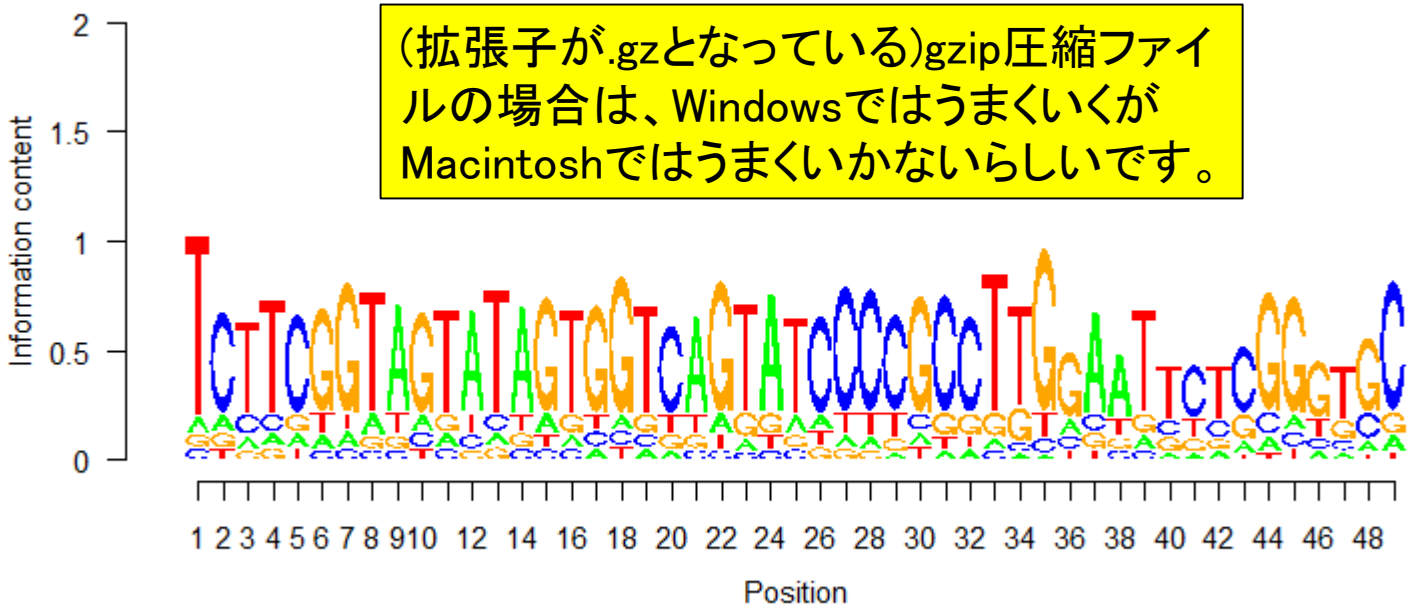
small RNA-seqデータ(400Mb弱、11,928,428リード)です。圧縮ファイルもreadDNAStringSet関数で通常手順で読み込めます。原著論文(Nie et al., BMC Genomics, 2013)中の記述からGSE41841を頼りに、SRP016842にたどりつき、イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたものが入力ファイルです。

1. 入力

```
in_f <- "SRR609266.fastq.gz"          #入力ファイル名を
out_f <- "hoge8.png"                  #出力ファイル名を
param_fig <- c(800, 370)              #ファイル出力時の

#必要なパッケージをロード
library(B
library(s

#入力ファ
fasta <-

#本番(seq
hoge <- c
out <- ma

#ファイル
png(out_f
seqLogo(c
dev.off()
```

small RNA-seqファイルをそのまま入力としてSequence logosを実行することもできる。**実習ではやりません。**

(拡張子が.gzとなっている)gzip圧縮ファイルの場合は、Windowsではうまくいくが Macintoshではうまくいかないらしいです。



- 解析 | 一般 | GC含量 (GC contents)(last modified 2014/05/01)
- 解析 | 一般 | Sequence logos(Schneider_1990)(last modified 2014/06/21) NEW
- 解析 | 一般 | 上流配列解析 | LDSS(Yamamoto ...07)(last modified 2012/07/17)
- 解析 | 一般 | 上流配列解析 | Relative Appearance Ratio(Yamamoto_2011)(last modified ...

• 解析 | 一般 | Sequence logos(Schneider_1990)

解析 | 一般 | Sequence logos(Schneider_1990) NEW

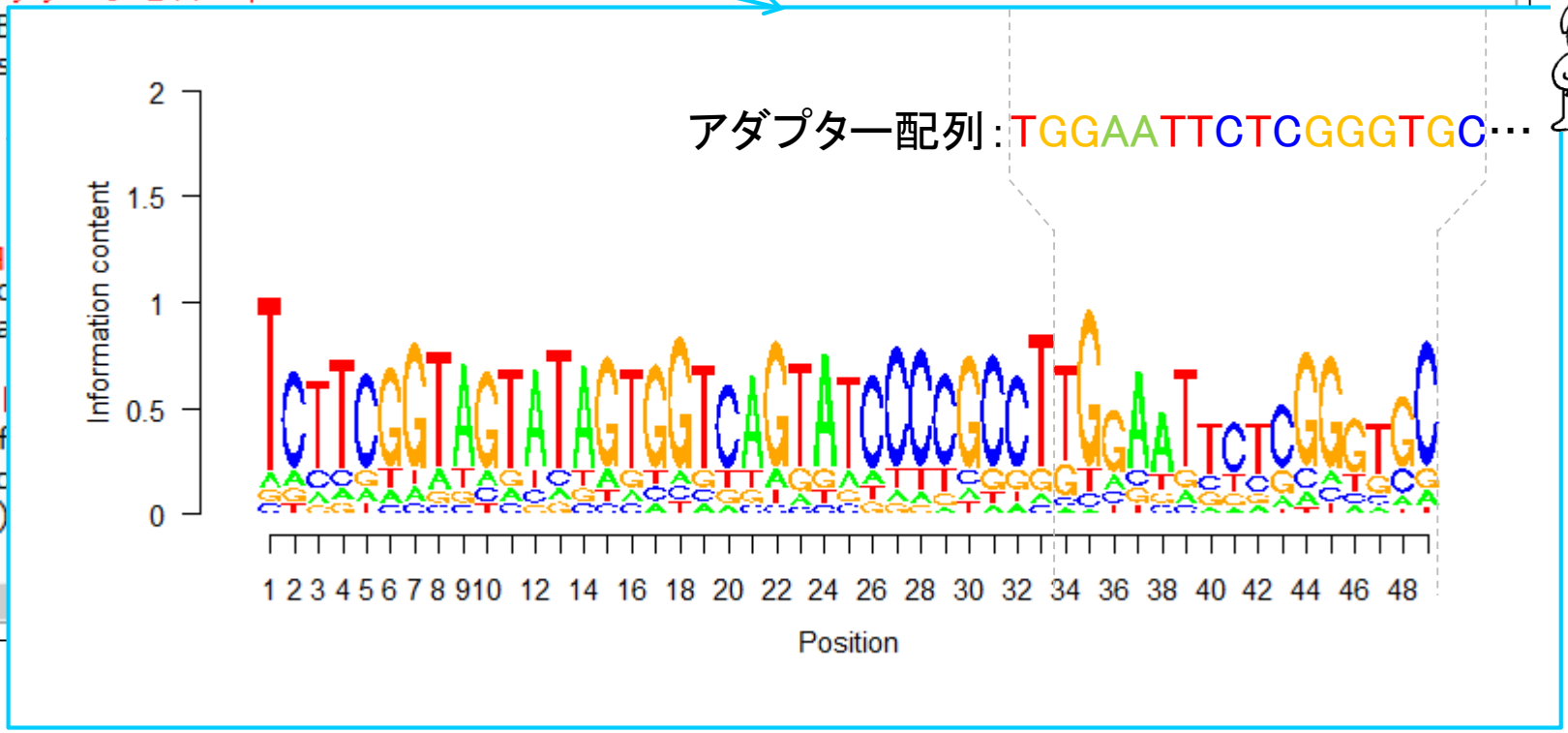
seqLogoパッケージを用いてsequence logos (Schneider and Stephens, 1990)を実行するやり方を示します。ここでは

8. FASTQ形式ファイル(SRR609266.fastq.gz)の場合:

small RNA-seqデータ(400Mb弱、11,928,428リード)です。圧縮ファイルもreadDNAStringSet関数で通常手順で読み込めます。原著論文(Nie et al., BMC Genomics, 2013)中の記述からGSE41841を頼りに、SRP016842にたどりつき、イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu_2013)の7を実行して得られたものが入力ファイルです。

```
in_f <- "SRR609266.fastq.gz"          #入力ファ
out_f <- "hoge8.png"                  #出力ファ
param_fig <- c(800, 370)              #ファイル

#必要なパッケージをロード
library(B
library(s

#入力ファ
fasta <-

#本番(seq
hoge <- c
out <- ma

#ファイル
png(out_f
seqLogo(c
dev.off()
```

アダプター配列除去**前**の実行結果。アダプター配列に相当する部分のロゴがよくわかる。



- 解析 | 一般 | GC含量 (GC contents)(last modified 2014/05/01)
- 解析 | 一般 | Sequence logos(Schneider_1990)(last modified 2014/06/21) NEW
- 解析 | 一般 | 上流配列解析 | LDSS(Yamamoto ...07)(last modified 2012/07/17)

• 解析 | 一般 | Sequence logos(Schneider_1990)

解析 | 一般 | Sequence logos(Schneider_1990) NEW

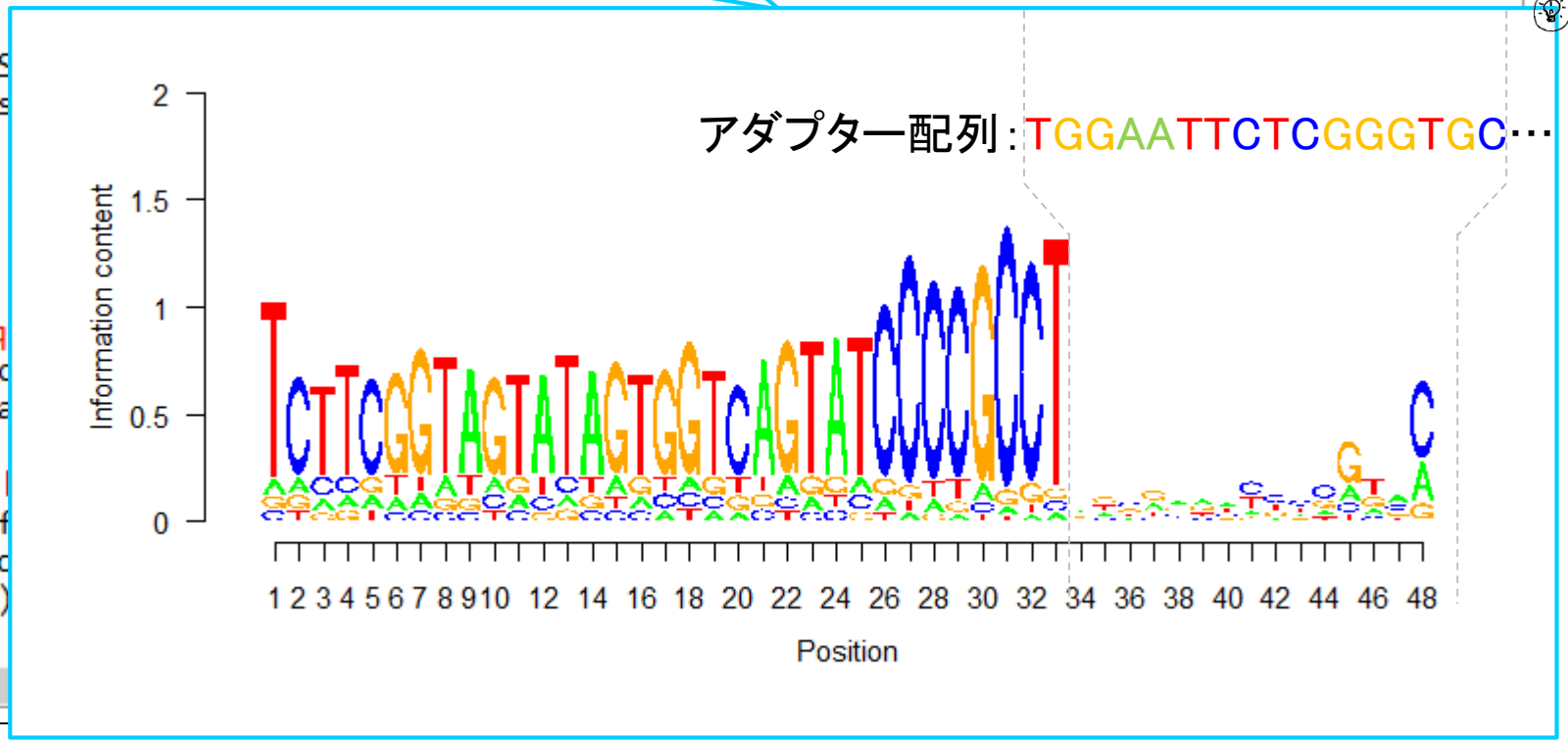
seqLogoパッケージを用いてsequence logos (Schneider and Stephens, 1990)を実行するやり方を示します。ここでは

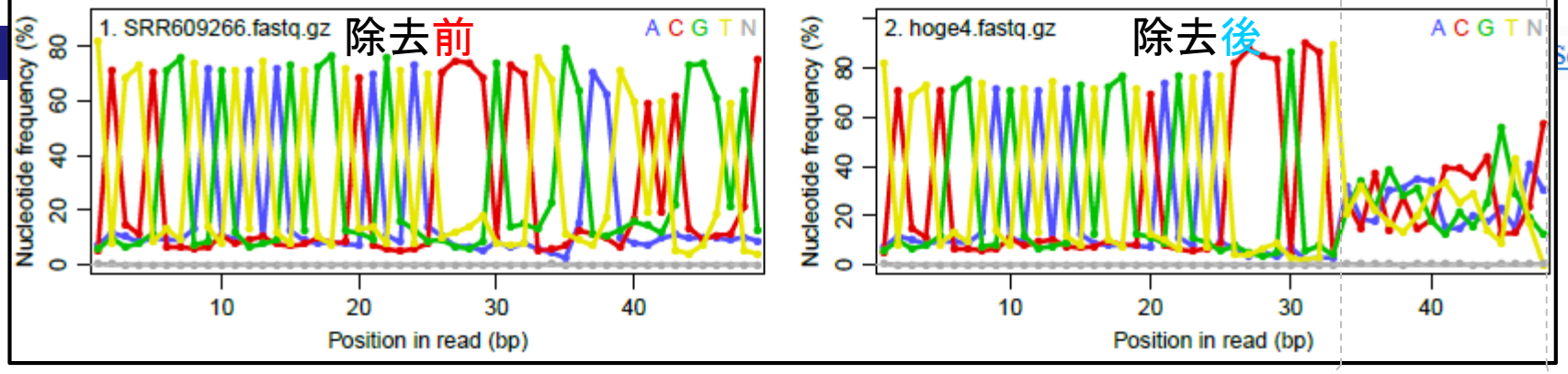
9. FASTQ形式ファイル(hoge4.fastq.gz)の場合:

small RNA-seqデータ(280Mb弱、11,928,428リード)です。原著論文(Nie et al., BMC Genomics, 2013)中の記述からGSE41841を頼りに、SRP016842にたどりつき、前処理 | トリミング | アダプター配列除去(応用) | ShortRead (Morgan 2009)の4を実行して得られたものが入力ファイルです。アダプター配列除去後のデータなので、リードごとに配列長が異なる場合でも読み込めるShortReadパッケージ中の...関数を用いています。

```
in_f <- "hoge4.fastq.gz"                    #入力ファ
out_f <- "hoge9.png"                        #出力ファ
param_fig <- c(787, 370)                    #ファイル

#必要なパ
library(S
library(s

#入力ファ
fastq <-
fasta <-

#本番(seq
hoge <- c
out <- ma

#ファイル
png(out_f
seqLogo(c
dev.off()
```

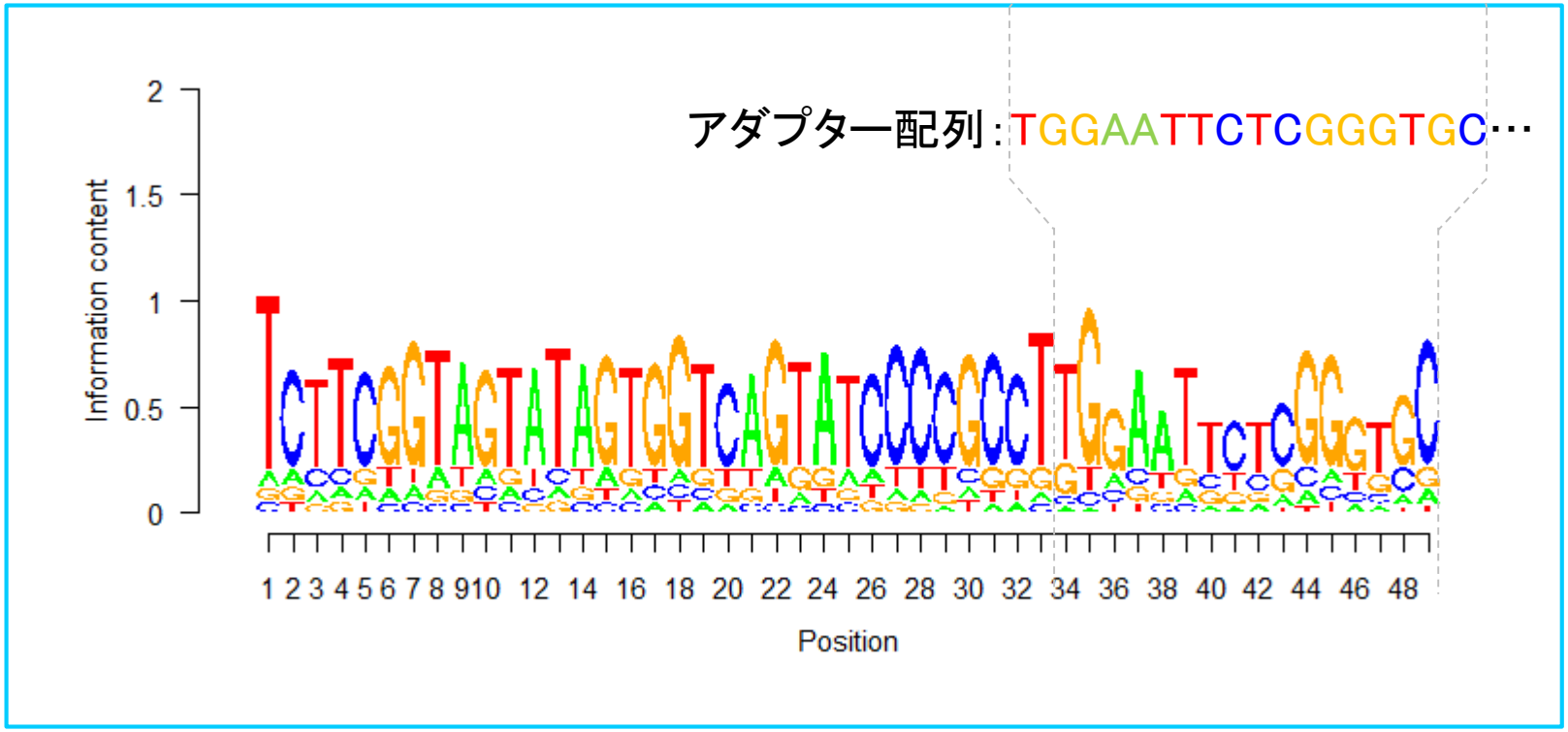
アダプター配列除去後の実行結果。アダプター配列に相当する部分のロゴが消えていることがわかる。実習ではやりません。





Schneider 1990)

正しくアダプター配列を除去できていることもわかる



Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - **軽量版非圧縮FASTQファイル(実習あり)**
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

軽量版FASTQファイル作成

(Rで)塩基配列解析

- 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 2014/07/17) NEW
- 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 2013/06/18)
- 前処理 | フィルタリング | 任意のIDを含む配列を抽出 (last modified 2013/06/18)

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 NEW

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"            #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                  #出力ファイル名を指定してout_fに格納
param <- 100000                         #ランダム抽出したいリード数を指定

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

#入力ファイルの読み込み
fastq <- readFastq(in_f)                #in_fで指定したファイルの読み込み
id(fastq)                               #確認してるだけです(description情報を表示

#本番
set.seed(1010)                          #おまじない(同じ乱数になるようにするため)
obj <- sample(1:length(fastq), param, replace=F)#リード数の数値の中からparamで指
fastq <- fastq[sort(obj)]               #objで指定したリードのみソートして抽出した
id(fastq)                               #確認してるだけです(description情報を表示

#ファイルに保存
writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファイル名で保存
```

実データの100分の1程度のリード数からなるファイルを作成して、マッピングやアセンブルなど目的の解析を仮実行。計算時間の見積もりや動作確認を行う際に利用。

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

• 前処理 | フィルタリング | 任意のリード(サブセット)を抽出 参考

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                #出力ファイル名を指定してout_fに格納
param <- 100000                       #ランダム抽出したいリード数を指定

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

#入力ファイルの読み込み
fastq <- readFastq(in_f)              #in_fで指定した
id(fastq)                             #確認してるだけ

#本番
set.seed(1010)                        #おまじない(同じ
obj <- sample(1:length(fastq), param, replace=F)#リー
fastq <- fastq[sort(obj)]             #objで指定した
id(fastq)                             #確認してるだけ

#ファイルに保存
writeFastq(fastq, out_f, compress=F)  #fastqの中身を指
```

コピペ後のR Console画面。エラーなく実行できており、指定した100000リード分のサブセットになっていることが読み取れる。

R Console

```
> fastq <- fastq[sort(obj)]                    #objで指定したリードのみソ$
> id(fastq)                                    #確認してるだけです(descrip$
  A BStringSet instance of length 100000
         width seq
     [1]    43 SRR609266.17 FCC0N94ACXX:4:1101:1445:2316/1
     [2]    43 SRR609266.93 FCC0N94ACXX:4:1101:8074:2365/1
     [3]    45 SRR609266.179 FCC0N94ACXX:4:1101:14519:2404/1
     [4]    44 SRR609266.310 FCC0N94ACXX:4:1101:4849:2685/1
     [5]    45 SRR609266.462 FCC0N94ACXX:4:1101:17605:2539/1
     ...   ... ...
 [99996]    51 SRR609266.11927688 FCC0N94ACXX:4:2316:4515:100592/1
 [99997]    51 SRR609266.11927705 FCC0N94ACXX:4:2316:5393:100578/1
 [99998]    52 SRR609266.11927883 FCC0N94ACXX:4:2316:14689:100559/1
 [99999]    51 SRR609266.11928052 FCC0N94ACXX:4:2316:2897:100869/1
[100000]    51 SRR609266.11928053 FCC0N94ACXX:4:2316:2843:100907/1
>
> #ファイルに保存
> writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファ$
>
```

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

- 前処理 | フィルタリング | 任意のリード(サブセット)を抽出

参考

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                #出力ファイル名を指定してout_fに格納
param <- 100000                       #ランダム抽出したいリード数を指定
```

作業ディレクトリ中に指定した出力ファイル名のものが生成されているはずです。

```
@SRR609266.17 FCCON94ACXX:4:1101:1445:2316/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiihiiiiiiihihifhiiiiiiiihiiiihfgagfh
@SRR609266.93 FCCON94ACXX:4:1101:8074:2365/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiiiiiiiiiiiiiiiiiiiiiighiiiiiiigaeg
@SRR609266.179 FCCON94ACXX:4:1101:14519:2404/1
TATTGTGGACAACTCGGAGTTTTGGAATTCTCGGGTGCCAAGGAACTCC
+
bbbeeeeeggfggiiiiiiiiiiiihSbffhiiiieffhiihhghiiii
@SRR609266.310 FCCON94ACXX:4:1101:4849:2685/1
CTCGCGGGTCGATGAAGAACTGGACTTCTCGGGTGCCAAGGAACTCCAG
+
bbbeeeeeggggfhiifhihiiifQae_efdgi¥`dd_fd`bgdeeeee
@SRR609266.462 FCCON94ACXX:4:1101:17605:2539/1
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGGTGGAATTCTC
+
bbbeeeeeggfgiiiiiifhiihihiiifhhiiihhih`geeebdddd
@SRR609266.503 FCCON94ACXX:4:1101:20897:2620/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
_b_eeeecggggfihhhhiiihhiiiiihfgihiiichiiiiigfhaag
@SRR609266.576 FCCON94ACXX:4:1101:6000:2828/1
TCCCTGGTTGATCCTGCCTGGAATTCTCGGGTGCCAAGGAACTCCAGTC
+
bbbeeeeeggggiiiiiiiiihighhiiiiieghiiiihhhiiiiiihi
@SRR609266.582 FCCON94ACXX:4:1101:6721:2797/1
ATCCTGACGAAAGAATCTGGAATTCTCGGGTGCCAAGGAACTCCAGTCA
+
bbbeeeeeggggiiiiiiiiiihihiiiiifghiiihhghiiiiifbgh
```

R Console

```
> fastq <- fastq[sort(obj)]                #objで指定したリードのみソ$
> id(fastq)                                #確認してるだけです(descrip$
  A BStringSet instance of length 100000
         width seq
     [1]    43 SRR609266.17 FCCON94ACXX:4:1101:1445:2316/1
     [2]    43 SRR609266.93 FCCON94ACXX:4:1101:8074:2365/1
     [3]    45 SRR609266.179 FCCON94ACXX:4:1101:14519:2404/1
     [4]    44 SRR609266.310 FCCON94ACXX:4:1101:4849:2685/1
     [5]    45 SRR609266.462 FCCON94ACXX:4:1101:17605:2539/1
     ...   ... ...
 [99996]    51 SRR609266.11927688 FCCON94ACXX:4:2316:4515:100592/1
 [99997]    51 SRR609266.11927705 FCCON94ACXX:4:2316:5393:100578/1
 [99998]    52 SRR609266.11927883 FCCON94ACXX:4:2316:14689:100559/1
 [99999]    51 SRR609266.11928052 FCCON94ACXX:4:2316:2897:100869/1
[100000]    51 SRR609266.11928053 FCCON94ACXX:4:2316:2843:100907/1
>
> #ファイルに保存
> writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファ$
>
```

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

- 前処理 | フィルタリング | 任意のリード(サブセット)を抽出

参考

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                #出力ファイル名を指定してout_fに格納
param <- 100000                       #ランダム抽出したいリード数を指定
```

```
@SRR609266.17 FCC0N94ACXX:4:1101:1445:2316/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiihiiiiiiihihifhiiiiiiiihiiiihfgagfh
@SRR609266.93 FCC0N94ACXX:4:1101:8074:2365/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiiiiiiiiiiiiiiiiiiiiiighiiiiiiigaeg
@SRR609266.179 FCC0N94ACXX:4:1101:14519:2404/1
TATTGTGGACAACTCGGAGTTTTGGAATTCTCGGGTGCCAAGGAACTCC
+
bbbeeeeeggfggiiiiiiiiiiiihSbffhiiiieffhiihhghiiii
@SRR609266.310 FCC0N94ACXX:4:1101:4849:2685/1
CTCGCGGGTCGATGAAGAACTGGACTTCTCGGGTGCCAAGGAACTCCAG
+
bbbeeeeeggggfhiifhihiiifQae_efdgi¥`dd_fd`bgdeeeee
@SRR609266.462 FCC0N94ACXX:4:1101:17605:2539/1
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGGTGGAATTCTC
+
bbbeeeeeggfgiiiiiifhiihihiiifhhiiihhih`geeebdddd
@SRR609266.503 FCC0N94ACXX:4:1101:20897:2620/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
_b_eeeecggggfihhhhiiihhiiiiihfgihiiichiiiiigfhaag
@SRR609266.576 FCC0N94ACXX:4:1101:6000:2828/1
TCCCTGGTTGATCCTGCCTGGAATTCTCGGGTGCCAAGGAACTCCAGTC
+
bbbeeeeeggggiiiiiiiiihighhiiiiieghiiiihhhiiiiiihi
@SRR609266.582 FCC0N94ACXX:4:1101:6721:2797/1
ATCCTGACGAAAGAATCTGGAATTCTCGGGTGCCAAGGAACTCCAGTCA
+
bbbeeeeeggggiiiiiiiiihihiiiiifghiiihhghiiiiifbgh
```

R Console画面上に表示されているものは、description部分に相当するものです。id という関数を利用しています。

R Console

```
> fastq <- fastq[sort(obj)]                #objで指定したリードのみソ$
> id(fastq)                                #確認してるだけです(descrip$
  A BStringSet instance of length 100000
         width seq
     [1]    43 SRR609266.17 FCC0N94ACXX:4:1101:1445:2316/1
     [2]    43 SRR609266.93 FCC0N94ACXX:4:1101:8074:2365/1
     [3]    45 SRR609266.179 FCC0N94ACXX:4:1101:14519:2404/1
     [4]    44 SRR609266.310 FCC0N94ACXX:4:1101:4849:2685/1
     [5]    45 SRR609266.462 FCC0N94ACXX:4:1101:17605:2539/1
     ...   ... ...
 [99996]    51 SRR609266.11927688 FCC0N94ACXX:4:2316:4515:100592/1
 [99997]    51 SRR609266.11927705 FCC0N94ACXX:4:2316:5393:100578/1
 [99998]    52 SRR609266.11927883 FCC0N94ACXX:4:2316:14689:100559/1
 [99999]    51 SRR609266.11928052 FCC0N94ACXX:4:2316:2897:100869/1
[100000]    51 SRR609266.11928053 FCC0N94ACXX:4:2316:2843:100907/1
>
> #ファイルに保存
> writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファ$
>
```

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

• 前処理 | フィルタリング | 任意のリード(サブセット)を抽出 参考

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                #出力ファイル名を指定してout_fに格納
param <- 100000                       #ランダム抽出したいリード数を指定
```

```
@SRR609266.17 FCC0N94ACXX:4:1101:1445:2316/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiihiiiiiiihihifhiiiiiiiihiiiihfgagfh
@SRR609266.93 FCC0N94ACXX:4:1101:8074:2365/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
bbbeeeeeggggiiiiiiiiiiiiiiiiiiiiiiighiiiiiiigaeg
@SRR609266.179 FCC0N94ACXX:4:1101:14519:2404/1
TATTGTGGACAACTCGGAGTTTTGGAATTCTCGGGTGCCAAGGAACTCC
+
bbbeeeeeggfggiiiiiiiiiiiiihSbffhiiiieffhiihhghiiii
@SRR609266.310 FCC0N94ACXX:4:1101:4849:2685/1
CTCGCGGGTCGATGAAGAACTGGACTTCTCGGGTGCCAAGGAACTCCAG
+
bbbeeeeeggggfhiifhihiiifQae_efdgi¥`dd_fd`bgdeeeee
@SRR609266.462 FCC0N94ACXX:4:1101:17605:2539/1
AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGGTGGAATTCTC
+
bbbeeeeeggfgiiiiiifhiihihiiifhhiiihhih`geeebdddd
@SRR609266.503 FCC0N94ACXX:4:1101:20897:2620/1
TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
+
_b_eeeecggggfihhhhiiihhiiiiihfgihiiichiiiiigfhaag
@SRR609266.576 FCC0N94ACXX:4:1101:6000:2828/1
TCCCTGGTTGATCCTGCCTGGAATTCTCGGGTGCCAAGGAACTCCAGTC
+
bbbeeeeeggggiiiiiiiihighhiiiiieghiiiihhhiiiiiihi
@SRR609266.582 FCC0N94ACXX:4:1101:6721:2797/1
ATCCTGACGAAAGAATCTGGAATTCTCGGGTGCCAAGGAACTCCAGTCA
+
bbbeeeeeggggiiiiiiiiihihiiiiifghiiihhghiiiiifbgh
```

sread関数を利用してリード塩基配列情報の最初と最後の5リード分が表示されています。49塩基長で揃っていることもわかります。

R Console

```
 [99999]    51 SRR609266.11928052 FCC0N94ACXX:4:2316:2897:100869/1
[100000]    51 SRR609266.11928053 FCC0N94ACXX:4:2316:2843:100907/1
>
> #ファイルに保存
> writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファ$
> sread(fastq)
  A DNAStringSet instance of length 100000
        width seq
    [1]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
    [2]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
    [3]    49 TATTGTGGACAACTCGGAGTTTTGGAATTCTCGGGTGCCAAGGAACTCC
    [4]    49 CTCGCGGGTCGATGAAGAACTGGACTTCTCGGGTGCCAAGGAACTCCAG
    [5]    49 AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGGTGGAATTCTC
    ...   ... ...
[99996]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
[99997]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
[99998]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
[99999]    49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTGGAATTCTCGGGTGCC
[100000]   49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
>
```

8. small RNA-seqのgzip圧縮FASTQ形式ファイル(SRR609266.fastq.gz)を読み込んでFASTQ形式で保存する場合:

• 前処理 | フィルタリング | 任意のリード(サブセット)を抽出 参考

イントロ | NGS | 配列取得 | FASTQ or SRALite | SRAdb(Zhu 2013)の7を実行して得られたカイコsmall RNA-seqデータ(Nie et al., BMC Genomics, 2013)です。入力ファイルサイズは400Mb弱、11,928,428リードです。この中から100000リード分をランダムに非復元抽出した結果をgzip圧縮なしで出力しています。出力ファイルはSRR609266_sub.fastqと同じもの(100000リード; 約16MB)になります。Macintoshではうまくいかないかもしれません。

```
in_f <- "SRR609266.fastq.gz"              #入力ファイル名を指定してin_fに格納
out_f <- "hoge8.fastq"                    #出力ファイル名を指定してout_fに格納
param <- 100000                           #ランダム抽出したいリード数を指定

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

#入力ファイルの読み込み
fastq <- readFastq(in_f)                  #in_fで指定した
id(fastq)                                 #確認してるだけ

#本番
set.seed(1010)                            #おまじない(同じ
obj <- sample(1:length(fastq), param, replace=F)#リー
fastq <- fastq[sort(obj)]                 #objで指定した
id(fastq)                                 #確認してるだけ

#ファイルに保存
writeFastq(fastq, out_f, compress=F)      #fastqの中身を指
```

約1,200万リードで見られた「同じ種類のsmall RNA配列(TCTTCGGT…)が沢山存在してそう」な傾向は、10万リードの場合でも見受けられます。

R Console

```
 [99999]     51 SRR609266.11928052 FCC0N94ACXX:4:2316:2897:100869/1
[100000]     51 SRR609266.11928053 FCC0N94ACXX:4:2316:2843:100907/1
>
> #ファイルに保存
> writeFastq(fastq, out_f, compress=F)    #fastqの中身を指定したファ$
> sread(fastq)
  A DNAStringSet instance of length 100000
         width seq
     [1]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
     [2]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
     [3]     49 TATTGTGGACAACTCGGAGTTTTGGAATTCTCGGGTGCCAAGGAACTCC
     [4]     49 CTCGCGGGTCGATGAAGAACTGGACTTCTCGGGTGCCAAGGAACTCCAG
     [5]     49 AAATCATTACCCTGGACGGTGGATCACTTGGCTCGCGGGTGGAATTCTC
     ...    ... ...
 [99996]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
 [99997]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
 [99998]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
 [99999]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTGGAATTCTCGGGTGCC
[100000]     49 TCTTCGGTAGTATAGTGGTCAGTATCCCCGCCTTGGAATTCTCGGGTGC
>
```

- 解析 | 一般 | GC含量 (GC contents)(last modified 2014/05/01)
- 解析 | 一般 | Sequence logos(Schneider_1990)(last modified 2014/06/21) NEW
- 解析 | 一般 | 上流配列解析 | LDSS(Yamamoto ...07)(last modified 2012/07/17)
- 解析 | 一般 | 上流配列解析 | Relative Appearance Ratio(Yamamoto, 2011)(last modified ...

解析 | 一般 | Sequence logos(Schneider_1990) NEW

seqLogoパッケージを用いてsequence logos (Schneider and Stephens, 1990)を実行するやり方を示します。ここで...
multi FASTAファイルを読み込んでポジションごとの出現頻度を調べる目的で利用します。上流-35 bpにTATA b...
が...
「フ...

10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsma...
1. 入力...
データ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)                    #パッケージの読み込み
library(seqLogo)                       #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="fastq")#in_fで指定したファイルの読みi

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])             #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, width=param_fig[1], height=param_fig[2])#出力ファイ
seqLogo(out)                           #塩基組成やicの情報を含むoutを入力とし
dev.off()                              #おまじない
```

WindowsのヒトもMacintoshのヒトもうまくいくはずです。作業ディレクトリ中に入力ファイル(SRR609266_sub.fastq)が存在することを確認した上で、コピペで実行してみましょう。

目的:短時間で実行できるように、ファイルサイズの小さい非圧縮版のファイルを用いてsequence logosをコピペで実行

基本はコピペ

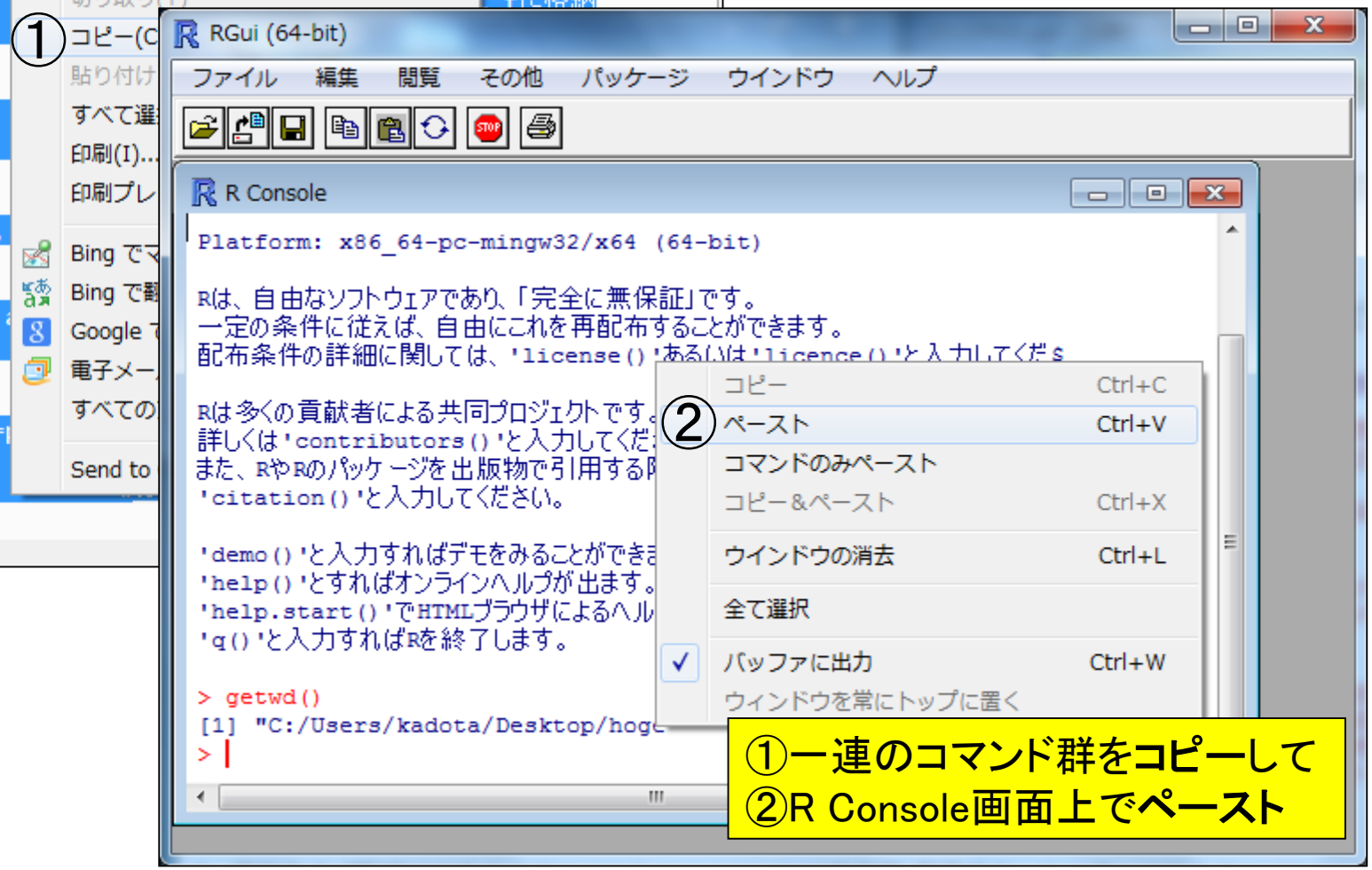
WindowsのヒトはCTRLとALTキーを押しながらコードの枠内で左クリックすると全選択できます

10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsmall RNA-seqデータ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"
out_f <- "hoge10.png"
param_fig <- c(800, 370)

#必要なパッケージをロード
library(Biostrings)
library(seqLogo)

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_f,

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta,
out <- makePWM(hoge[1:4,])

#ファイルに保存
png(out_f, pointsize=13, width=
seqLogo(out)
dev.off()
```

①



②

①一連のコマンド群をコピーして
②R Console画面上でペースト

基本はコピペ

WindowsのヒトもMacintoshのヒトも、うまくいくと以下のようなエラーメッセージのないR Console画面になっているはずです。

10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsmall RNA-seqデータ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"                    #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                            #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)                         #画像サイズを指定(単位はピクセル)

#必要なパッケージをロード
library(Biostrings)
library(seqLogo)

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_f,

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta,
out <- makePWM(hoge[1:4,])

#ファイルに保存
png(out_f, pointsize=13, width=
seqLogo(out)
dev.off()
```

①

切り取り(T)
コピー(C)
貼り付け
すべて選択(A)
印刷(I)...
印刷プレビュー(N)...
Bing でマップ
Bing で翻訳
Google で検索
電子メール (Windows Li
すべてのアクセラレータ
Send to OneNote

R Console

```
 要求されたパッケージ IRanges をロード中です
 要求されたパッケージ XVector をロード中です
> library(seqLogo)                                #パッケージの$
 要求されたパッケージ grid をロード中です
>
> #入力ファイルの読み込み
> fasta <- readDNAStringSet(in_f, format="fastq")#in_f$
>
> #本番(sequence logoを実行)
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T$
> out <- makePWM(hoge[1:4,])                      #hogeはACGT以$
>
> #ファイルに保存
> png(out_f, pointsize=13, width=param_fig[1], height=$
> seqLogo(out)                                    #塩基組成やic$
> dev.off()                                       #おまじない
null device
          1
>
```

実行結果

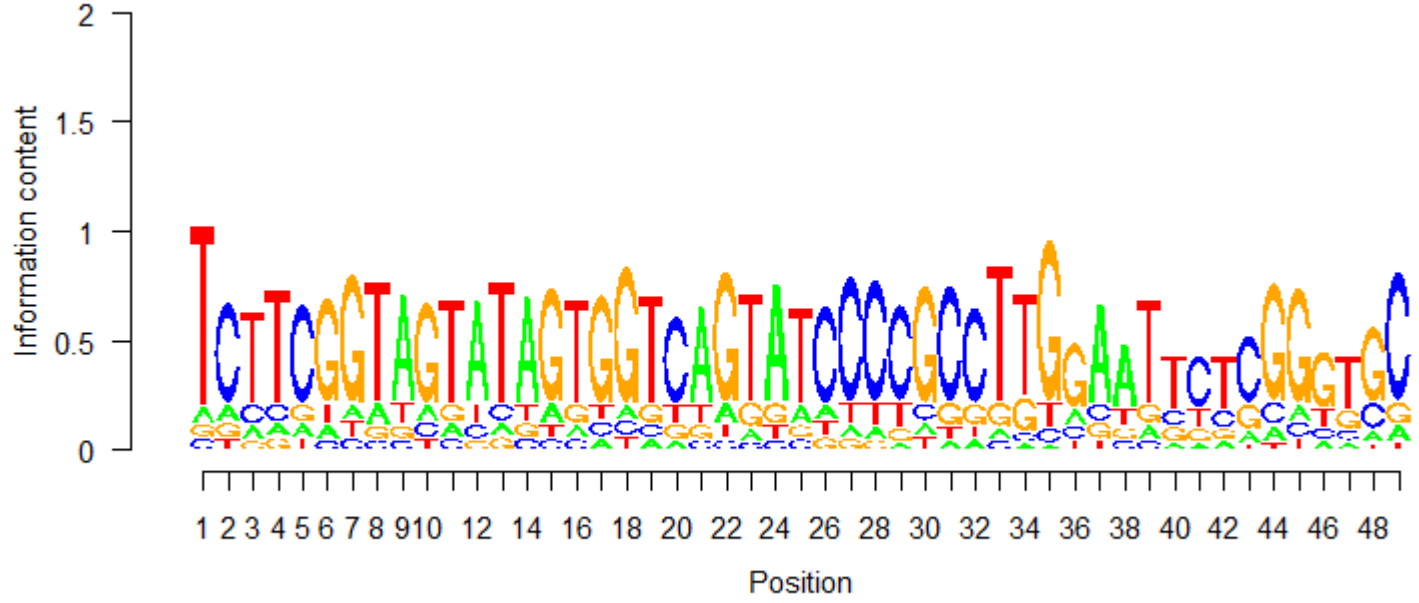
サイズが800×370ピクセルからなるPNG形式ファイル(hoge10.png)が生成される。

10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られた small RNA-seq データ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位i

#必要なパッケージをロード
library(Biostr
library(seqLog

#入力ファイルの
fasta <- readD

#本番(sequence
hoge <- conser
out <- makePWM

#ファイルに保存
png(out_f, poi
seqLogo(out)
dev.off()
```

hoge10.png (100,000リードの結果)



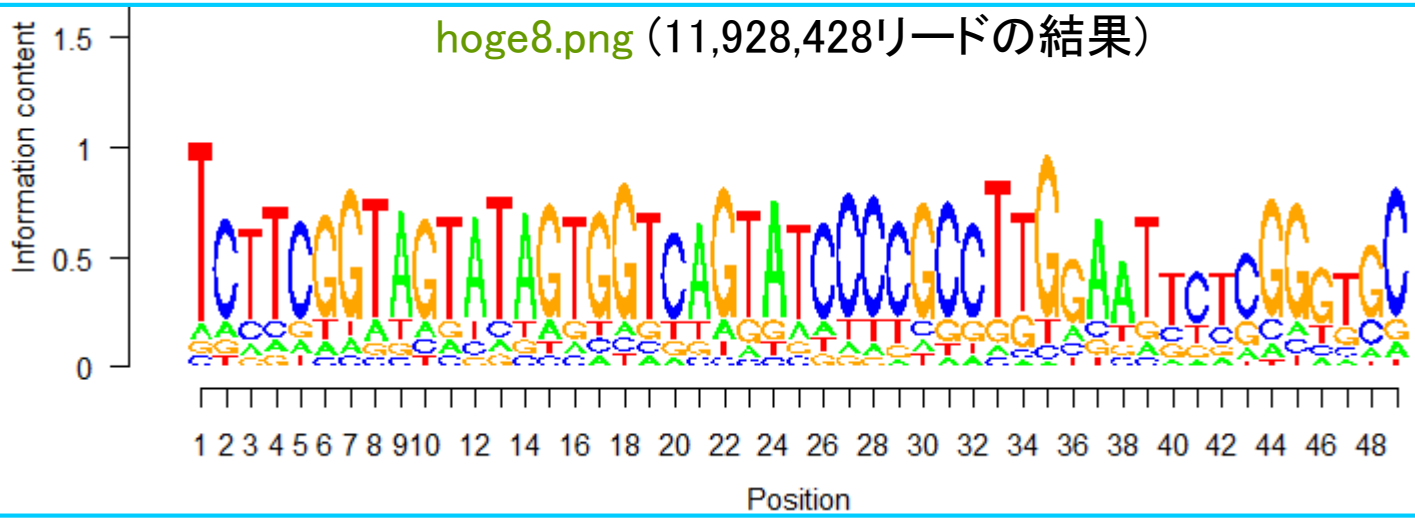
370ピクセル

800ピクセル

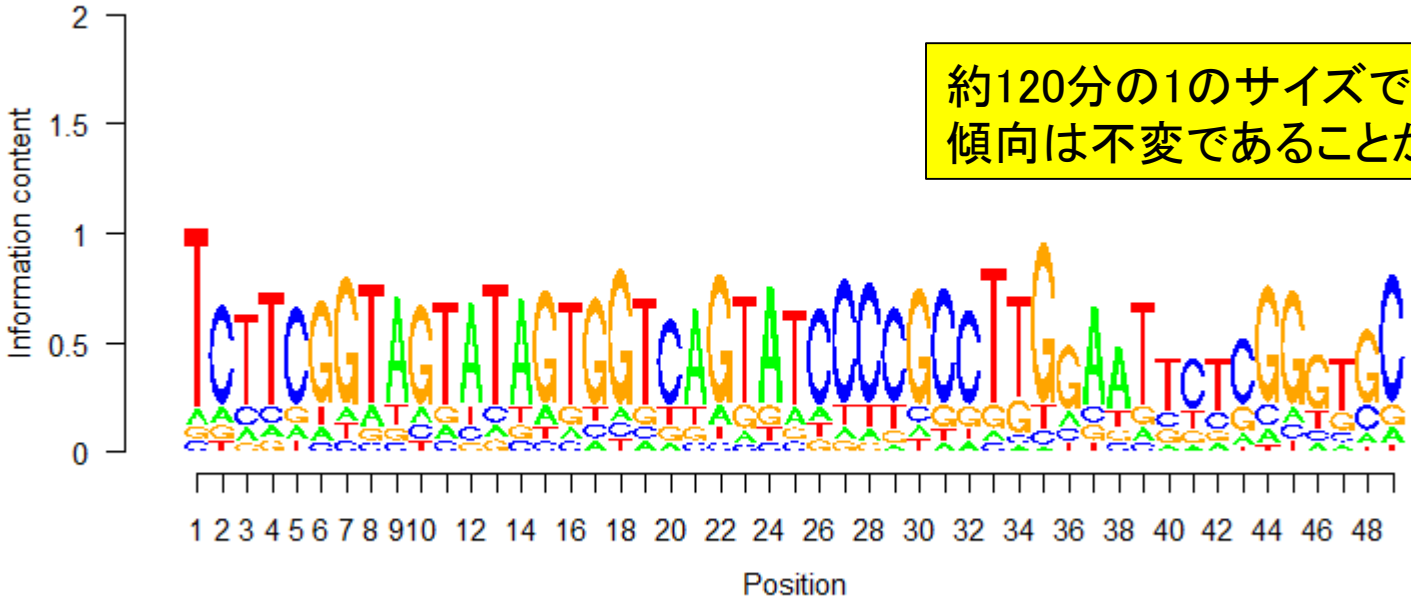
比較

neider 1990)

hoge8.png (11,928,428リードの結果)



hoge10.png (100,000リードの結果)



約120分の1のサイズでも全体的な傾向は不変であることがわかります。

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - **計算手順の説明**
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

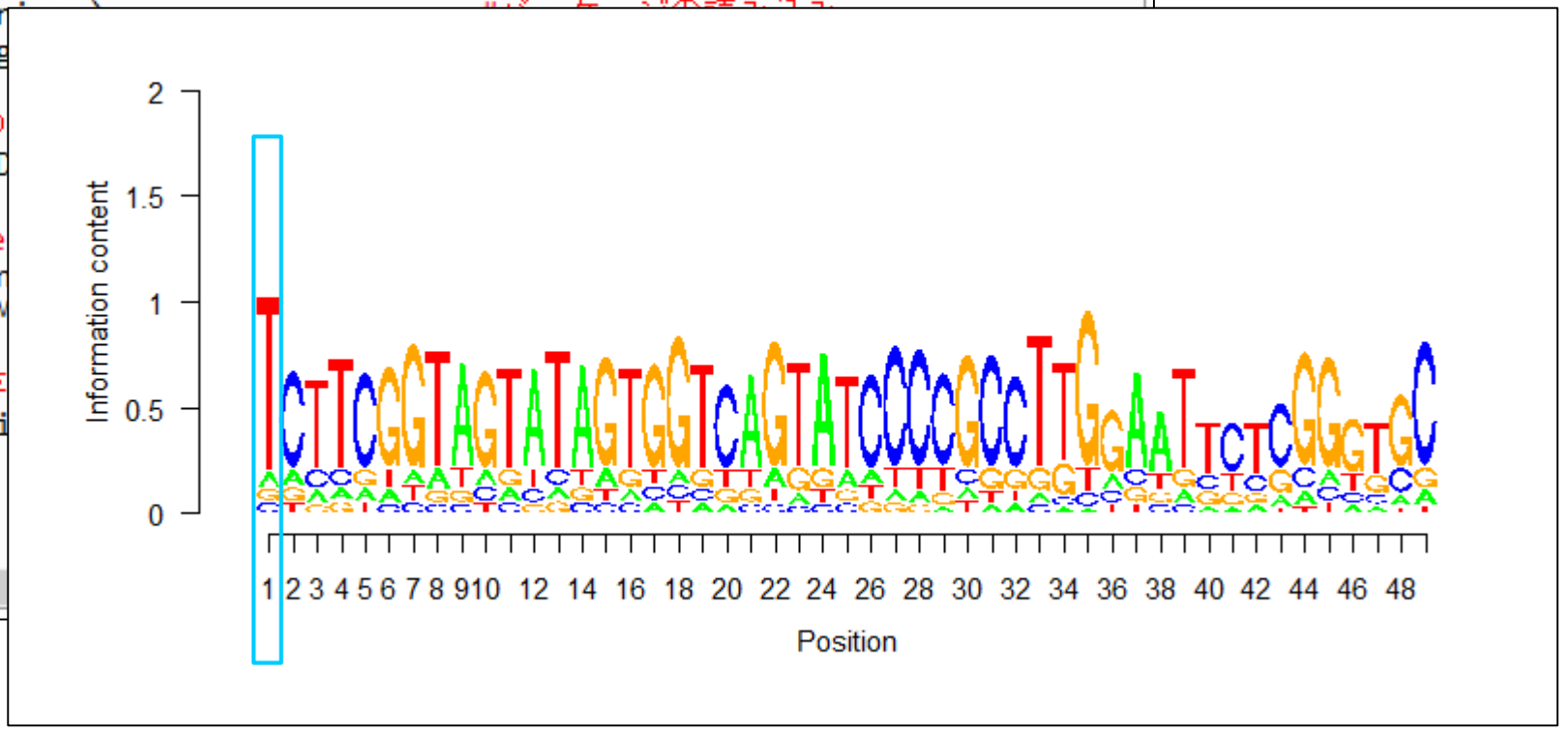
Sequence logos実行結果の解釈

10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

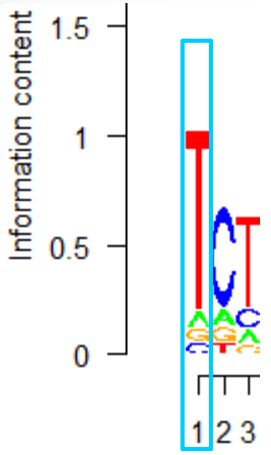
前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsmall RNA-seqデータ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定

#必要なパッケージをロード
library(Biostr
library(seqLog

#入力ファイルの
fasta <- readD

#本番(sequence
hoge <- conser
out <- makePWM

#ファイルに保存
png(out_f, poi
seqLogo(out)
dev.off()
```

全部で49塩基からなるリードの1番目のポジションはTが7割程度を占め、残りの塩基が1割程度ずつを占める、と解釈する。



Sequence logos実行結果の解釈



10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsmall RNA-seqデータ(100,000リード; 約16MB)です。

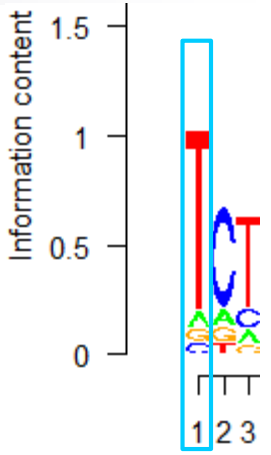
```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)
library(seqLogo)

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_

#本番(sequence logoを実行)
hoge <- consensusMatrix(fast
out <- makePWM(hoge[1:4,])

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

R Console

```
> out <- makePWM(hoge[1:4,])
>
> #ファイルに保存
> png(out_f, pointsize=13, wi
> seqLogo(out)
> dev.off()                                          #のまじない
null device
          1
> in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_$
> out_f <- "hoge10.png"                  #出力ファイル名を指定してout$
> param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅$
>
> #必要なパッケージをロード
> library(Biostrings)                    #パッケージの読み込み
> library(seqLogo)                       #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="fastq")#in_fで指定したファ$
>
```

黒枠で囲まれた部分を実行することで、fastaというオブジェクトの中に入力ファイル情報(の一部)が格納される。

Sequence logos実行結果の解釈



10. FASTQ形式ファイル(SRR609266_sub.fastq)の場合:

前処理 | フィルタリング | 任意のリード(サブセット)を抽出 の8.を実行して得られたsmall RNA-seqデータ(100,000リード; 約16MB)です。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位

#必要なパッケージをロード
library(Biostrings)
library(seqLogo)

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_

#本番(sequence logoを実行)
hoge <- consensusMatrix(fast
out <- makePWM(hoge[1:4,])

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

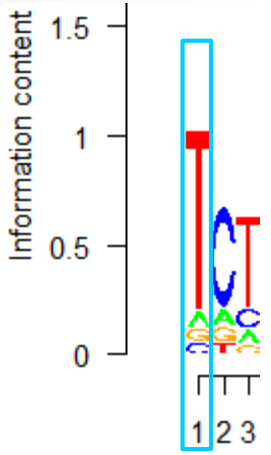
R Console

```
> library(seqLogo)
>
> #入力ファイルの読み込み
> fasta <- readDNAStringSet(in_f, forma
> fasta
  A DNAStringSet instance of length 100000
         width seq                                               names
     [1]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.17 FCC0...
     [2]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.93 FCC0...
     [3]    49 TATTGTGGACAACTCG...GTGCCAAGGAACTCC SRR609266.179 FCC...
     [4]    49 CTCGCGGGTCGATGAA...GCCAAGGAACTCCAG SRR609266.310 FCC...
     [5]    49 AAATCATTACCCTGGA...GCGGGTGGAATTCTC SRR609266.462 FCC...
     ...   ... ...
 [99996]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192768...
 [99997]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192770...
 [99998]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192788...
 [99999]    49 TCTTCGGTAGTATAGT...GAATTCTCGGGTGCC SRR609266.1192805...
[100000]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192805...
>
```

fastaオブジェクト中には、widthに配列長、seqにリード塩基配列、namesにdescription情報が含まれていることがわかる。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)                    #パッケージの読み込み
library(seqLogo)                       #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="fastq")#in_fで指定したファイルの読みi

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])             #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

• 解析 | 一般 | Sequence logos(Schneider 1990)



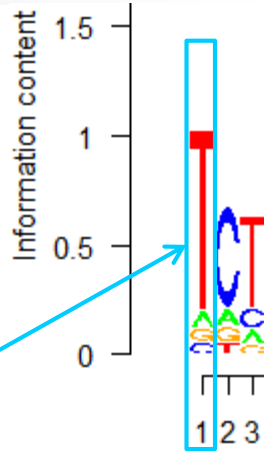
R Console

```
[100000]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192805...
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジション$
> dim(hoge)
[1]  5 49
> hoge
          [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
A      0.06986 0.11697 0.10234 0.08205 0.09633 0.09279 0.09053 0.13074
C      0.05186 0.70833 0.14691 0.11037 0.70610 0.06400 0.06491 0.05886
G      0.05654 0.09745 0.06470 0.07838 0.11479 0.71257 0.75528 0.07441
T      0.81998 0.07709 0.68595 0.72919 0.08277 0.13063 0.08928 0.73599
other  0.00176 0.00016 0.00010 0.00001 0.00001 0.00001 0.00000 0.00000
          [,9]   [,10]   [,11]   [,12]   [,13]   [,14]   [,15]   [,16]
A      0.71732 0.10897 0.09355 0.71019 0.08017 0.71808 0.12496 0.08746
C      0
G      0
T      0
other  0

A      0
```

consensusMatrix関数はポジションごとの塩基組成を計算しているだけです。計算結果を格納したhogeオブジェクトは5行×49列からなる数値行列。列数はリード数に相当し、5行である理由は、ACGT以外の文字をotherとして計数しているから。

```
in_f <- "SRR609266_sub.fastq"              #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                      #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)                   #ファイル出力時の横幅と縦幅を指定(単位

#必要なパッケージをロード
library(Biostrings)                        #パッケージの読み込み
library(seqLogo)                           #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="fastq")#in_fで指定したファイルの読み

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])                 #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

• 解析 | 一般 | Sequence logos(Schneider 1990)



R Console

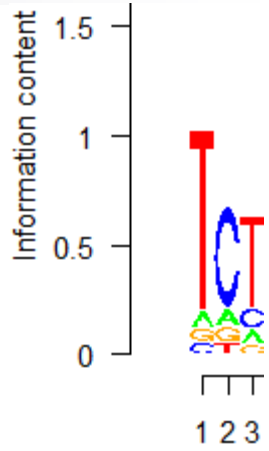
```
[100000]    49 TCTTCGGTAGTATAGT...GGAATTCTCGGGTGC SRR609266.1192805...
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジション$
> dim(hoge)
[1]  5 49
> hoge
        [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
A     0.06986 0.11697 0.10234 0.08205 0.09633 0.09279 0.09053 0.13074
C     0.05186 0.70833 0.14691 0.11037 0.70610 0.06400 0.06491 0.05886
G     0.05654 0.09745 0.06470 0.07838 0.11479 0.71257 0.75528 0.07441
T     0.81998 0.07709 0.68595 0.72919 0.08277 0.13063 0.08928 0.73599
other 0.00176 0.00016 0.00010 0.00001 0.00001 0.00001 0.00000 0.00000
        [,9]   [,10]   [,11]   [,12]   [,13]   [,14]   [,15]   [,16]
A     0.71732 0.10897 0.09355 0.71019 0.08017 0.71808 0.12496 0.08746
C     0
G     0
T     0
other 0

A     0
```

1番目のポジションは、Aが6.986%, Cが5.186%, Gが5.654%, Tが81.998%の組成比であったことがわかる。この組成比がsequence logosにおける文字の長さ比に相当する。

```
in_f <- "SRR609266_sub.fastq"              #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                      #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)                   #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)                        #パッケージの読み込み
library(seqLogo)                           #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="fastq")#in_fで指定したファイルの読み込

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])                 #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

参考

• 解析 | 一般 | Sequence logos(Schneider 1990)



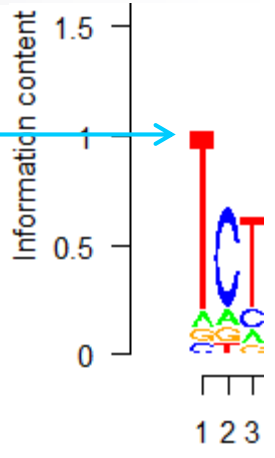
R Console

```
>
> uge <- consensusMatrix(fasta, as.prob=F, baseOnly=T)#各ポジションの$
> dim(uge)
[1]  5 49
> uge
       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
A      6986 11697 10234  8205  9633  9279  9053 13074 71732 10897
C      5186 70833 14691 11037 70610  6400  6491  5886  6342 10636
G      5654  9745  6470  7838 11479 71257 75528  7441  8214 70874
T     81998  7709 68595 72919  8277 13063  8928 73599 13712  7593
other   176    16    10     1     1     1     0     0     0     0
       [,11] [,12] [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20]
A       9355 71019  8017 71808 12496  8746  7814  7984  7522  7051
C       7647
G      11592
T      71406 1
other      0
       [,21] [
```

出現確率ではなく出現頻度情報を得たい場合は、確率として出力するか否かを指定するas.probというオプションを(TRUEを意味する)Tから(FALSEを意味する)Fにすればよい。このデータは全部で10万リードからなるので、小数点の位置が変わっただけのように見える。

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)                    #パッケージの読み込み
library(seqLogo)                       #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="fastq")#in_fで指定したファイルの読み込

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])             #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

• 解析 | 一般 | Sequence logos(Schneider 1990)



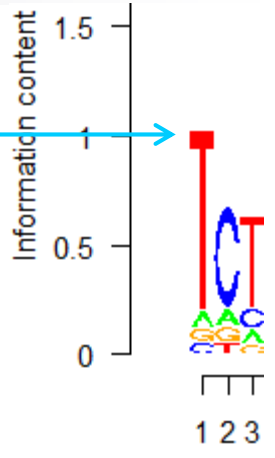
R Console

```
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジション$
> out <- makePWM(hoge[1:4,])                    #hogeはACGT以外の塩基(例えば$
> out@ic
 [1] 1.0412504 0.6731075 0.6283423 0.7329244 0.6642823 0.6959703
 [7] 0.8131625 0.7713118 0.7146064 0.6732465 0.6892893 0.6892965
[13] 0.7751348 0.7011329 0.7453320 0.6883483 0.7095412 0.8491315
[19] 0.7045857 0.6073074 0.6517864 0.8241109 0.7157949 0.7576213
[25] 0.6435576 0.6576795 0.
[31] 0.7549008 0.6498298 0.
[37] 0.6661977 0.4731273 0.
[43] 0.5134507 0.7690447 0.
[49] 0.8238861
> out
        1      2      3      4      5      6      7      8      9
A 0.0699 0.1170 0.1023 0.0820 0.0963 0.0928 0.0905 0.1307 0.7173
C 0.0519 0.7083 0.1469 0.1104 0.7061 0.0640 0.0649 0.0589 0.0634
G 0.0565 0.0974 0.0647 0.0784 0.1148 0.7126 0.7553 0.0744 0.0821
T 0.8200 0.0771 0.6859 0.7292 0.0828 0.1306 0.0893 0.7360 0.1371
```

sequence logosの縦軸の値(情報量; information content; ic)は、makePWM関数実行結果のoutオブジェクト中に存在する

```
in_f <- "SRR609266_sub.fastq"          #入力ファイル名を指定してin_fに格納
out_f <- "hoge10.png"                  #出力ファイル名を指定してout_fに格納
param_fig <- c(800, 370)               #ファイル出力時の横幅と縦幅を指定(単位I

#必要なパッケージをロード
library(Biostrings)                    #パッケージの読み込み
library(seqLogo)                       #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="fastq")#in_fで指定したファイルの読み込

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジションの塩基組成
out <- makePWM(hoge[1:4,])             #hogeはACGT以外の塩基(例えばN)のprobal

#ファイルに保存
png(out_f, pointsize=13, widt
seqLogo(out)
dev.off()
```

• 解析 | 一般 | Sequence logos(Schneider 1990)



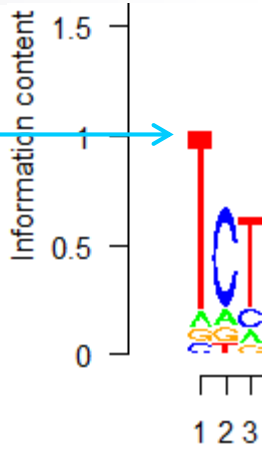
R Console

```
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジション$
> out <- makePWM(hoge[1:4,])                    #hogeはACGT以外の塩基(例えば$
> out@ic
 [1] 1.0412504 0.6731075 0.6283423 0.7329244 0.6642823 0.6959703
 [7] 0.8131625 0.7713118 0.7146064 0.6732465 0.6892893 0.6892965
[13] 0.7751348 0.7011329 0.7453320 0.6883483 0.7095412 0.8491315
[19] 0.7045857 0.6073074 0.6517864 0.8241109 0.7157949 0.7576213
[25] 0.6435576 0.6576795 0.
[31] 0.7549008 0.6498298 0.
[37] 0.6661977 0.4731273 0.
[43] 0.5134507 0.7690447 0.
[49] 0.8238861
> out
        1      2      3      4      5      6      7      8      9
A 0.0699 0.1170 0.1023 0.0820 0.0963 0.0928 0.0905 0.1307 0.7173
C 0.0519 0.7083 0.1469 0.1104 0.7061 0.0640 0.0649 0.0589 0.0634
G 0.0565 0.0974 0.0647 0.0784 0.1148 0.7126 0.7553 0.0744 0.0821
T 0.8200 0.0771 0.6859 0.7292 0.0828 0.1306 0.0893 0.7360 0.1371
```

縦軸の情報量の値(=1.0412504)は、出現確率のみから計算することができます。

20141004_ic.txt (の下のほう)

```
##################################↓
###  情報量計算手段(その6)  ###↓
##################################↓
p <- c(0.06986, 0.05186, 0.05654, 0.81998)# A, C, G, Tの出現確率↓
↓
N <- length(p)↓
H <- sum(-p*log2(p))↓
log2(N) - H↓
←
```



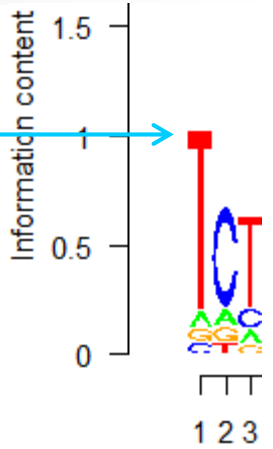
R Console

```
> p <- c(0.06986, 0.05186, 0.05654, 0.81998)# A, C, G, Tの出現確率
>
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 1.04125
>
> p
[1] 0.06986 0.05186 0.05654 0.81998
> N
[1] 4
> log2(N)
[1] 2
> H
[1] 0.9587496
> |
```

縦軸の情報量の値(=1.0412504)は、出現確率情報を格納したオブジェクトpのみから計算されていることがわかります。

20141004_ic.txt (の下のほう)

```
##################################↓
###  情報量計算手段(その6)  ###↓
##################################↓
p <- c(0.06986, 0.05186, 0.05654, 0.81998)# A, C, G, Tの出現確率↓
↓
N <- length(p)↓
H <- sum(-p*log2(p))↓
log2(N) - H↓
←
```



R Console

```
> p <- c(0.06986, 0.05186, 0.05654, 0.81998)# A, C, G, Tの出現確率
>
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 1.04125
>
> p
[1] 0.06986 0.05186 0.05654 0.81998
> N
[1] 4
> log2(N)
[1] 2
> H
[1] 0.9587496
>
```

個別のオブジェクトの中身を表示させているだけです。
Nは塩基の種類数、Hはエントロピーです。

仮想データで全体的なイメージをつかむ

20141004_ic.txt (の下のほう)

```
################################
###  情報量(仮想データ)     ###
################################
p <- c(0.997, 0.001, 0.001, 0.001)#
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H

p <- c(0.25, 0.25, 0.25, 0.25)    #
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H

p <- c(0.499, 0.499, 0.001, 0.001)#
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H
```

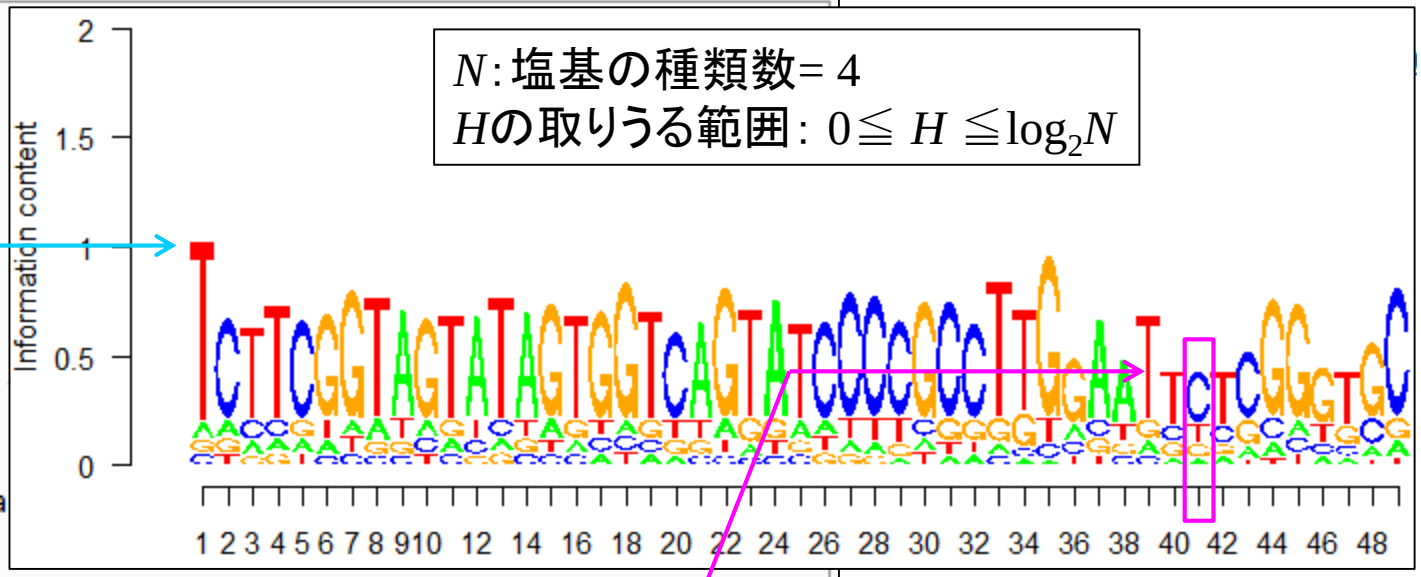
R Console

```
> p <- c(0.997, 0.001, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 1.965781
> H
[1] 0.03421894
>
> p <- c(0.25, 0.25, 0.25, 0.25)     # A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0
> H
[1] 2
>
> p <- c(0.499, 0.499, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0.9791859
> H
[1] 1.020814
>
```

特定の塩基のみの出現確率が高い場合には**低いエントロピー**。情報量の値は大きい。

塩基の出現確率が等しい場合には**高いエントロピー**。情報量の値は小さい。

```
in_f <- "SRR609266_sub.fastq"
out_f <- "hoge10.png"
param_fig <- c(800, 370)

#必要なパッケージをロード
library(Biostrings)
library(seqLogo)

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_f,

#本番(sequence logoを実行)
hoge <- consensusMatrix(fasta, a
out <- makePWM(hoge[1:4,])

#ファイル
png(out_
seqLogo(
dev.off(
```



N: 塩基の種類数= 4
Hの取りうる範囲: $0 \leqq H \leqq \log_2 N$

R Console

```
> hoge <- consensusMatrix(fasta, as.prob=T, baseOnly=T)#各ポジション$
> out <- makePWM(hoge[1:4,])                    #hogeはACGT以外の塩基(例えば$
> out@ic
 [1] 1.0412504 0.6731075 0.6283423 0.7329244 0.6642828 0.6959703
 [7] 0.8131625 0.7713118 0.7146064 0.6732465 0.6892893 0.6892965
[13] 0.7751348 0.7011329 0.7453320 0.6883483 0.7095412 0.8491315
[19] 0.7045857 0.6073074 0.6517864 0.8241109 0.7157949 0.7576213
[25] 0.6435576 0.6576795 0.7988209 0.7848888 0.6626476 0.7556507
[31] 0.7549008 0.6498298 0.8526858 0.7098731 0.9771971 0.4794189
[37] 0.6661977 0.4731273 0.6805330 0.4180461 0.4208357 0.4147627
[43] 0.5134507 0.7690447 0.7460059 0.4399337 0.4128287 0.5569739
[49] 0.8238861
> out
        1      2      3      4
A 0.0699 0.1170 0.1023 0.0820 0.
C 0.0519 0.7083 0.1469 0.1104 0.
G 0.0565 0.0974 0.0647 0.0784 0.
T 0.8200 0.0771 0.6859 0.7292 0.0828 0.1306 0.0893 0.7360 0.1371
```

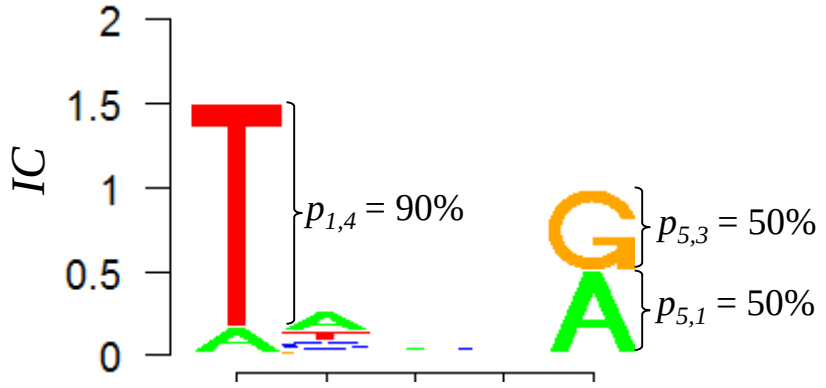
特定の塩基のみの出現確率が高いポジションほど、エントロピーが低くなる。しかし意味のあるものほど縦軸の値が大きくなるようにしたいので、情報量を用いて表現している。

他の仮想データの計算例

- position iの情報量 $IC_i = \log_2(N) - H(x_i)$
  $= 2$

position i

	1	2	3	4	5	…
配列 1	T	A	C	G	G	…
配列 2	T	A	A	C	G	…
配列 3	T	G	T	A	G	…
配列 4	A	C	T	T	A	…
配列 5	T	T	G	G	A	…
配列 6	T	C	A	A	G	…
配列 7	T	A	C	T	A	…
配列 8	T	T	G	C	A	…
配列 9	T	A	A	C	A	…
配列10	T	A	C	T	G	…



IC	1.53	0.24	0.03	0.03	1.00	…

x_{ij}	1	2	3	4	5	…
Aの数 (j=1)	1	5	3	2	5	…
Cの数 (j=2)	0	2	3	3	0	…
Gの数 (j=3)	0	1	2	2	5	…
Tの数 (j=4)	9	2	2	3	0	…
$\Sigma_j\ x_{ij}$	10	10	10	10	10	

j

p_{ij}	1	2	3	4	5	…
1	0.1	0.5	0.3	0.2	0.5	…
2	0.0	0.2	0.3	0.3	0.0	…
3	0.0	0.1	0.2	0.2	0.5	…
4	0.9	0.2	0.2	0.3	0.0	…
Σ_j	1.0	1.0	1.0	1.0	1.0	

$-p_{ij}\log_2(p_{ij})$	1	2	3	4	5	…
1	0.33	0.50	0.52	0.46	0.50	…
2	0.00	0.46	0.52	0.52	0.00	…
3	0.00	0.33	0.46	0.46	0.50	…
4	0.14	0.46	0.46	0.52	0.00	…
$H = \Sigma_j$	0.47	1.76	1.97	1.97	1.00	

水色の枠内がエントロピーの値

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- **組織特異的遺伝子の検出(内部的にエントロピーを利用)**
  - **基本形(Schug et al., *Genome Biol.*, 2005)**
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

仮想データで全体的なイメージをつかむ

20141004_ic.txt (の下のほう)

```
################################↓
###  情報量(仮想データ)     ###↓
################################↓
p <- c(0.997, 0.001, 0.001, 0.001)# 
N <- length(p)↓
H <- sum(-p*log2(p))↓
log2(N) - H↓
H↓
↓
p <- c(0.25, 0.25, 0.25, 0.25)    # 
N <- length(p)↓
H <- sum(-p*log2(p))↓
log2(N) - H↓
H↓
↓
p <- c(0.499, 0.499, 0.001, 0.001)# 
N <- length(p)↓
H <- sum(-p*log2(p))↓
log2(N) - H↓
H↓
←
```

R Console

```
> p <- c(0.997, 0.001, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 1.965781
> H
[1] 0.03421894
>
> p <- c(0.25, 0.25, 0.25, 0.25)    # A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0
> H
[1] 2
>
> p <- c(0.499, 0.499, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0.9791859
> H
[1] 1.020814
>
```

ある遺伝子(gene1)の4つの組織(tissue1-4)における相対発現レベルだと解釈すればよい。gene1はtissue1特異的高発現遺伝子。

gene2はどの組織でも同程度の発現レベル。

gene3はtissue1と2で高発現、それ以外で低発現。

仮想データで全体的なイメージをつかむ

20141004_ic.txt (の下のほう)

```
################################
###  情報量(仮想データ)     ###
################################
p <- c(0.997, 0.001, 0.001, 0.001)#
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H

p <- c(0.25, 0.25, 0.25, 0.25)    #
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H

p <- c(0.499, 0.499, 0.001, 0.001)#
N <- length(p)
H <- sum(-p*log2(p))
log2(N) - H
H
```

R Console

```
> p <- c(0.997, 0.001, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 1.965781
> H
[1] 0.03421894
>
> p <- c(0.25, 0.25, 0.25, 0.25)    # A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0
> H
[1] 2
>
> p <- c(0.499, 0.499, 0.001, 0.001)# A, C, G, Tの出現確率
> N <- length(p)
> H <- sum(-p*log2(p))
> log2(N) - H
[1] 0.9791859
> H
[1] 1.020814
>
```

(情報量でもよいが)遺伝子ごとにエントロピー*H*を計算しておき、*H*の低いものが全体的な組織特異性が高いと判断すればよい。

仮想データで全体的なイメージをつかむ

20141004_roku.txt (の上のほう)

```
####################################↓
### エントロピー(ACGT出現確率) ###↓
####################################↓
library(TCC)↓
↓
p <- c(0.997, 0.001, 0.001, 0.001)# A, C, G, Tの出現確率↓
out <- ROKU(matrix(p, nrow=1)) # ROKU法の実行↓
out$H                          # エン
↓
p <- c(0.25, 0.25, 0.25, 0.25) # A, C
out <- ROKU(matrix(p, nrow=1)) # ROKU
out$H                          # エン
↓
p <- c(0.499, 0.499, 0.001, 0.001)# A
out <- ROKU(matrix(p, nrow=1)) # ROKU
out$H                          # エン
```

TCCパッケージ中のROKU関数を用いてエントロピー計算することもできる

R Console

```
> library(TCC)
>
> p <- c(0.997, 0.001, 0.001, 0.001)# A, C, G, Tの出現確率
> out <- ROKU(matrix(p, nrow=1)) # ROKU法の実行
> out$H
[1] 0.03421894
>
> p <- c(0.25, 0.25, 0.25, 0.
> out <- ROKU(matrix(p, nrow=
> out$H
[1] 2
>
> p <- c(0.499, 0.499, 0.001, 0.001)# A, C, G, Tの出現確率
> out <- ROKU(matrix(p, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 1.020814
>
```

(情報量でもよいが)遺伝子ごとにエントロピー*H*を計算しておき、*H*の低いものが全体的な組織特異性が高いと判断すればよい。

仮想データで全体的なイメージをつかむ

20141004_roku.txt (の上のほう)

```
####################################↓
### エントロピー(ACGT出現頻度) ###↓
####################################↓
library(TCC)↓
↓
x <- c(997, 1, 1, 1)              # A, C, G, Tの出現頻度↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行↓
out$H                             # エン
↓
x <- c(25, 25, 25, 25)            # A, C
out <- ROKU(matrix(x, nrow=1)) # ROKU
out$H                             # エン
↓
x <- c(499, 499, 1, 1)            # A, C
out <- ROKU(matrix(x, nrow=1)) # ROKU
out$H                             # エン
↓
```

TCCパッケージ中のROKU関数は、出現**確率**でなく出現**頻度**を入力としてもエントロピー計算することもできる

R Console

```
> library(TCC)
>
> x <- c(997, 1, 1, 1)              # A, C, G, Tの出現頻度
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H
[1] 0.03421894
>
> x <- c(25, 25, 25, 25)
> out <- ROKU(matrix(x, nrow=
> out$H
[1] 2
>
> x <- c(499, 499, 1, 1)            # A, C, G, Tの出現頻度
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                             # エントロピーHを表示
[1] 1.020814
>
```

(情報量でもよいが)遺伝子ごとにエントロピー*H*を計算しておき、*H*の低いものが全体的な組織特異性が高いと判断すればよい。

仮想データで全体的なイメージをつかむ

20141004_roku.txt (の真ん中あたり)

```
#####################################↓
### エントロピー(発現データ)   ###↓
#####################################↓
library(TCC)↓
↓
x <- c(997, 1, 1, 1)              # tissue1-4の発現データ
out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
out$H                             # エン
↓
x <- c(25, 25, 25, 25)            # tiss
out <- ROKU(matrix(x, nrow=1)) # ROKU
out$H                             # エン
↓
x <- c(499, 499, 1, 1)            # tiss
out <- ROKU(matrix(x, nrow=1)) # ROKU
out$H                             # エン
↓
```

ACGTの出現**頻度**をそのままtissue1-4とした場合。ポジションごとから遺伝子ごとのエントロピー計算に転用可能。

R Console

```
> library(TCC)
>
> x <- c(997, 1, 1, 1)              # tissue1-4の発現データ
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H
[1] 0.03421894
>
> x <- c(25, 25, 25, 25)
> out <- ROKU(matrix(x, nrow=
> out$H
[1] 2
>
> x <- c(499, 499, 1, 1)            # tissue1-4の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                             # エントロピーHを表示
[1] 1.020814
>
```

(情報量でもよいが)遺伝子ごとにエントロピー*H*を計算しておき、*H*の低いものが全体的な組織特異性が高いと判断すればよい。

仮想データで全体的なイメージをつかむ

20141004_roku.txt (の真ん中あたり)

```
#####################################↓
### エントロピー(不都合な例)  ###↓
#####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

8組織分の仮想発現データ。特異的高発現組織以外の発現レベルが0の場合にエントロピーが最小値となる。

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0, 0, 0, 0)  # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6)  # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4)  # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 2.958981
>
```

仮想データで全体的なイメージをつかむ

20141004_roku.txt (の真ん中あたり)

```
#####################################↓
### エントロピー(不都合な例)  ###↓
#####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

8組織分の仮想発現データ。全組織で一定の発現レベルの場合にエントロピーが最大値($\log_2 8 = 3$)となる。

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
>
```

不都合な例も存在（ROKU法開発の動機）

20141004_roku.txt（の真ん中あたり）

```
####################################↓
### エントロピー(不都合な例)  ###↓
####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

8組織分の仮想発現データ。特異的発現以外の組織の発現レベルが比較的高い場合にエントロピーが最大値(= 3)に近い値となり、うまくランキングできない（こととその対応策を示したのがROKU）。

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0
> out <- ROKU(matrix(x
> out$H                          # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
>
```

不都合な例も存在（ROKU法開発の動機）

20141004_roku.txt（の真ん中あたり）

```
####################################↓
### エントロピー(不都合な例)  ###↓
####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

8組織分の仮想発現データ。特異的発現以外の組織の発現レベルが比較的高い場合にエントロピーが最大値(= 3)に近い値となり、うまくランキングできない（こととその対応策を示したのがROKU）。

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0
> out <- ROKU(matrix(x
> out$H                          # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
>
```

Contents

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - **発展形(Kadota et al., *BMC Bioinformatics*, 2006)**

ROKU法の目的

20141004_roku.txt (の真ん中あたり)

```
#####################################↓
### エントロピー(不都合な例)  ###↓
#####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

特異的高発現であろうが低発現であろうが、特異的発現パターンをもつ下記3遺伝子をエントロピーの低さに基づいて上位にランクインさせたい!

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
>
```

ROKU法の戦略

20141004_roku.txt (の真ん中あたり)

```
#####################################↓
### エントロピー(不都合な例)  ###↓
#####################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
xt↓
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
↓
```

特異的高発現組織以外の発現レベルが0の場合にエントロピーが最小値となるのだから、そうなるように予めデータの変換をしておけばよい。

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
>
```

データ変換の基本形

20141004_roku.txt (の下のほう)

```
##########################################↓
### エントロピー(データ変換の基本形)  ###↓
##########################################
library(TCC)↓
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
t <- x - median(x)             # データ
t                              # 変換後
out <- ROKU(matrix(t, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
t <- x - median(x)             # データ
t                              # 変換後
out <- ROKU(matrix(t, nrow=1)) # ROKU法
out$H                          # エント
↓
```

R Console

```
> library(TCC)
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4)  # tissue1-8の発現データ
> t <- x - median(x)              # データ変換の基本形
> t                               # 変換後のデータを表示
[1] 0 0 0 0 6 0 0 0
> out <- ROKU(matrix(t, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 0
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> t <- x - median(x)              # データ変換の基本形
> t                               # 変換後のデータを表示
[1]  0  0  0  0 -6  0  0  0
> out <- ROKU(matrix(t, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 0
>
```

データ変換の基本は外れ値(この場合は10)に影響されない頑健な要約統計量で引く。実質的には下記例のように中央値のようなもので十分だが、ROKU原著論文中では中央値よりも頑健なTukey's biweightを利用。

データ変換のほぼ最終形

20141004_roku.txt (の下のほう)

```
##############################################↓
### エントロピー(データ変換のほぼ最終形) ###↓
##############################################
library(TCC)↓
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
t <- abs(x - median(x))        # データ
t                              # 変換後
out <- ROKU(matrix(t, nrow=1)) # ROKU法
out$H                          # エント
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
t <- abs(x - median(x))        # データ
t                              # 変換後
out <- ROKU(matrix(t, nrow=1)) # ROKU法
out$H                          # エント
↓
```

```
R Console
> library(TCC)
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4)  # tissue1-8の発現データ
> t <- abs(x - median(x))          # データ変換のほぼ最終$
> t                                # 変換後のデータを表示
[1] 0 0 0 0 6 0 0 0
> out <- ROKU(matrix(t, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 0
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> t <- abs(x - median(x))          # データ変換のほぼ最終$
> t                                # 変換後のデータを表示
[1] 0 0 0 0 6 0 0 0
> out <- ROKU(matrix(t, nrow=1))  # ROKU法の実行
> out$H                           # エントロピーHを表示
[1] 0
>
```

データ変換の基本は外れ値(この場合は10)に影響されない頑健な要約統計量で引く。変換後の発現レベルがマイナスになるのを防ぐため、abs関数を適用して絶対値をとっている。

ROKU法

20141004_roku.txt (の下のほう)

```
########################################↓
### エントロピー(ROKUの通常の手順) ###↓
########################################↓
library(TCC)↓
↓
x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
out$modH                       # 変換後
↓
x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
out$modH                       # 変換後
↓
x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
out$modH                       # 変換後
↓
x <- c(10,10,10,10, 4,10,10,10)# tissue
out <- ROKU(matrix(x, nrow=1)) # ROKU法
out$H                          # エント
out$modH                       # 変換後
←
```

R Console

```
> library(TCC)
>
> x <- c(0, 0, 0, 9, 0, 0, 0, 0) # tissue1-8の発現データ
> out <- ROKU(matrix(x,
> out$H
[1] 0
> out$modH
[1] 0
>
> x <- c(6, 6, 6, 6, 6, 6, 6, 6) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 3
> out$modH                       # 変換後のH(modH)を表示
[1] 3
>
> x <- c(4, 4, 4, 4,10, 4, 4, 4) # tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.900052
> out$modH                       # 変換後のH(modH)を表示
[1] 0
>
> x <- c(10,10,10,10, 4,10,10,10)# tissue1-8の発現データ
> out <- ROKU(matrix(x, nrow=1)) # ROKU法の実行
> out$H                          # エントロピーHを表示
[1] 2.958981
> out$modH                       # 変換後のH(modH)を表示
[1] 0
>
```

入力データ変換後にエントロピー計算を行ったものをmodified Entropy (modH)と呼び、modHでランキングすることを提唱。

(Rで)マイクロアレイデータ解析

(last m

What'

- 門田
- お知

- はじ
- 過去
- Rの
- Rの
- 使用
- サン
- 書籍
- 書籍
- 書籍
- 書籍
- 書籍
- 書籍
- 書籍

- 解析 | 発現変動 | 3群間 | 対応なし | Kruskal-Wallis(クラスカル-ウォリス) 検定 (last modified 2013/6/2)
- 解析 | 発現変動 | 多群間 | について(last modified 2013/6/2)
- 解析 | 発現変動 | 多群間 | SpeCond(Cavalli 2011) (last modified 2013/6/10)
- 解析 | 発現変動 | 多群間 | ROKU(Kadota 2006) (last modified 2014/05/30)
- 解析 | 発現変動 | 多群間 | Sprent's non-parametric method(Ge 2005) (last modified 2009/07/31)
- 解析 | 発現変動 | 多群間 | Sehug's H(x) statistic(Sehug 2005) (last modified 2011/10/13)
- 解析 | 発現変動 | 多群間 | 2006/7/11)
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |
- 解析 | 機能解析 |

- 解析 | 発現変動 | 多群間 | ROKU(Kadota 2006)

解析 | 発現変動 | 多群間 | ROKU (Kadota_2006)

TCCパッケージで提供しているROKU法(Kadota et al., 2006)を用いて、遺伝子発現行列中の遺伝子を全体的な組織特異性の度合いでランキングします。出力ファイル中の"modH"列の数値は、「ROKU論文中のAdditional file 1(Suppl.xls)の"H(x')"列の数値」と対応しています。つまり、データ変換後のエントロピー値です。"ranking"列は、modHの値でランキングした結果です。"ranking"列で昇順にソートすることで、全体的な組織特異性の度合いでランキングしていることになります。つまり、上位が「(どの組織で特異的かはこのスコアだけでは分からないが)組織特異性が高い遺伝子」ということになります。残りの結果は「1:特異的高発現、-1:特異的低発現、0:その他」からなる「外れ値行列」です。例えば、組織AとBで1, それ以外の組織で0を示す遺伝子(群)は「AとB特異的高発現遺伝子」と判断します。

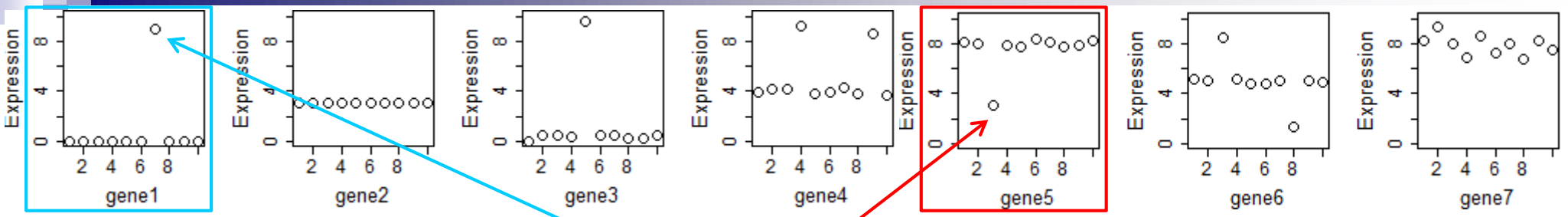
「ファイル」-「ディレクトリの変更」で解析したいファイルを置いてあるディレクトリに移動し、以下をコピペ

1. サンプルデータ21のsample21.txtの場合:

log2変換後のデータであるという前提です。

入力と出力の関係を簡単に説明します

```
in_f <- "sample21.txt"                    #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(TCC)                              #パッケージの読み込み

#入力ファイルの読み込み
data <- read.table(in_f, header=TRUE, row.names=1, sep="\t", quote="")#in_fで指定し

#本番
hoge <- ROKU(data)                        #ROKUを実行した結果をhogeに格納
outlier <- hoge$outlier                   #外れ値行列をoutlierに格納
modH <- hoge$modH                         #データ変換後のエントロピー値をmodHに格納(原著論
ranking <- hoge$rank                      #modHでランキングした結果をrankingに格納

#ファイルに保存
tmp <- cbind(rownames(data), outlier, modH, ranking)#左端の列が遺伝子ID, 次にサンプル
write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの中身を指定した
```



入力:sample21.txt

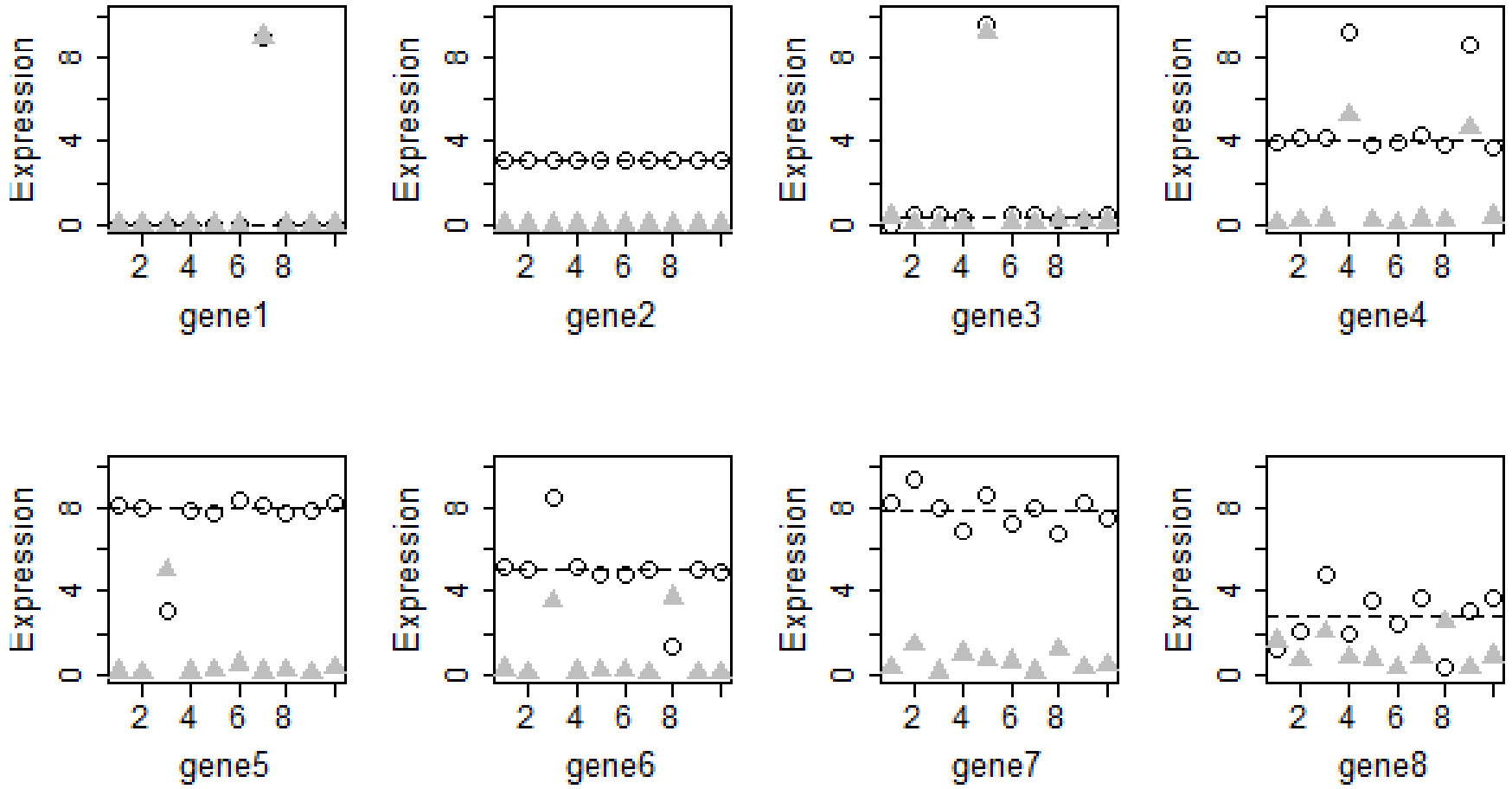
	tissue1	tissue2	tissue3	tissue4	tissue5	tissue6	tissue7	tissue8	tissue9	tissue10
gene1	0.00	0.00	0.00	0.00	0.00	0.00	9.00	0.00	0.00	0.00
gene2	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
gene3	0.02	0.41	0.41	0.38	9.60	0.49	0.44	0.16	0.21	0.52
gene4	3.95	4.12	4.20	9.20	3.84	3.97	4.23	3.80	8.60	3.64
gene5	8.06	7.93	3.00	7.82	7.75	8.42	8.06	7.75	7.88	8.26
gene6	5.20	5.00	8.50	5.10	4.84	4.78	5.00	1.30	5.00	4.89
gene7	8.20	9.30	8.00	6.90	8.60	7.30	8.00	6.70	8.20	7.50
gene8	1.20	2.10	4.80	2.00	3.50	2.50	3.65	0.30	3.10	3.63

これがデータ変換後のエントロピーとその順位

出力:hoge1.txt

	tissue1	tissue2	tissue3	tissue4	tissue5	tissue6	tissue7	tissue8	tissue9	tissue10	modH	ranking
gene1	0	0	0	0	0	0	1	0	0	0	0.000	1
gene2	0	0	0	0	0	0	0	0	0	0	3.322	8
gene3	0	0	0	0	1	0	0	0	0	0	0.768	2
gene4	0	0	0	1	0	0	0	0	1	0	1.718	5
gene5	0	0	−1	0	0	0	0	0	0	0	1.492	3
gene6	0	0	1	0	0	0	0	−1	0	0	1.645	4
gene7	0	0	0	0	0	0	0	0	0	0	2.952	6
gene8	0	0	0	0	0	0	0	0	0	0	3.032	7

エントロピー(組織特異的遺伝子検出)



ROKU法はデータの変換を行うことでよりよいエントロピーでのランキング結果を得ている(変換前:○、変換後:▲)

まとめ

- 講義資料を取得(Rでできることの全体像を把握)
- 2連続塩基出現頻度解析(CpG解析)
  - ヒトゲノム情報を含むRパッケージを入力とする場合
  - multi-FASTAファイルを入力とする場合
- Sequence logos(ポジションごとに特徴的な塩基を強調表示)
  - イントロダクション
    - small RNA-seqのカイコゲノムへのマッピング、およびアダプター配列除去前後の比較
    - マッピング結果レポートファイル中のポジションごとの塩基組成を眺めて全体像を把握
  - Sequence logosの実行
    - 実データのgzip圧縮FASTQファイル(実習なし)
    - 軽量版非圧縮FASTQファイル(実習あり)
  - 計算手順の説明
- 組織特異的遺伝子の検出(内部的にエントロピーを利用)
  - 基本形(Schug et al., *Genome Biol.*, 2005)
  - 発展形(Kadota et al., *BMC Bioinformatics*, 2006)

Sequence logosとROKU、解析目的は違っても同じエントロピーを内部的に利用。基本形から発展形への思考回路を紹介。詳細は書籍中にも記載してます。