ゲノム・トランスクリプトームの各種解析をRで行う

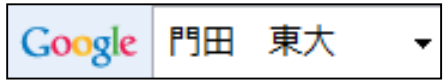
東京大学・大学院農学生命科学研究科

アグリバイオインフォマティクス教育研究プログラム

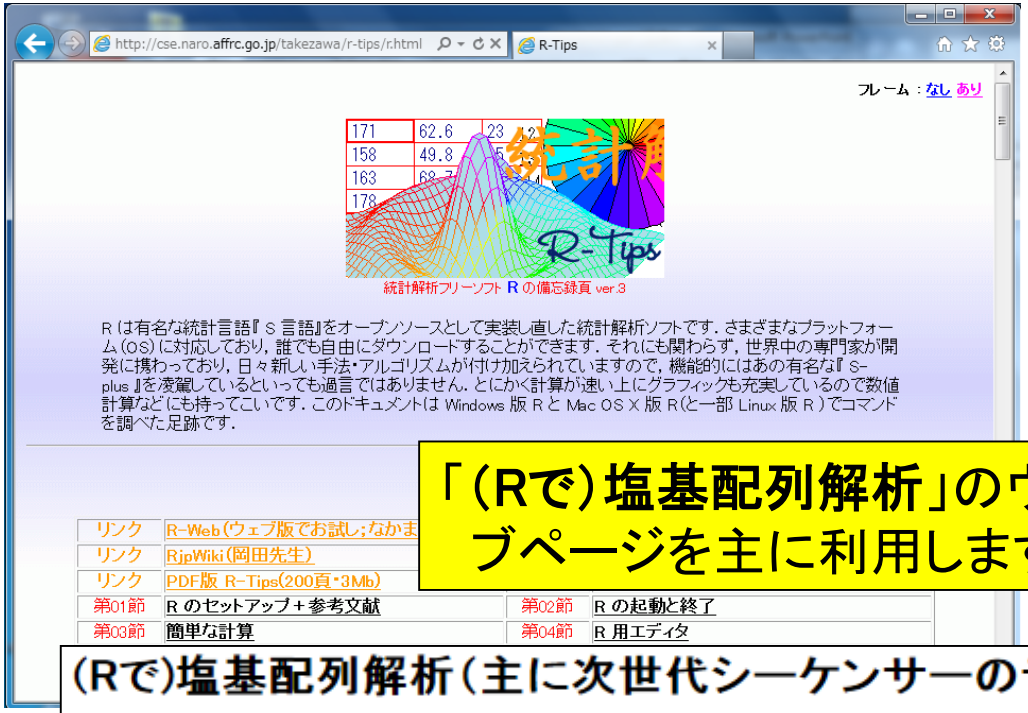
門田幸二（かどた　こうじ）

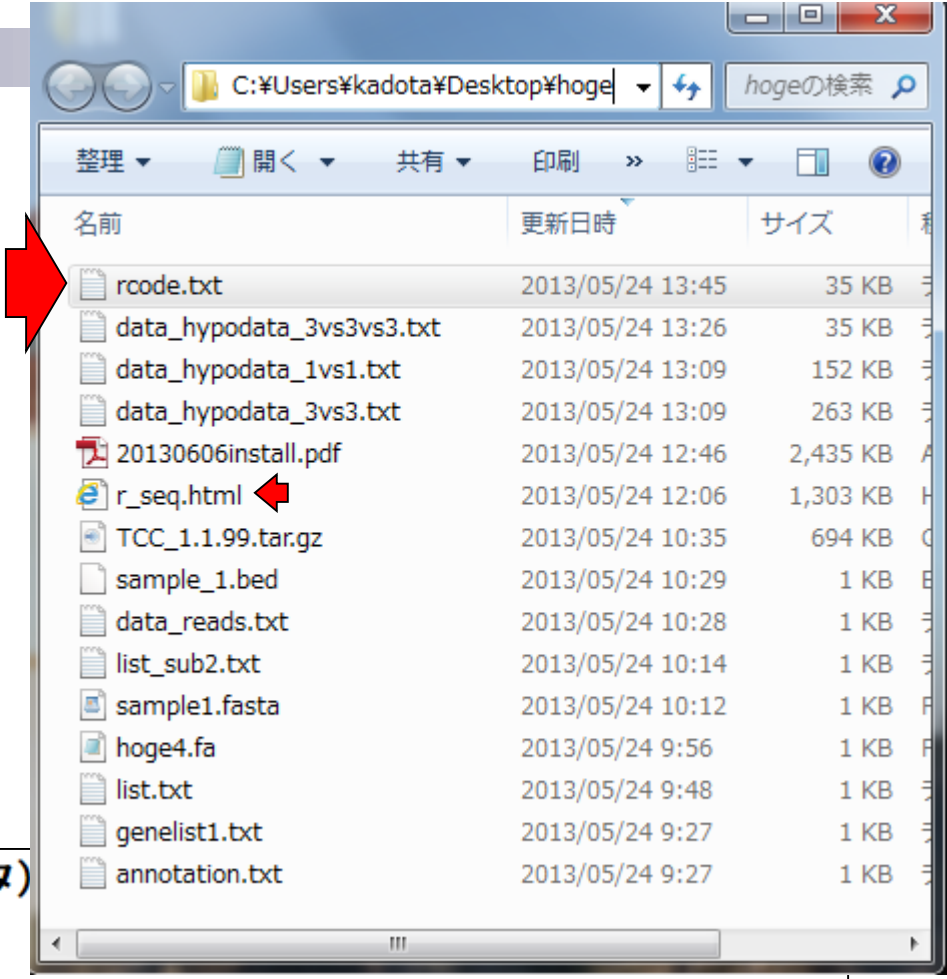
kadota@iu.a.u-tokyo.ac.jp

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



参考ウェブページ





「（Rで）塩基配列解析」のウェブページを主に利用します

(Rで)塩基配列解析(主に次世代シーケンサーのデータ)

What's new?
RパッケージTCCの最新版は1.1.99です。6月6日に開催されるNAIST植物グローバル教育プロジェクト・平成25年度ワークショップでは、R(ver. 3.0.1)とTCC(ver. 1.1.99)をベースに話をする予定です。Rのバージョン自体は2.15.3でもたぶん大丈夫だとは思いますが。。。Rのインストールと起動を実行したあとにTCC(ver. 1.1.99)のインストールも忘れずに行っておいてください。Windowsのヒト用の詳細なインストール方法はこちらです。また、当日はhoge.zipという圧縮ファイルを解凍して得られる"hoge"というフォルダがデスクトップ上にあるという前提でセミナーを行いますのでこの圧縮ファイルもダウンロードと解凍をやっておいてください。(2013/05/24)NEW

・平成25年6月27日、7月3, 4日にこのウェブページ関連の実習を含む講義(農学生命情報科学特論)を行います。東大生以外の外部の方も受講可能です。詳しくは事務局までお問い合わせください。(2013/05/23)NEW
・R3.0.1がリリースされていたのでこれに変更しました。(2013/05/17)NEW

自己紹介

- 2002年3月
  - 東京大学・大学院農学生命科学研究科・応用生命工学専攻　博士課程修了
  - 学位論文：「cDNAマイクロアレイを用いた遺伝子発現解析手法の開発」
    （指導教官：清水謙多郎教授）
- 2002/4/1~
  - 産総研・生命情報科学研究センター（CBRC）　産総研特別研究員
  - **マイクロアレイ解析手法開発**
- 2003/11/1~
  - 放医研・先端遺伝子発現研究センター　研究員
  - **一次元電気泳動波形解析手法開発**
- 2005/2/16~
  - 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プログラム
  - **マイクロアレイ解析手法開発**
  - **RNA-seqデータ解析手法開発**

（トランスクリプトーム解析周辺の）手法開発系のヒトです


+ サイトマップ + English
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
Agricultural Bioinformatics Research Unit
受講生の方へ
研究者の方へ
+ ホーム
+ 本ユニットについて
+ メンバー
+ 教育プログラム
+ 研究フォーラム
+ イベント
+ お問い合わせ
+ リンク
+ モバイルサイト
東京大学
THE UNIVERSITY OF TOKYO
ホーム > 教育プログラム > 各講義のページ
各講義のページ
（科目名をクリックすると各講義のページに移動します）
先端トピックス
セミナー・討論形式研究指導
農学生命情報科学特別演習
農学生命情報科学特論Ⅰ
農学生命情報科学特論Ⅱ
農学生命情報科学特論Ⅲ
農学生命情報科学特論Ⅳ
方法論
講義・実習を一体化
生物配列統計学
システム生物学概論
知識情報処理論
オーム情報解析
機能ゲノム学
分子モデリングと分子シミュレーション
基礎
講義・実習を一体化
ゲノム情報解析基礎
構造バイオインフォマティクス基礎
生物配列解析基礎
バイオスタティスティクス基礎論
カテゴリー
科目名
学期・単位
実施曜日
基礎
1. 生物配列解析基礎
生命科学のためのデータベースの利用と基本的な解析手法について講義します。データベースの基礎、配列データベース、機能データベース、ホモロジー検索、モチーフ解析などの基本的な手法について解説します。
夏・1
火曜

講義風景

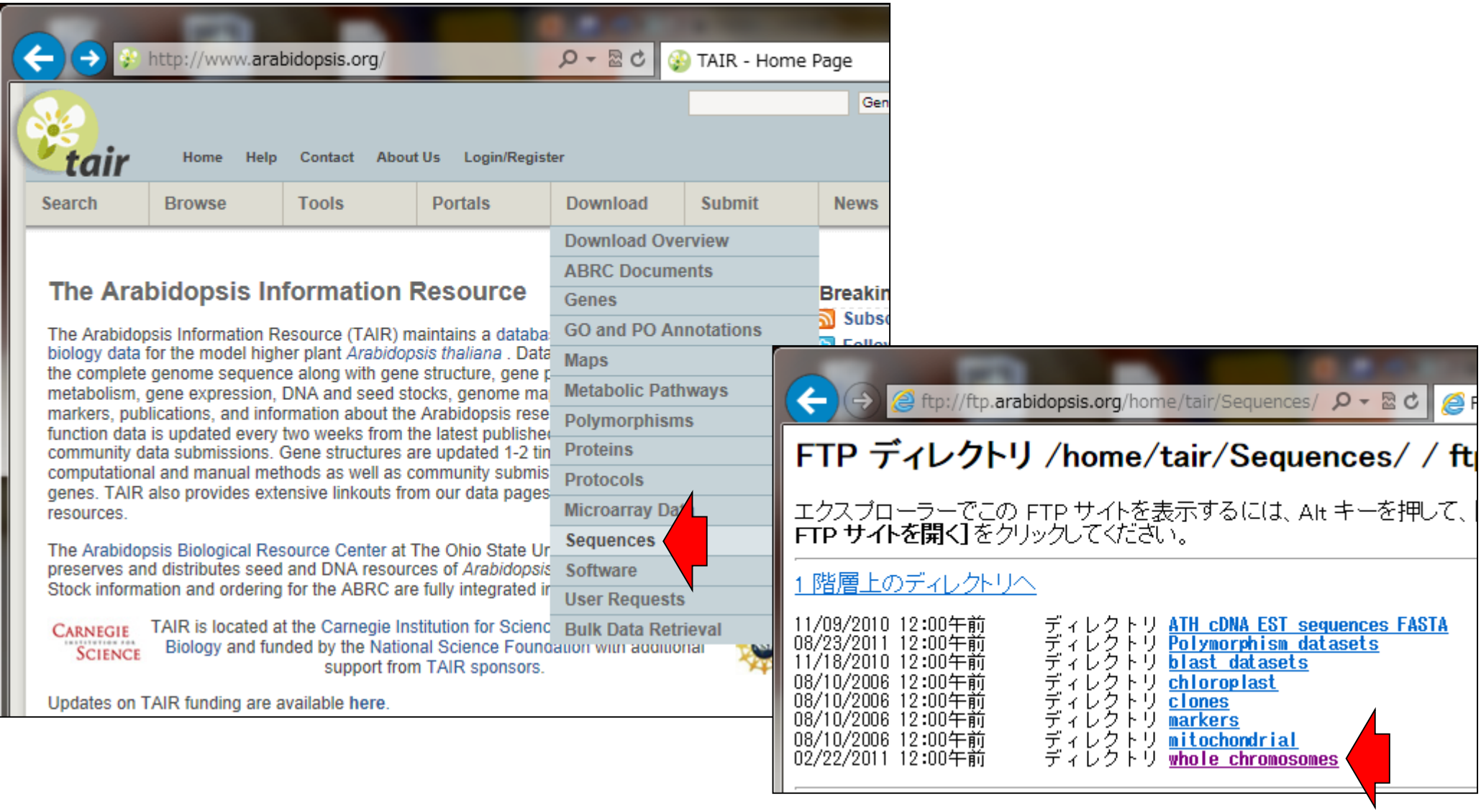
ゲノム・トランスクリプトーム解析

- 例：シロイヌナズナ（*Arabidopsis thaliana*）
  - ゲノム配列決定（chr1-5, chrC, chrM）
    - 1番染色体：Theologis et al., *Nature*, **408**: 816-820, 2000
    - 2番染色体：Lin et al., *Nature*, **402**: 761-768, 1999
    - 3番染色体：Salanoubat et al., *Nature*, **408**: 820-822, 2000
    - …
  - トランスクリプトーム配列（cDNA配列）決定
    - アノテーション：Seki et al., *Science*, **296**: 141-145, 2002
    - …
  - まとめサイト
    - The Arabidopsis Information Resource (TAIR)
    - Lamesch et al., *Nucleic Acids Res*., **40**(Database issue): D1202-1210, 2011
    - http://www.arabidopsis.org/

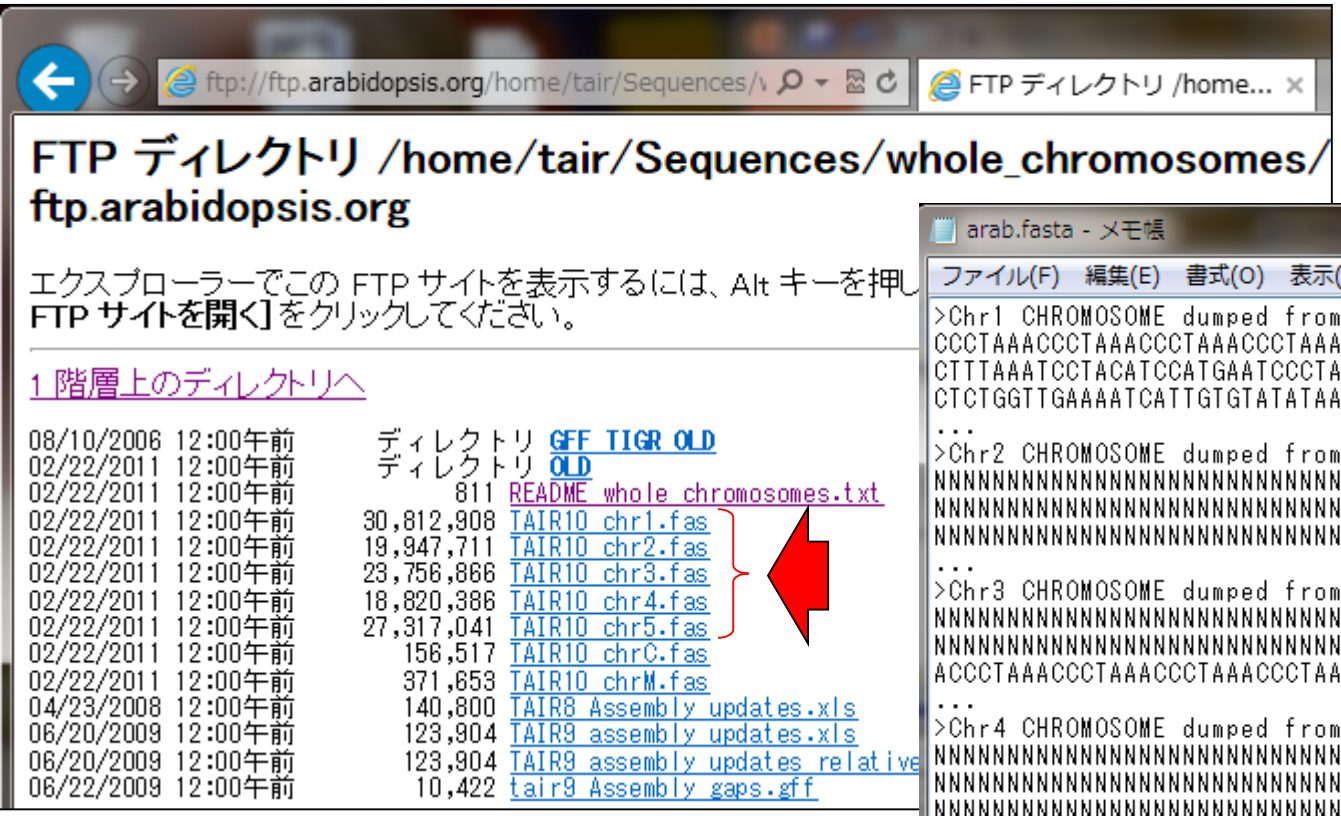
	Length	GC contents
chr1	28.76MB	35.80%
chr2	19.60MB	35.80%
chr3	23.17MB	35.40%
chr4	17.40MB	36.02%
chr5	25.95MB	34.50%

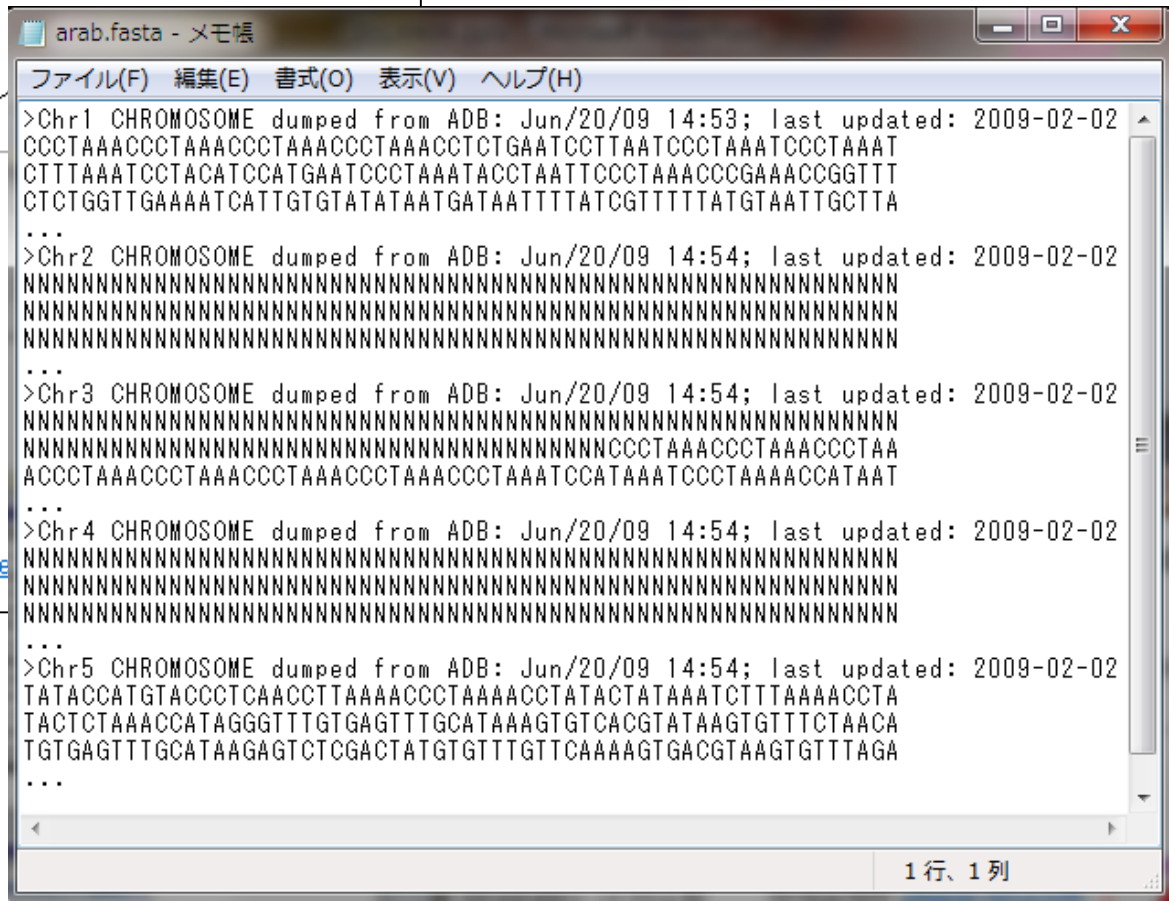
Rは論文中に書かれている
ことの一部を再現?!できます

ゲノムデータ取得


http://www.arabidopsis.org/
TAIR - Home Page
tair
Home
Help
Contact
About Us
Login/Register
Search
Browse
Tools
Portals
Download
Submit
News
Download Overview
ABRC Documents
Genes
GO and PO Annotations
Maps
Metabolic Pathways
Polymorphisms
Proteins
Protocols
Microarray Da
Sequences
Software
User Requests
Bulk Data Retrieval
The Arabidopsis Information Resource
The Arabidopsis Information Resource (TAIR) maintains a databa
biology data for the model higher plant Arabidopsis thaliana . Data
the complete genome sequence along with gene structure, gene p
metabolism, gene expression, DNA and seed stocks, genome ma
markers, publications, and information about the Arabidopsis rese
function data is updated every two weeks from the latest publishe
community data submissions. Gene structures are updated 1-2 tin
computational and manual methods as well as community submis
genes. TAIR also provides extensive linkouts from our data pages
resources.
The Arabidopsis Biological Resource Center at The Ohio State Ur
preserves and distributes seed and DNA resources of Arabidopsis
Stock information and ordering for the ABRC are fully integrated ir
CARNEGIE INSTITUTION FOR SCIENCE
TAIR is located at the Carnegie Institution for Scienc
Biology and funded by the National Science Foundation with additional
support from TAIR sponsors.
Updates on TAIR funding are available here.
ftp://ftp.arabidopsis.org/home/tair/Sequences/
FTP ディレクトリ /home/tair/Sequences/ / ft
エクスプローラーでこの FTP サイトを表示するには、Alt キーを押して、
FTP サイトを開く] をクリックしてください。
1 階層上のディレクトリへ
11/09/2010 12:00午前 ディレクトリ ATH cDNA EST sequences FASTA
08/23/2011 12:00午前 ディレクトリ Polymorphism datasets
11/18/2010 12:00午前 ディレクトリ blast datasets
08/10/2006 12:00午前 ディレクトリ chloroplast
08/10/2006 12:00午前 ディレクトリ clones
08/10/2006 12:00午前 ディレクトリ markers
08/10/2006 12:00午前 ディレクトリ mitochondrial
02/22/2011 12:00午前 ディレクトリ whole chromosomes

ゲノムデータ取得


ftp://ftp.arabidopsis.org/home/tair/Sequences/
FTP ディレクトリ /home...
FTP ディレクトリ /home/tair/Sequences/whole_chromosomes/
ftp.arabidopsis.org
エクスプローラーでこの FTP サイトを表示するには、Alt キーを押し
FTP サイトを開く] をクリックしてください。
1 階層上のディレクトリへ
08/10/2006 12:00午前 ディレクトリ GFF TIGR OLD
02/22/2011 12:00午前 ディレクトリ OLD
02/22/2011 12:00午前 811 README whole_chromosomes.txt
02/22/2011 12:00午前 30,812,908 TAIR10_chr1.fas
02/22/2011 12:00午前 19,947,711 TAIR10_chr2.fas
02/22/2011 12:00午前 23,756,866 TAIR10_chr3.fas
02/22/2011 12:00午前 18,820,386 TAIR10_chr4.fas
02/22/2011 12:00午前 27,317,041 TAIR10_chr5.fas
02/22/2011 12:00午前 156,517 TAIR10_chrC.fas
02/22/2011 12:00午前 371,653 TAIR10_chrM.fas
04/23/2008 12:00午前 140,800 TAIR8_Assembly_updates.xls
06/20/2009 12:00午前 123,904 TAIR9_assembly_updates.xls
06/20/2009 12:00午前 123,904 TAIR9_assembly_updates_relative
06/22/2009 12:00午前 10,422 tair9_Assembly_gaps.gff


arab.fasta - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
>Chr1 CHROMOSOME dumped from ADB: Jun/20/09 14:53; last updated: 2009-02-02
CCCTAAACCCTAAACCCTAAACCCTAAACCTCTGAATCCTTAATCCCTAAATCCCTAAAT
CTTTAAATCCTACATCCATGAATCCCTAAATACCTAATTCCCTAAACCCGAAACCGGTTT
CTCTGGTTGAAAATCATTGTGTATATAATGATAATTTTATCGTTTTTATGTAATTGCTTA
...
>Chr2 CHROMOSOME dumped from ADB: Jun/20/09 14:54; last updated: 2009-02-02
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
...
>Chr3 CHROMOSOME dumped from ADB: Jun/20/09 14:54; last updated: 2009-02-02
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNCCCTAAACCCTAAACCCTAA
ACCCTAAACCCTAAACCCTAAACCCTAAACCCTAAATCCATAAATCCCTAAAACCATAAT
...
>Chr4 CHROMOSOME dumped from ADB: Jun/20/09 14:54; last updated: 2009-02-02
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
...
>Chr5 CHROMOSOME dumped from ADB: Jun/20/09 14:54; last updated: 2009-02-02
TATACCATGTACCCTCAACCTTAAAACCCTAAAACCTATACTATAAATCTTTAAAACCTA
TACTCTAAACCATAGGGTTTGTGAGTTTGCATAAAGTGTCACGTATAAGTGTTTCTAACA
TGTGAGTTTGCATAAGAGTCTCGACTATGTGTTTGTTCAAAAGTGACGTAAGTGTTTAGA
...
1 行、1 列

Rで配列長とGC含量計算

	Length	GC contents
chr1	28.76MB	35.80%
chr2	19.60MB	35.80%
chr3	23.17MB	35.40%
chr4	17.40MB	36.02%
chr5	25.95MB	34.50%

arab.fasta - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
>Chr1 CHROMOSOME dumped from ADB: Jun/20/09 14:53; last updated: 2009-02-02
CCCTAAACCCTAAACCCTAAACCCTAAACCTCTGAATCCTTAATCCCTAAATCCCTAAAT
CTTTAAATCCTACATCCATGAATCCCTAAATACCTAATTCCCTAAACCCGAAACCGGTTT
CTCTGGTTGAAAATCATTCTCTATATAATCATAATTTTATCGTTTTTATCTAATTCGTTA
...
>Chr2 CHROMO
NNNNNNNNNNNN
NNNNNNNNNNNN
NNNNNNNNNNNN
...
>Chr3 CHROMO
NNNNNNNNNNNN
NNNNNNNNNNNN
ACCCTAAACCCT
...
>Chr4 CHROMO
NNNNNNNNNNNN
NNNNNNNNNNNN
NNNNNNNNNNNN
...
>Chr5 CHROMO
TATACCATGTAC
TACTCTAAACCA
TGTGAGTTTGCA
...
```

R Console

```
> in_f <- "arab.fasta"                                    #multi-fa$
> out_f <- "result_arab.txt"                              #出力ファ$
>
> #必要なパッケージなどをロード
> library(Biostrings)                                     #パッケージ$
>
> #染色体ごとのファイルを読み込み
> reads <- readDNAStringSet(in_f,
>
> #GC含量(GC content)計算のところ
> count <- alphabetFrequency(read
> CG <- rowSums(count[,2:3])
> ACGT <- rowSums(count[,1:4])
> out <- CG/ACGT*100
>
> #出力用に結果をまとめている
> tmp <- cbind(names(reads), CG,
> colnames(tmp) <- c("description
> write.table(tmp, out_f, sep="\t
>
```

result_arab.txt - Microsoft Excel

ファイル ホーム 挿入 ページ レイアウト 数式 データ 校閲 表示

H10

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	Chr1 CHROMOSOME	10856525	30263312	30427671	35.87
3	Chr2 CHROMOSOME	7063739	19695728	19698289	35.86
4	Chr3 CHROMOSOME	8521037	23453853	23459830	36.33
5	Chr4 CHROMOSOME	6727440	18582024	18585056	36.20
6	Chr5 CHROMOSOME	9691012	26965224	26975502	35.94

result_arab

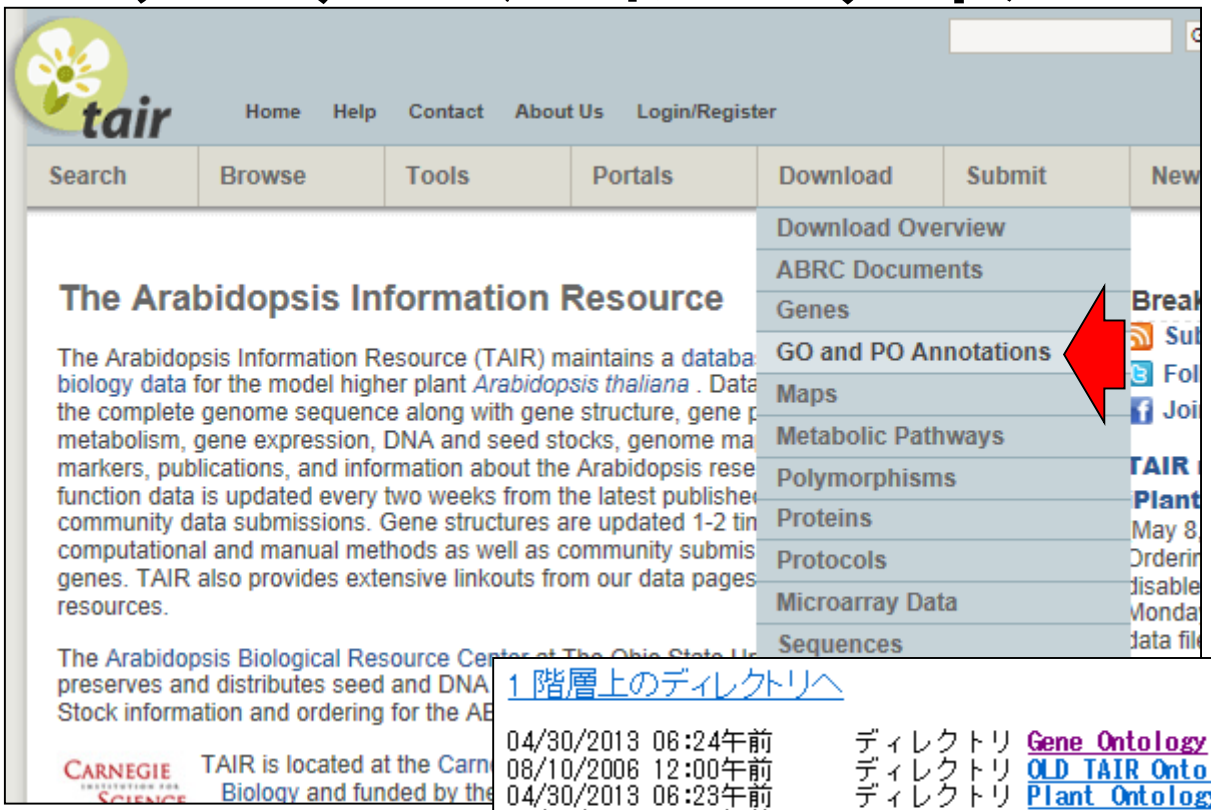
コマンド

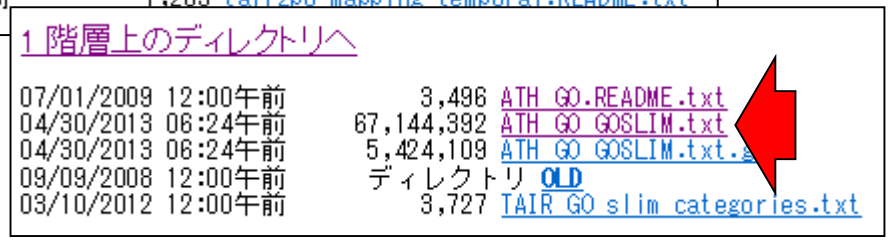
最新のゲノム配列を読み込んで
GC含量を計算することができます

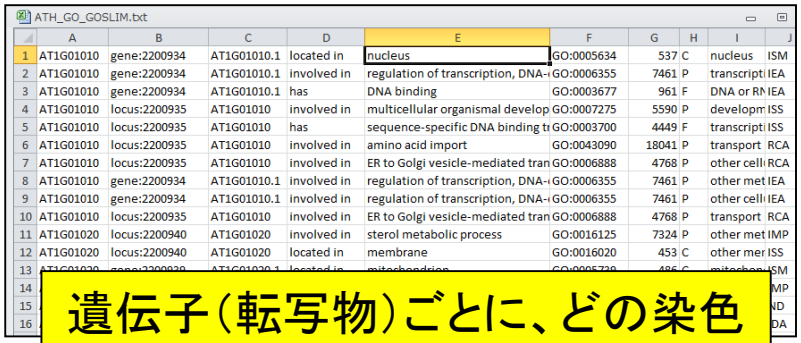
Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - コンティグごとのGC含量計算。一定の長さ以上のものを抽出とか。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

アノテーションファイル?!






ATH_GO_GOSLIM.txt

	A	B	C	D	E	F	G	H	I	J
1	AT1G01010	gene:2200934	AT1G01010.1	located in	nucleus	GO:0005634	537	C	nucleus	ISM
2	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-(	GO:0006355	7461	P	transcripti	IEA
3	AT1G01010	gene:2200934	AT1G01010.1	has	DNA binding	GO:0003677	961	F	DNA or RN	IEA
4	AT1G01010	locus:2200935	AT1G01010	involved in	multicellular organismal develop	GO:0007275	5590	P	developm	ISS
5	AT1G01010	locus:2200935	AT1G01010	has	sequence-specific DNA binding t	GO:0003700	4449	F	transcripti	ISS
6	AT1G01010	locus:2200935	AT1G01010	involved in	amino acid import	GO:0043090	18041	P	transport	RCA
7	AT1G01010	locus:2200935	AT1G01010	involved in	ER to Golgi vesicle-mediated tran	GO:0006888	4768	P	other cell	RCA
8	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-(	GO:0006355	7461	P	other met	IEA
9	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-(	GO:0006355	7461	P	other cell	IEA
10	AT1G01010	locus:2200935	AT1G01010	involved in	ER to Golgi vesicle-mediated tran	GO:0006888	4768	P	transport	RCA
11	AT1G01020	locus:2200940	AT1G01020	involved in	sterol metabolic process	GO:0016125	7324	P	other met	IMP
12	AT1G01020	locus:2200940	AT1G01020	located in	membrane	GO:0016020	453	C	other mer	ISS
13										
14										MP
15										ND
16										DA

遺伝子（転写物）ごとに、どの染色体上に存在するのか、どんなGene Ontology IDが割り当てられているのか、などの情報を含むファイル

アノテーションファイルからの情報抽出

ATH_GO_GOSLIM.txt

	A	B	C	D	E	F	G	H	I	J
1	AT1G01010	gene:2200934	AT1G01010.1	located in	nucleus	GO:0005634	537	C	nucleus	ISM
2	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-	GO:0006355	7461	P	transcripti	IEA
3	AT1G01010	gene:2200934	AT1G01010.1	has	DNA binding	GO:0003677	961	F	DNA or RN	IEA
4	AT1G01010	locus:2200935	AT1G01010	involved in	multicellular organismal develop	GO:0007275	5590	P	developm	ISS
5	AT1G01010	locus:2200935	AT1G01010	has	sequence-specific DNA binding tr	GO:0003700	4449	F	transcripti	ISS
6	AT1G01010	locus:2200935	AT1G01010	involved in	amino acid import	GO:0043090	18041	P	transport	RCA
7	AT1G01010	locus:2200935	AT1G01010	involved in	ER to Golgi vesicle-mediated tran	GO:0006888	4768	P	other cell	RCA
8	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-	GO:0006355	7461	P	other met	IEA
9	AT1G01010	gene:2200934	AT1G01010.1	involved in	regulation of transcription, DNA-	GO:0006355	7461	P	other cell	IEA
10	AT1G01010	locus:2200935	AT1G01010	involved in	ER to Golgi vesicle-mediated tran	GO:0006888	4768	P	transport	RCA
11	AT1G01020	locus:2200940	AT1G01020	involved in	sterol metabolic process	GO:0016125	7324	P	other met	IMP
12	AT1G01020	locus:2200940	AT1G01020	located in	membrane	GO:0016020	453	C	other mer	ISS
13	AT1G01020	gene:2200939	AT1G01020.1	located in	mitochondrion	GO:0005739	486	C	mitochon	ISM
14	AT1G01020	locus:2200940	AT1G01020	involved in	sterol metabolic process	GO:0016125	7324	P	other met	IMP
15	AT1G01020	locus:2200940	AT1G			GO:0003674	3226	F	unknown	ND
16	AT1G01020	locus:2200940	AT1G			GO:0005783	268	C	ER	IDA

「核」に存在する遺伝子のみからなるリストを得たいときにもRが利用可能

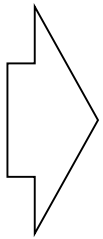
アノテーションファイルからの情報抽出

genelist1.txt

	A
1	gene1
2	gene7
3	gene9

annotation.txt

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic



hoge1.txt

hoge1.txt

	A	B	C	D
1	gene1	hoge01	plasma_mem	nuclear
2	gene7	hoge07	tebasaki	nuclear
3	gene9	hoge09	nihonshu	nuclear

目的：アノテーションファイル（annotation.txt）中の第1列目に対して、リストファイル（genelist1.txt）中の文字列と一致する行を抜き出して、hoge1.txtというファイル名で出力したい

http://www.iu.a.u-tokyo.ac.jp/~kadota/r_seq.html

Plant Global Education ...

(Rで)塩基配列解析 (...

(Rで)塩基配列解析(主に次世代シーケンサーのデータ) (last modified 2013/05/24, since 2010)

What's new?
RパッケージTCCの最新版は1.1.99です。6月6日に開催されるNAIST植物グローバル教育プロジェクト・平成25年度ワークショップでは、R(ver. 3.0.1)とTCC(ver. 1.1.99)をベースに話をする予定です。Rのバージョン自体は2.15.3でもたぶん大丈夫だとは思いますが。。。Rのインストールと起動を実行したあとにTCC(ver. 1.1.99)のインストールも忘れずに行っておいてください。Windowsのヒト用の詳細なインストール方法はこちらです。また、当日はhoge.zipという圧縮ファイルを解凍して得られる"hoge"というフォルダがデスクトップ上にあるという前提でセミナーを行いますのでこの圧縮ファイルもダウンロードと解凍をやっておいてください。(2013/05/24)NEW

・平成25年6月27日、7月3, 4日にこのウェブページ関連の実習を含む講義(農学生命情報科学特論)を行います。東大生以外の外部の方も受講可能です。詳しくは事務局までお問い合わせください。(2013/05/23)NEW
・R3.0.1がリリースされていたのでこれに変更しました。(2013/05/17)NEW

- はじめに (last modified 2012/03/29)
- Rのインストールと起動 (last modified 2013/05/17) NEW
- サンプルデータ (last modified 2013/01/23)
- イントロダクション | NGS | 各種覚書 (last modified 2010/12/10)
- イントロダクション | NGS | 様々なプラットフォーム (last modified 2011/07/15)
- イントロダクション | NGS | リファレンス配列取得(マップされる側) (last modified 2011/02/03)
- イントロダクション | NGS | リファレンス配列取得後の各種情報抽出(特にRefSeq) (last modified 2011/03/20)
- イントロダクション | NGS | リファレンス配列取得後の各種情報抽出2(readFASTA関数の利用) (last modified 2011/04/07)
- イントロダクション | NGS | アノテーション情報取得(refFlatファイル) (last modified 2010/12/07)
- イントロダクション | NGS | アノテーション情報取得(BioMart and
- イントロダクション | 一般 | 遺伝子の転写開始点近傍配列(上流
- イントロダクション | 一般 | 任意のキーワードを含む行を抽出(
- イントロダクション | 一般 | ランダムな塩基配列を作成 (last mo
- イントロダクション | 一般 | 任意の長さの可能な全ての塩基配列を作
- イントロダクション | 一般 | 配列取得 (last modified 2010/7/7)
- イントロダクション | 一般 | 指定した範囲の配列を取得 (last modified
- イントロダクション | 一般 | 翻訳配列(translate)を取得 (last modified
- イントロダクション | 一般 | 相補鎖(complement)を取得 (last modified
- イントロダクション | 一般 | 逆相補鎖(reverse complement)を取得 (la

・ イントロダクション | 一般 | 任意のキーワードを含む行を抽出

例えばタブ区切りテキストファイルのannotation.txtが手元にあり、この中からgenelist1.txtのようなリストファ
り方を示します。
Linux(UNIX)のgrepコマンドのようなものです。perlのハッシュのようなものです。

「ファイル」ー「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. 目的のタブ区切りテキストファイル(annotation.txt)中の第1列目をキーとして、リストファイル(genelist1.txt)中

```
------    ここから    ------
in_f1 <- "annotation.txt"                                    #入力ファイル名(目的のタブ区切りテキスト
in_f2 <- "genelist1.txt"                                     #入力ファイル名(キーワードなどのリストフ
out_f <- "hoge1.txt"                                         #出力ファイル名を指定
param <- 1                                                   #in_f1で読み込む目的のファイルの何列目の

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")   #入力ファイル(目的のファイル)を読み込んで
keywords <- readLines(in_f2)                                 #入力ファイル(リストファイル)を読み込んで
dim(data)                                                    #オブジェクトdataの行数と列数を表示

#本番
obj <- is.element(as.character(data[,param]), keywords)      #in_f1で読み込んだファイル中の(param)列目
out <- data[obj,]                                            #行列dataからobjがTRUEとなる行のみを抽出し
dim(out)                                                     #オブジェクトoutの行数と列数を表示
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)#outの中身をout_fで指定したファイル名で保

------    ここまで    ------
```

```
1. 目的のタブ区切りテキストファイル(annotation.txt)中の第1列目をキーとして、リストファイル(genelist1.txt)中のものが含まれる行全体を出力したい場合：
------    ここから    ------
in_f1 <- "annotation.txt"                                   #入力ファイル名(目的のタブ区切りテキストファイル)を指定
in_f2 <- "genelist1.txt"                                    #入力ファイル名(キーワードなどのリストファイル)を指定
out_f <- "hoge1.txt"                                        #出力ファイル名を指定
param <- 1                                                  #in_f1で読み込む目的のファイルの何列目のデータに対してサーチしたいかを指定

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")  #入力ファイル(目的のファイル)を読み込んでdataに格納
keywords <- readLines(in_f2)                                #入力ファイル(リストファイル)を読み込んでkeywordsに格納
dim(data)                                                   #オブジェクトdataの行数と列数を表示

#本番
obj <- is.element(as.character(data[,param]), keywords)     #in_f1で読み込んだファイル中の(param)列目の文字列ベクトル中の各要素がベクトルkeywo
out <- data[obj,]                                           #行列dataからobjがTRUEとなる行のみを抽出した結果をoutに格納
dim(out)                                                    #オブジェクトoutの行数と列数を表示
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)#outの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

annotation.txt

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

genelist1.txt

	A
1	gene1
2	gene7
3	gene9

hoge1.txt

hoge1.txt

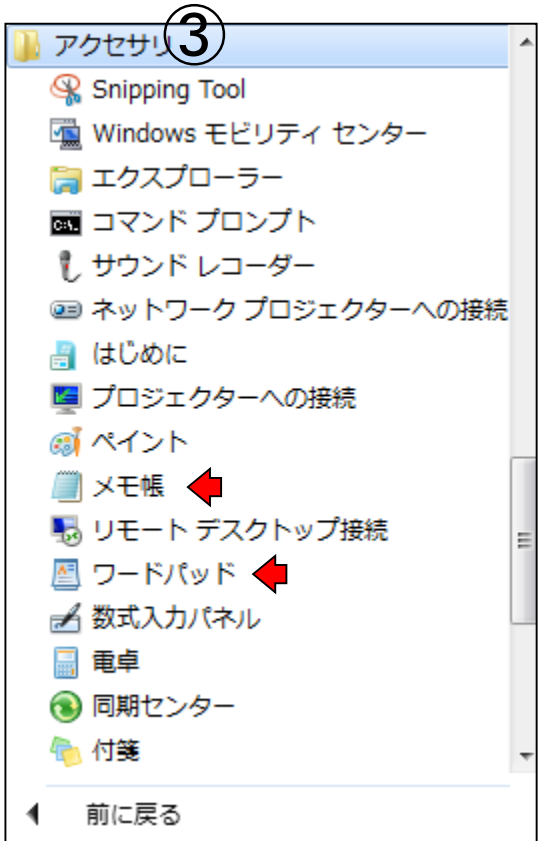
	A	B	C	D
1	gene1	hoge01	plasma_mem	nuclear
2	gene7	hoge07	tebasaki	nuclear
3	gene9	hoge09	nihonshu	nuclear

デスクトップ上に「hoge」という名前のフォルダがあり、フォルダ中にannotation.txtとgenelist1.txtが存在する、という前提です。

「**メモ帳**」で開くと改行コードが崩れている場合は、「**ワードパッド**」などで開くとよい

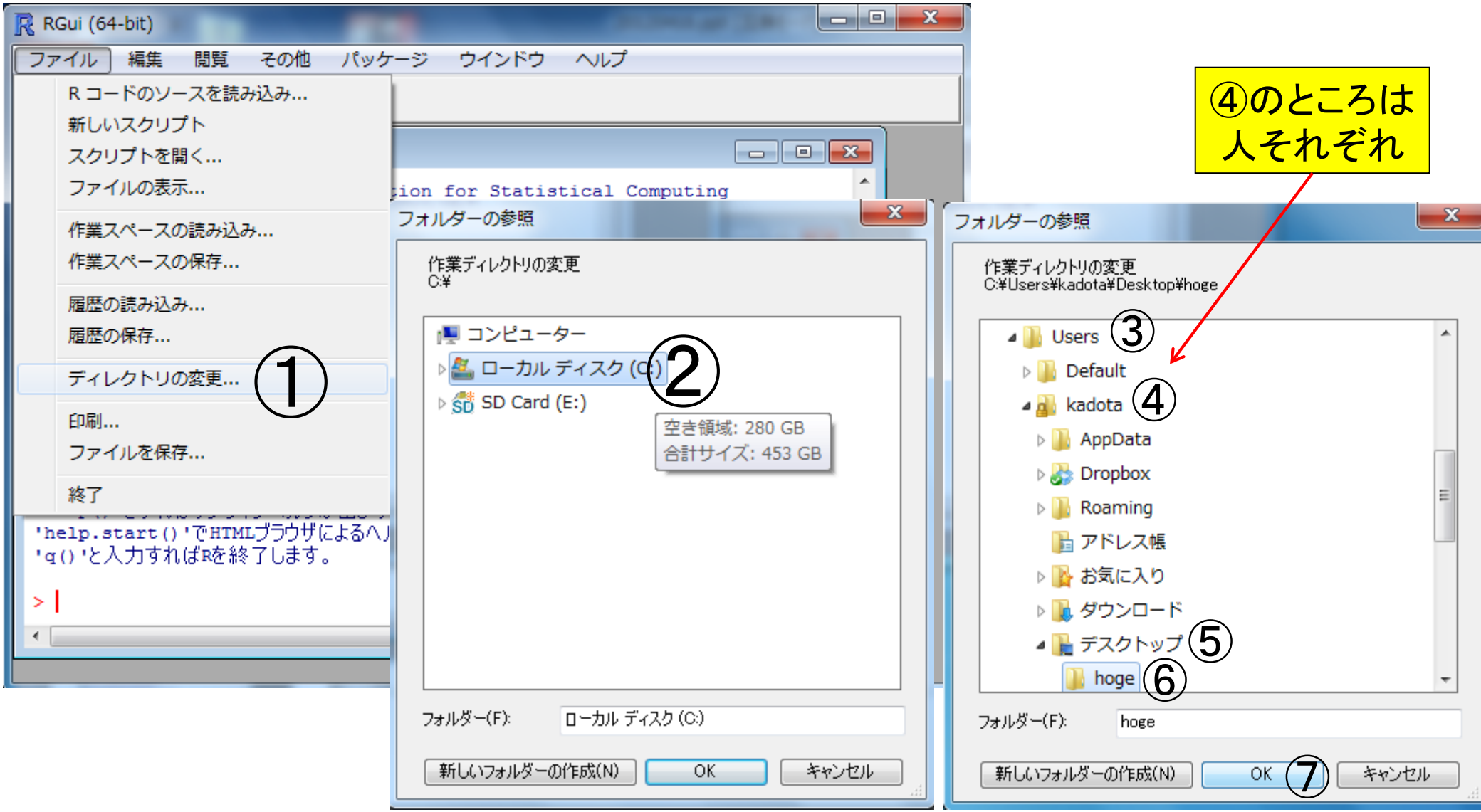
各種ソフトの場所

①

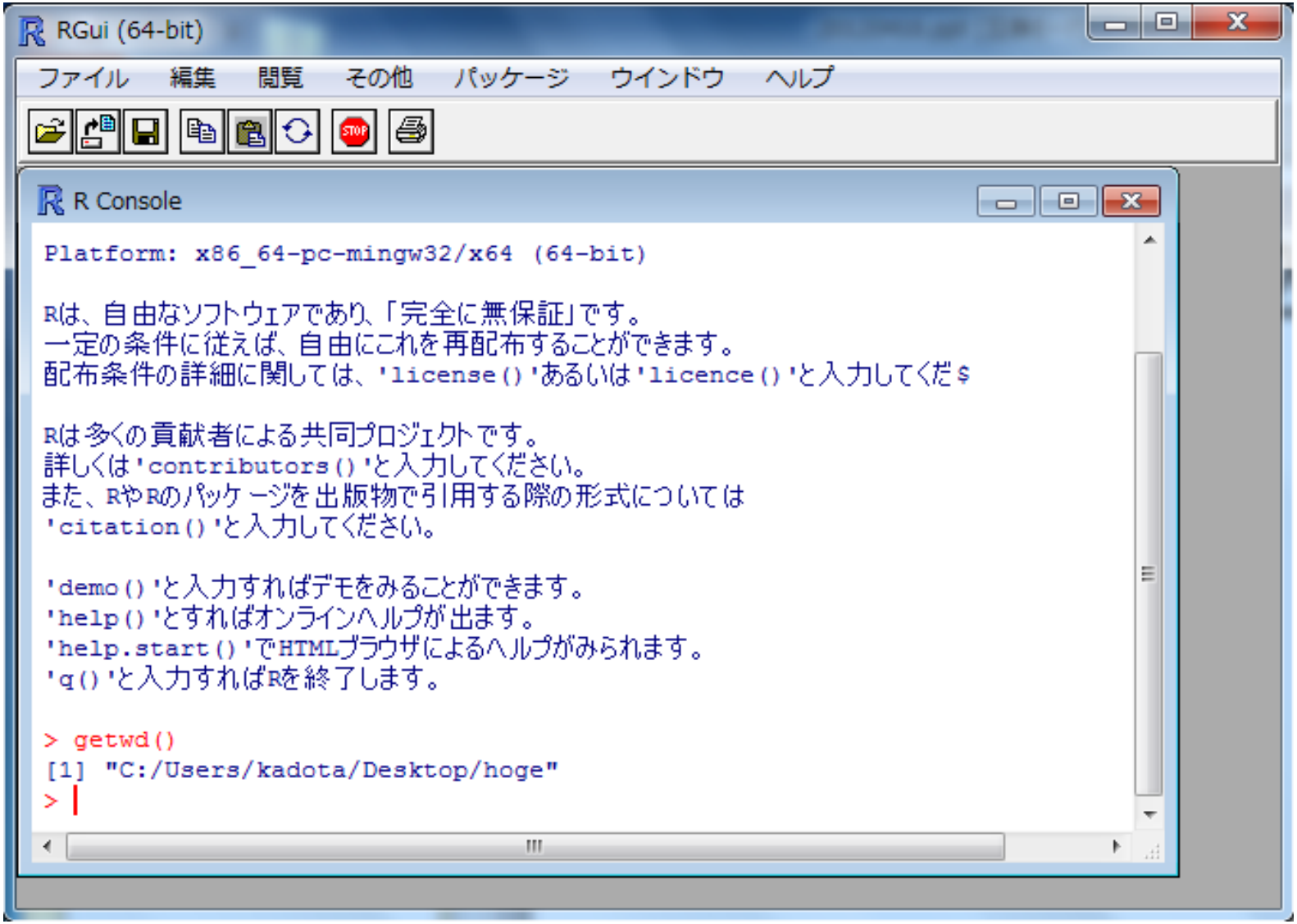


スタート ― すべてのプログラム ― …

Rの起動および作業ディレクトリの変更



「getwd()」と打ち込んで確認



基本はコピペ

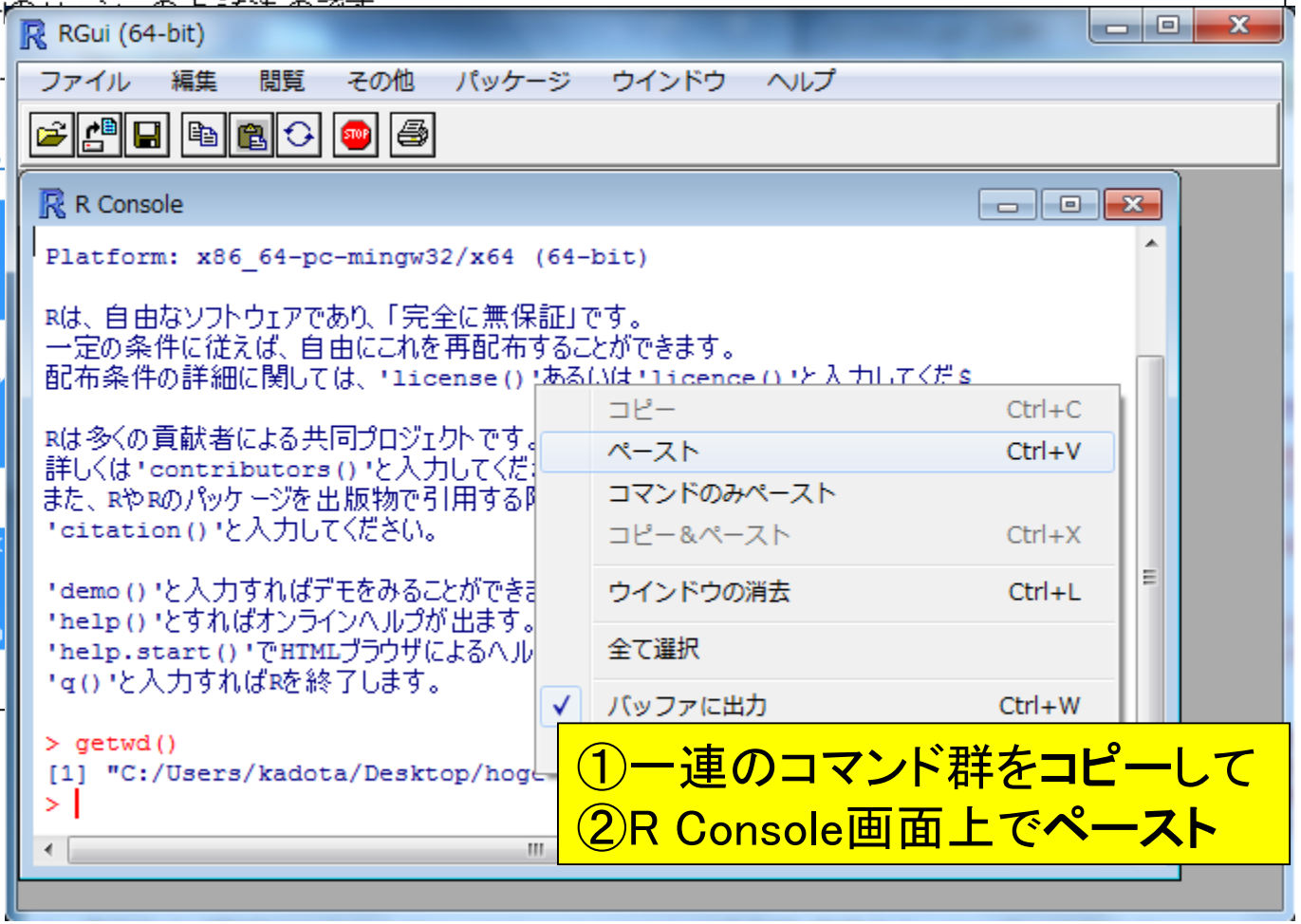
- イントロダクション | 一般 | 任意のキーワードを含む行を抽出

例えばタブ区切りテキストファイルのannotation.txtが手元にあり、この中からgenelist1.txtのようなリストファイル中の文字列を含む行を抽出するやり
します。
Linux (UNIX)のgrepコマンドのようなものです。per

「ファイル」－「ディレクトリの変更」で解析したいフ

1. 目的のタブ区切りテキストファイル(annotation.

```
------      ここから      ------
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t"
keywords <- readLines(in_f2)
dim(data)

#本番
obj <- is.element(as.character(data[,param]), k
out <- data[obj,]
dim(out)
write.table(out, out_f, sep="¥t", append=F, quo

------      ここまで      ------
```



実行結果

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

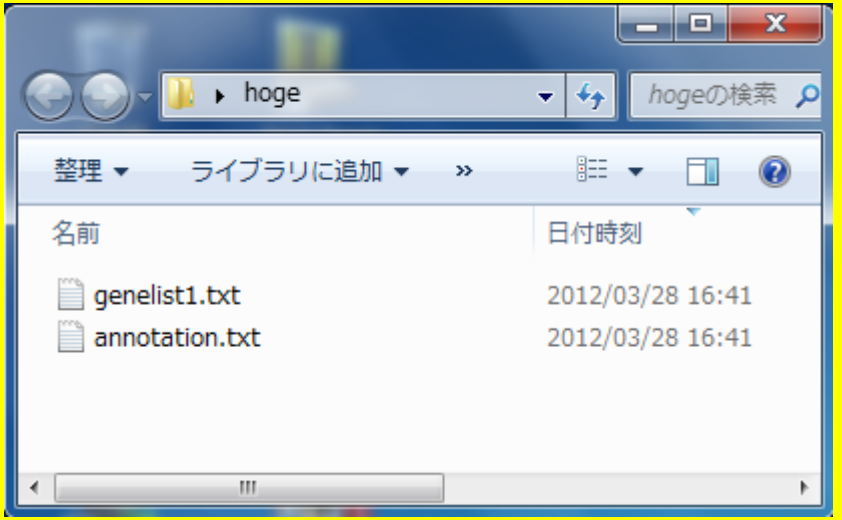
R Console

```
> in_f2 <- "genelist1.txt"
> out_f <- "hoge1.txt"
> param <- 1
>
> #ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", quote="")
> keywords <- readLines(in_f2)
> dim(data)
[1] 11  4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)
> out <- data[obj,]
> dim(out)
[1] 3 4
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F)
>
>
```

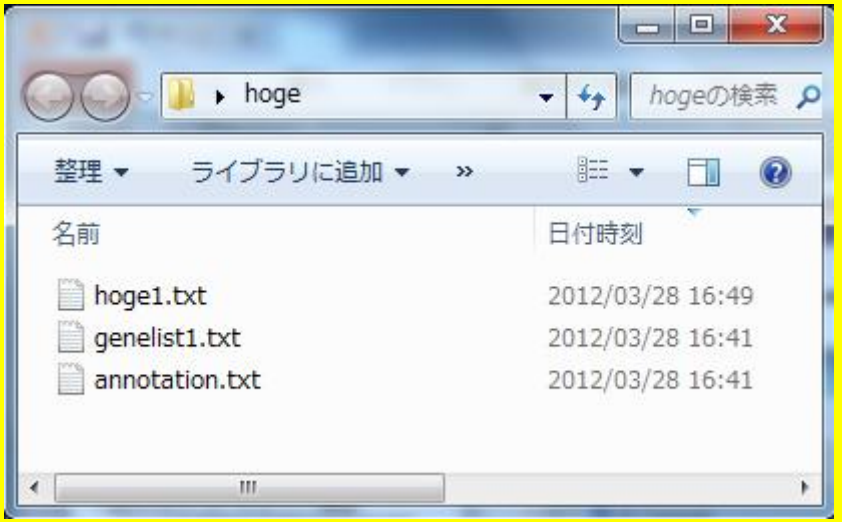
hoge1.txt

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene7	hoge07	tebasaki	nuclear
4	gene9	hoge09	nihonshu	nuclear

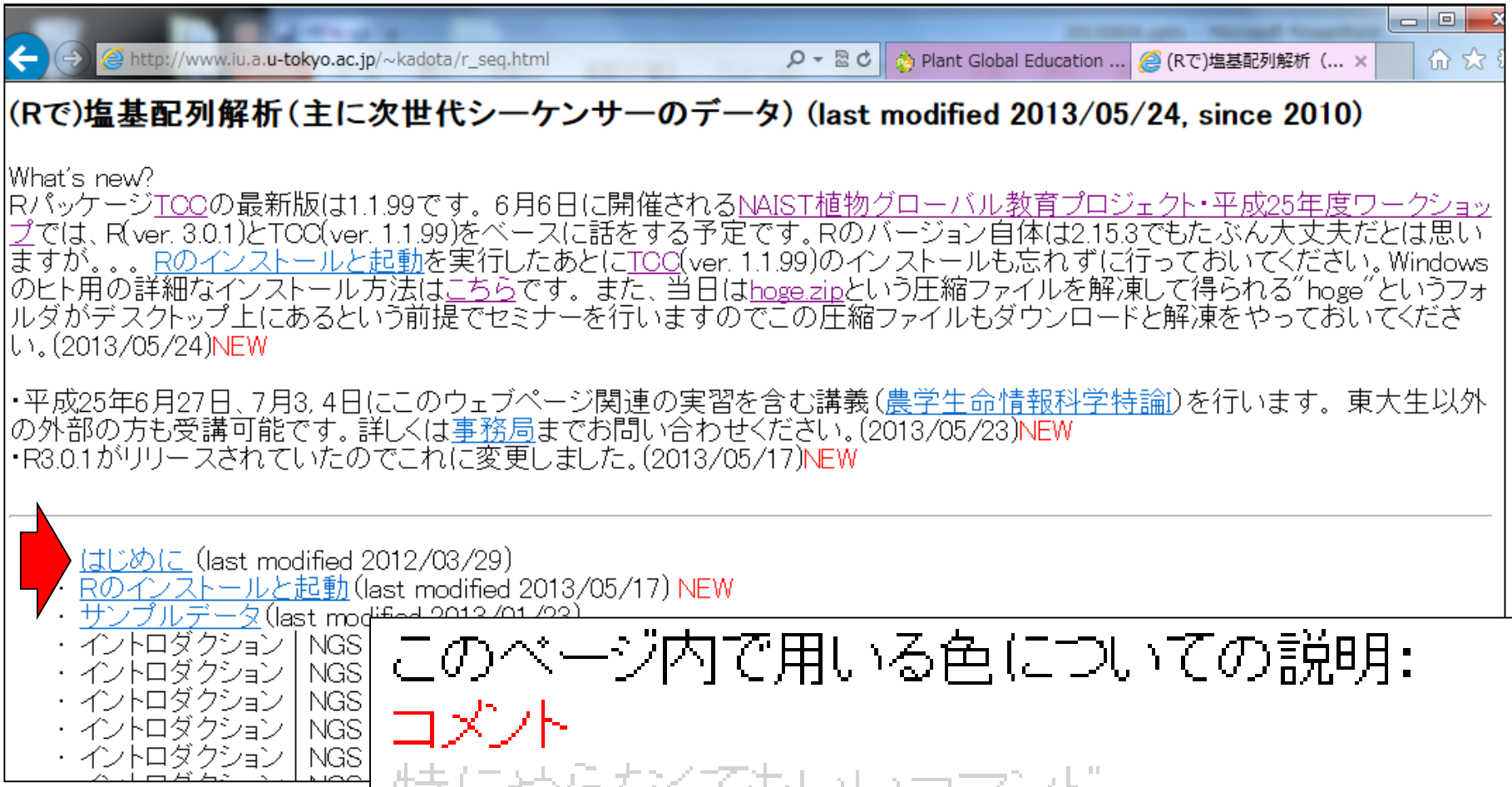
実行前のhogeフォルダ



実行後のhogeフォルダ



色についての説明


http://www.iu.a.u-tokyo.ac.jp/~kadota/r_seq.html
Plant Global Education ...
(Rで)塩基配列解析 (...
(Rで)塩基配列解析(主に次世代シーケンサーのデータ) (last modified 2013/05/24, since 2010)
What's new?
RパッケージTCCの最新版は1.1.99です。6月6日に開催されるNAIST植物グローバル教育プロジェクト・平成25年度ワークショップでは、R(ver. 3.0.1)とTCC(ver. 1.1.99)をベースに話をする予定です。Rのバージョン自体は2.15.3でもたぶん大丈夫だとは思いますが。。。Rのインストールと起動を実行したあとにTCC(ver. 1.1.99)のインストールも忘れずに行っておいてください。Windowsのヒト用の詳細なインストール方法はこちらです。また、当日はhoge.zipという圧縮ファイルを解凍して得られる"hoge"というフォルダがデスクトップ上にあるという前提でセミナーを行いますのでこの圧縮ファイルもダウンロードと解凍をやっておいてください。(2013/05/24)NEW
・平成25年6月27日、7月3, 4日にこのウェブページ関連の実習を含む講義(農学生命情報科学特論I)を行います。東大生以外の外部の方も受講可能です。詳しくは事務局までお問い合わせください。(2013/05/23)NEW
・R3.0.1がリリースされていたのでこれに変更しました。(2013/05/17)NEW
はじめに (last modified 2012/03/29)
Rのインストールと起動 (last modified 2013/05/17) NEW
サンプルデータ (last mod
イントロダクション NGS
イントロダクション NGS
イントロダクション NGS
イントロダクション NGS
イントロダクション NGS
このページ内で用いる色についての説明:
コメント
特にやらなくてもいいコマンド
プログラム実行時に目的に応じて変更すべき箇所

色についての説明

このページ内で用いる色についての説明:
コメント
特にやらなくてもいいコマンド
プログラム実行時に目的に応じて変更すべき箇所

```
------    ここから    ------
in_f1 <- "annotation.txt"                          #入力ファイル名(目的のタブ区切りテキストファイル
in_f2 <- "genelist1.txt"                           #入力ファイル名(キーワードなどのリストファイル)
out_f <- "hoge1.txt"                               #出力ファイル名を指定
param <- 1                                         #in_f1で読み込む目的のファイルの何列目のデータに

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")   #入力ファイル(目的のファイル)を読み込んでdataに
keywords <- readLines(in_f2)                       #入力ファイル(リストファイル)を読み込んでkeywor
dim(data)                                          #オブジェクトdataの行数と列数を表示

#本番
obj <- is.element(as.character(data[,param]), keywords)   #in_f1で読み込んだファイル中の(param)列目の文字
out <- data[obj,]                                  #行列dataからobjがTRUEとなる行のみを抽出した結果
dim(ou
write.                                                                              保存。

------
```

上記は1列目でキーワード検索する場合

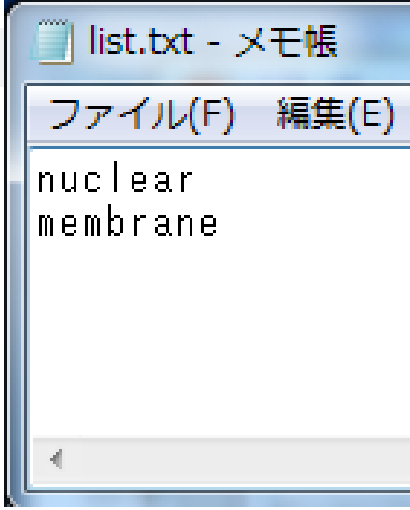
	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

4列目でキーワード検索したいときは?

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

解答例

1. 目的のキーワードリストを含むファイルを作成し、例：list.txt
2. 該当箇所を変更し、Rコンソール画面上でコピペ



run1.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表

```
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1

#ファイルの読み込み
data <- read.table(in_f1,
keywords <- readLines(in_f
dim(data)

#本番
obj <- is.element(as.chara
out <- data[obj,]
dim(out)
write.table(out, out_f, se
```

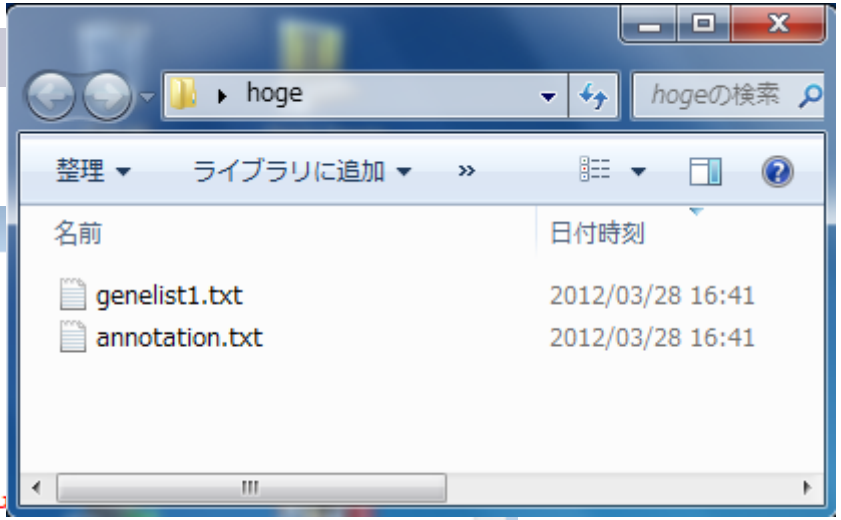
run1.txt - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
in_f1 <- "annotation.txt"
in_f2 <- "list.txt"
out_f <- "hoge1.txt"
param <- 4

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")
keywords <- readLines(in_f2)
dim(data)

#本番
obj <- is.element(as.character(data[,param]), keywords)
out <- data[obj,]
dim(out)
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=
```

一連の作業手順を記述した「スクリプト」を一つのファイルとして保存することをお勧め。（list.txtファイル作成時に、”membrane”と打った後に改行を入れた場合と入れない場合を試してみよう）

ありがちなミス1

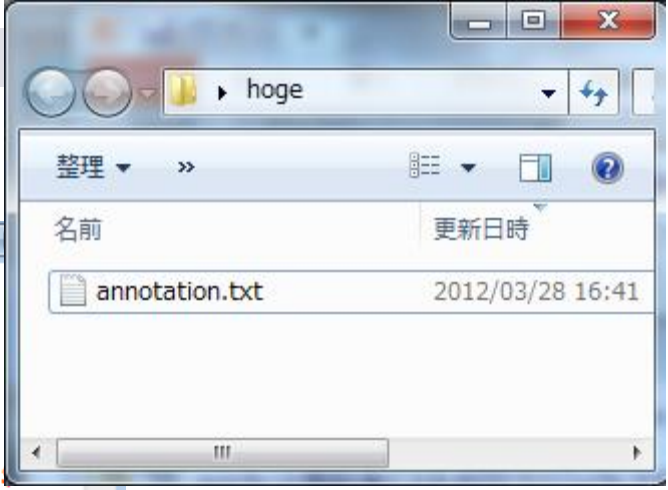


R Console

```
> in_f1 <- "annotation.txt"
> in_f2 <- "genelist1.txt"
> out_f <- "hoge1.txt"
> param <- 1
>
> #ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", qu
 以下にエラー file(file, "rt") :  コネクションを開くことができません
 追加情報:    警告メッセージ:
In file(file, "rt") :
   ファイル 'annotation.txt' を開くことができません: No such file or director$
> keywords <- readLines(in_f2)                                    #入力ファ$
 以下にエラー file(con, "r") :  コネクションを開くことができません
 追加情報:    警告メッセージ:
In file(con, "r") :
   ファイル 'genelist1.txt' を開くことができません: No such file or directory
> dim(data)                                                       #オブジェ$
NULL
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)         #in_f1で読$
 以下にエラー data[, param] :
   'closure' 型のオブジェクトは部分代入可能ではありません
> out <- data[obj,]                                               #行列data$
 エラー:  オブジェクト 'obj' がありません
> dim(out)                                                        #オブジェ$
 エラー:  オブジェクト 'out' がありません
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F)#outの中身$
 以下にエラー is.data.frame(x) :  オブジェクト 'out' がありません
>
> getwd()
[1] "C:/Users/kadota/Documents"
```

作業ディレクトリの変更を忘れている…

ありがちなミス2

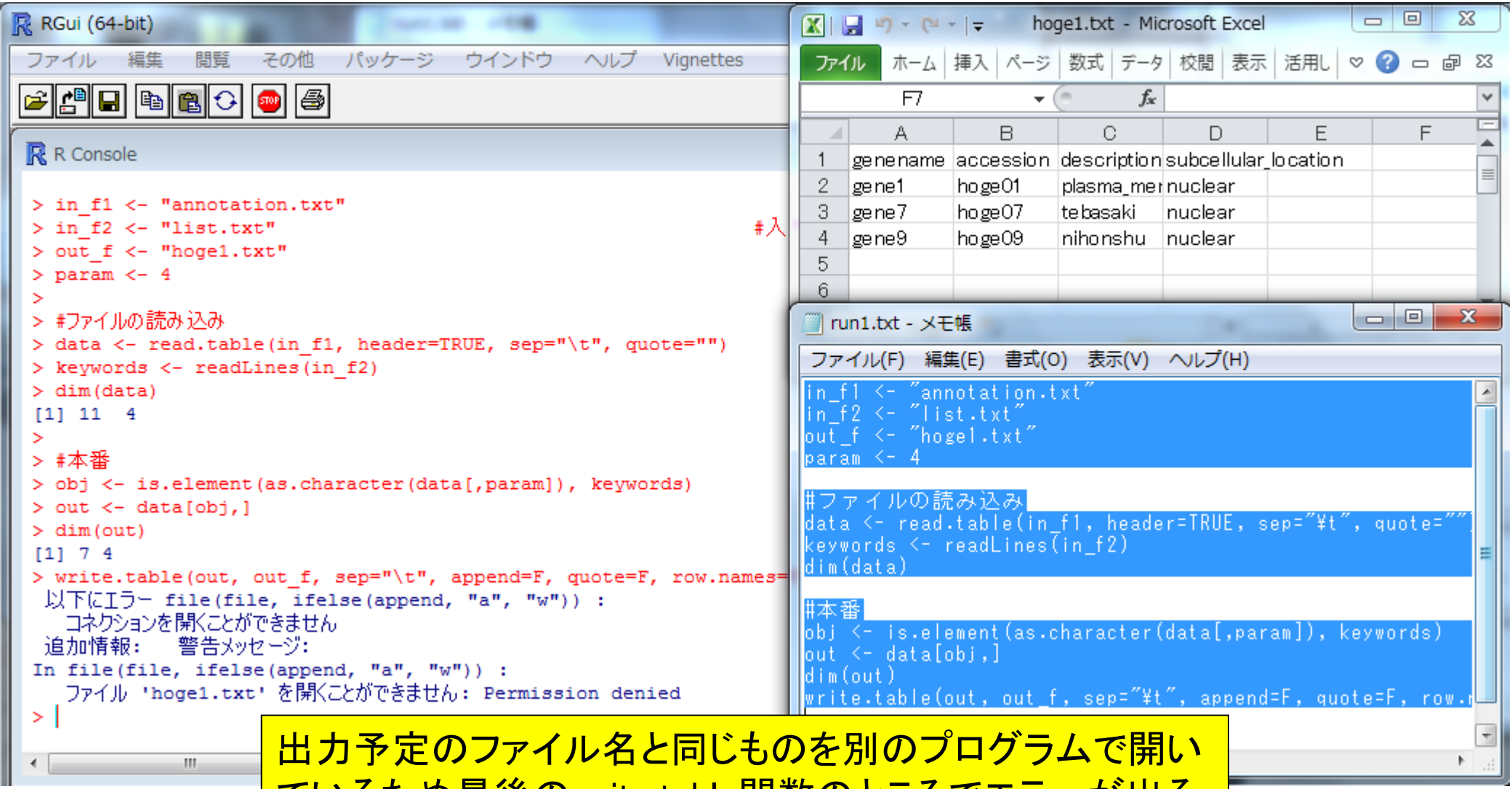


R Console

```
> getwd()
[1] "C:/Users/kadota/Desktop/hoge"
> in_f1 <- "annotation.txt"                                    #入力ファイル$
> in_f2 <- "genelist1.txt"                                     #入力ファイル$
> out_f <- "hoge1.txt"                                         #出力ファイル$
> param <- 1                                                   #in_f1で読み
>
> #ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", quote="")   #入力ファイル$
> keywords <- readLines(in_f2)                                 #入力ファイル$
 以下にエラー file(con, "r") :  コネクションを開くことができません
 追加情報:    警告メッセージ:
In file(con, "r") :
   ファイル 'genelist1.txt' を開くことができません: No such file or directory
> dim(data)                                                    #オブジェクト$
[1] 11  4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)      #in_f1で読み$
 以下にエラー match(el, set, 0L) :  オブジェクト 'keywords' がありません
> out <- data[obj,]                                            #行列dataから$
 以下にエラー `[.data.frame`(data, obj, ) :  オブジェクト 'obj' がありません
> dim(out)                                                     #オブジェクト$
 エラー:  オブジェクト 'out' がありません
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F)#outの中身をo$
 以下にエラー is.data.frame(x) :  オブジェクト 'out' がありません
>
```

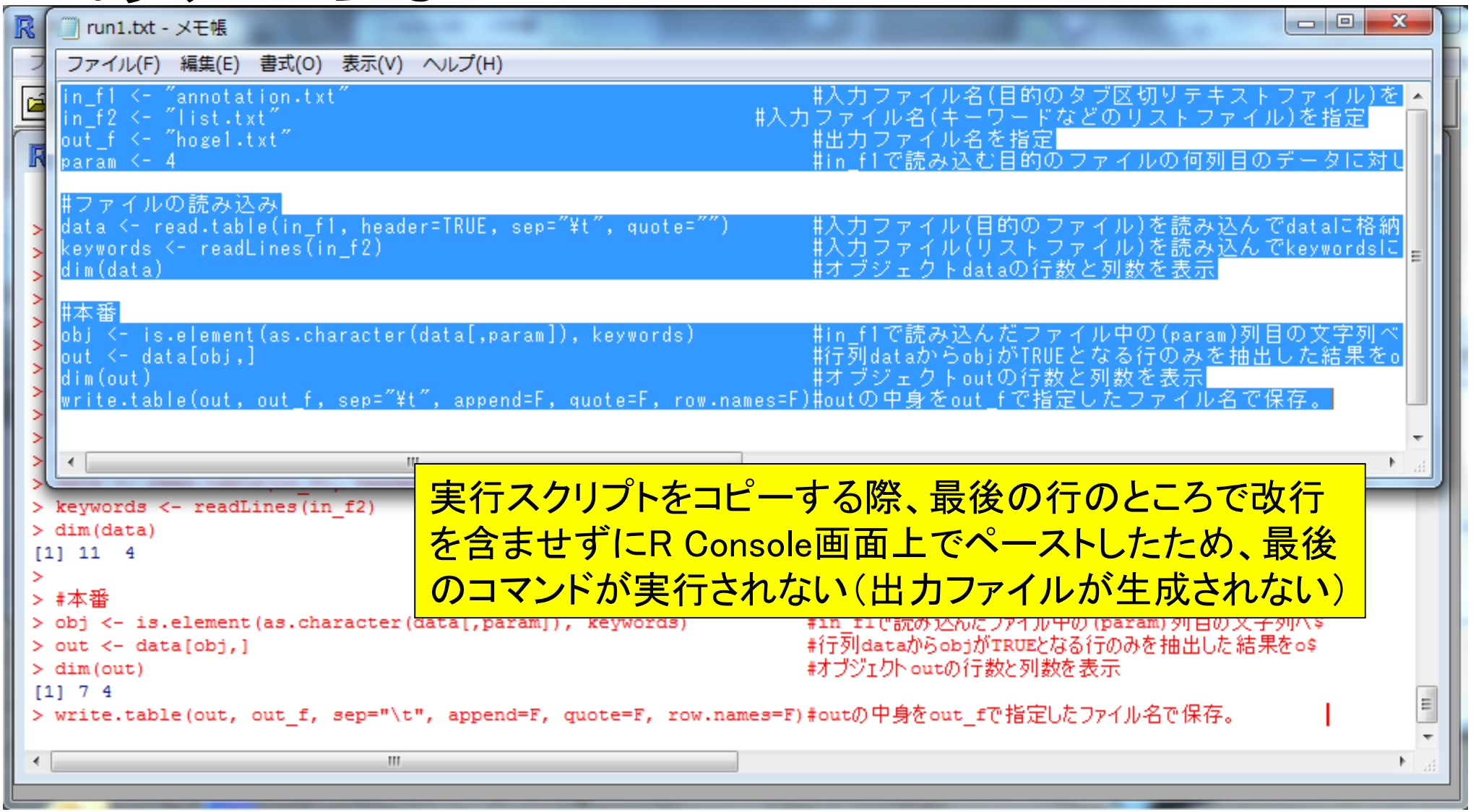
必要な入力ファイルが作業ディレクトリ中に存在しない…

ありがちなミス3



出力予定のファイル名と同じものを別のプログラムで開いているため最後のwrite.table関数のところでエラーが出る

ありがちなミス4



実行スクリプトをコピーする際、最後の行のところで改行を含ませずにR Console画面上でペーストしたため、最後のコマンドが実行されない（出力ファイルが生成されない）

• イントロダクション | 一般 | 任意のキーワードを含む行を抽出

例えばタブ区切りテキストファイルのannotation.txtが手元にあり、この中からgenelist1.txtのようなリストファイル中の文字列を含
を示します。
Linux(UNIX)のgrepコマンドのようなものです。perlのハッシュのようなものです。

「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. 目的のタブ区切りテキストファイル(annotation.txt)中の第1列目をキーとして、リストファイル(genelist1.txt)中のものが含ま

```
------    ここから    ------
in_f1 <- "annotation.txt"                                    #入力ファイル名(目的のタブ区切りテキストファイル)を指定
in_f2 <- "genelist1.txt"                                     #入力ファイル名(キーワードなどのリストファイル)を指定
out_f <- "hoge1.txt"                                         #出力ファイル名を指定
param <- 1                                                   #in_f1で読み込む目的のファイルの何列目のデータに対して

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")   #入力ファイル(目的のファイル)を読み込んでdataに格納
keywords <- readLines(in_f2)                                 #入力ファイル(リストファイル)を読み込んでkeywordsに格納
dim(data)                                                    #オブジェクトdataの行数と列数を表示

#本番
obj <- is.element(as.character(data[,param]), keywords)      #in_f1で読み込んだファイル中の(param)列目の文字列ベクト
out <- data[obj,]                                            #行列dataからobjがTRUEとなる行のみを抽出した結果をoutに
dim(out)                                                     #オブジェクトoutの行数と列数を表示
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)#outの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

「------ ここまで ------」の一つ上の空行には「スクリプト最終行のコマンドを確実に実行するため」という深い意味があります

読み込み

annotation.txt 参考

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

```
------    ここから    ------
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1

#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")
keywords <- readLines(in_f2)
dim(data)

#本番
obj <- is.element(as.character
out <- data[obj,]
dim(out)
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)

------    ここまで    ------
```

①in_f1で指定したファイルを読み込め
②読み込むファイルの最初の行はヘッダー部分です
③ファイルの区切り文字は「タブ」です

行列data

annotation.txt 参考

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

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

```
>
> #ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", qu
>
> data
   genename accession description subcellular_location
1     gene1     hoge01  plasma_mem              nuclear
2     gene2     hoge02      hohinu             membrane
3     gene3     hoge03     agribio           endoplasmic
4     gene4     hoge04     genesis           endoplasmic
5     gene5     hoge05        kamo             membrane
6     gene6     hoge06     netteba               humei
7     gene7     hoge07    tebasaki              nuclear
8     gene8     hoge08       biiru              nuclear
9     gene9     hoge09    nihonshu              nuclear
10   gene10     hoge10      agene1             membrane
11   gene11     hoge11      iyaaaa           endoplasmic
>
```

入力ファイルの中身を正しく
読み込めていることがわかる

```
------    ここから    ------
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1

#ファイルの読み込み
data <- read.table(i
keywords <- readLine
dim(data)

#本番
obj <- is.element(as
out <- data[obj,]
dim(out)
write.table(out, out

------    ここまで
```

annotation.txt　参考

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

RGui (64-bit)

ファイル　編集　閲覧

R Console

```
> data
   genename accessio
1     gene1    hoge01  plasma_mem          nuclear
2     gene2    hoge02      hohinu         membrane
3     gene3    hoge03     agribio       endoplasmic
4     gene4    hoge04
5     gene5    hoge05
6     gene6    hoge06
7     gene7    hoge07
8     gene8    hoge08
9     gene9    hoge09    nihonshu          nuclear
10   gene10    hoge10      agene1         membrane
11   gene11    hoge11      iyaaaa       endoplasmic
> dim(data)
[1] 11  4
>
```

オブジェクトdataの行数と列数は11と4。
webpage中の表記が灰色なのは、「特にやらなくてもいいコマンド」だから。

行列の要素へのアクセス

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

```
10    gene10     hoge10       agene1              memb
11    gene11     hoge11       iyaaaa           endopla
> dim(data)
[1] 11  4
> data[6,4]
[1] humei
Levels: endoplasmic humei membrane nuclear
> data[2,]
  genename accession description subcellular_location
2    gene2    hoge02      hohinu              membrane
> data[,2]
 [1] hoge01 hoge02 hoge03 hoge04 hoge05 hoge06 hoge07
 [8] hoge08 hoge09 hoge10 hoge11
11 Levels: hoge01 hoge02 hoge03 hoge04 hoge05 ... hoge11
> data[,param]
 [1] gene1  gene2  gene3  gene4  gene5  gene6  gene7
 [8] gene8  gene9  gene10 gene11
11 Levels: gene1 gene10 gene11 gene2 gene3 ... gene9
>
```

data[行，列]

annotation.txt

	A	B	C	D
1	genename	accession	description	subcellular_location
2	gene1	hoge01	plasma_mem	nuclear
3	gene2	hoge02	hohinu	membrane
4	gene3	hoge03	agribio	endoplasmic
5	gene4	hoge04	genesis	endoplasmic
6	gene5	hoge05	kamo	membrane
7	gene6	hoge06	netteba	humei
8	gene7	hoge07	tebasaki	nuclear
9	gene8	hoge08	biiru	nuclear
10	gene9	hoge09	nihonshu	nuclear
11	gene10	hoge10	agene1	membrane
12	gene11	hoge11	iyaaaa	endoplasmic

```
------    ここから    ---
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1
```

paramには1という数値が代入されていたから

やりたかったことをおさらい

参考
genelist1.txt

	A
1	gene1
2	gene7
3	gene9

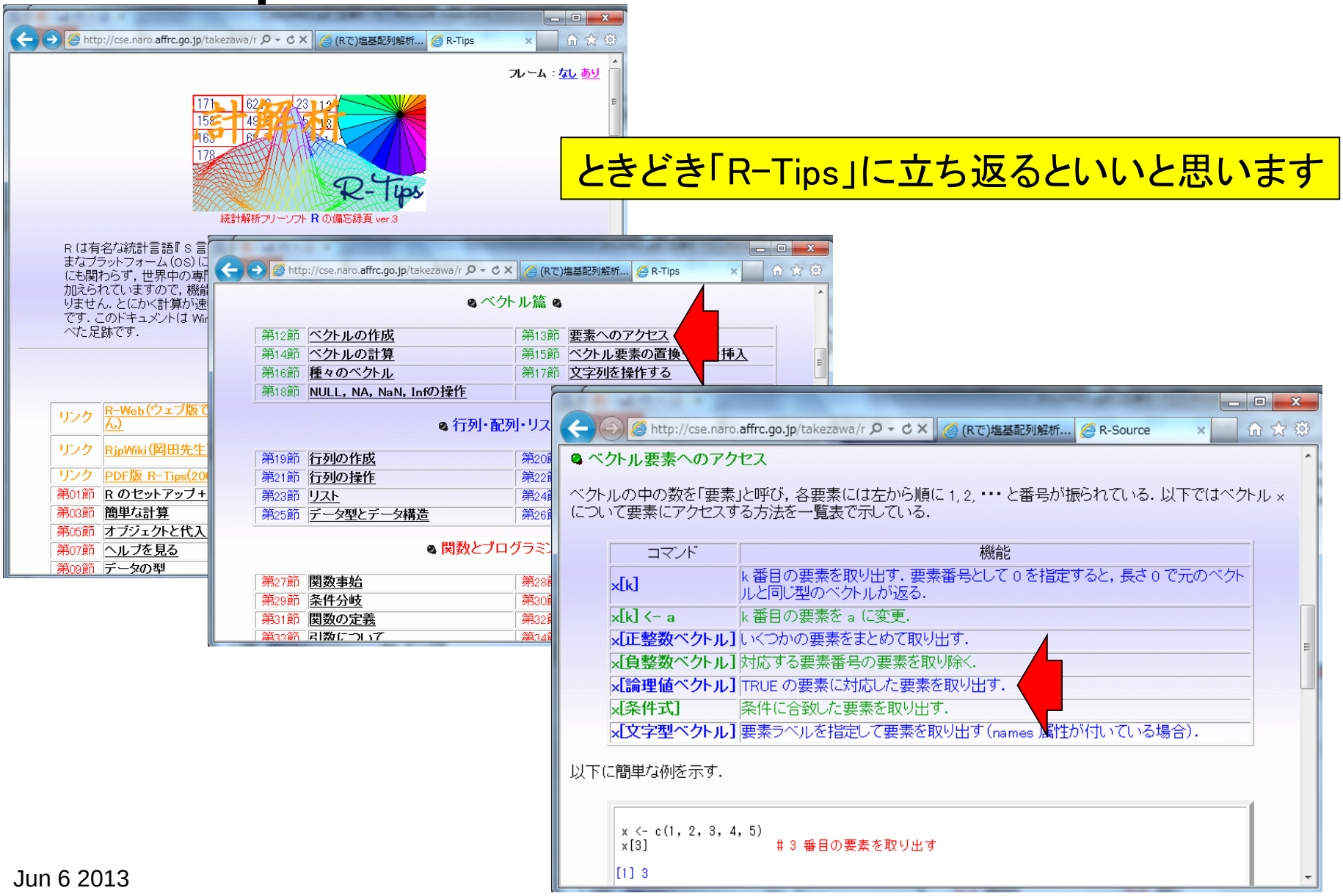
```
------    ここから    ------
in_f1 <- "annotation.txt"
in_f2 <- "genelist1
out_f <- "hoge1.txt
param <- 1

#ファイルの読み込み
data <- read.table(
keywords <- readLin
dim(data)

#本番
obj <- is.element(a
out <- data[obj,]
dim(out)
write.table(out, ou

------    ここまで
```

R Console

```
> obj
 [1]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
> data
   genename accession description subcellular_location
1     gene1    hoge01  plasma_mem               nuclear
2     gene2    hoge02      hohinu              membrane
3     gene3    hoge03     agribio           endoplasmic
4     gene4    hoge04     genesis           endoplasmic
5     gene5    hoge05        kamo              membrane
6     gene6    hoge06     netteba                 humei
7     gene7    hoge07    tebasaki               nuclear
8     gene8    hoge08      hijiru               nuclear
9     gene9    hoge09    nihonshu               nuclear
10   gene10    hoge10
11   gene11    hoge11
> obj
 [1]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
> data[obj,]
  genename accession description subcellular_location
1    gene1    hoge01  plasma_mem               nuclear
7    gene7    hoge07    tebasaki               nuclear
9    gene9    hoge09    nihonshu               nuclear
>
```

論理値ベクトルobjを用いてTRUEの要素に対応する行を抽出している

R-Tipsでお勉強



ときどき「R-Tips」に立ち返るといいと思います

論理値ベクトルを理解

genelist1.txt

	A
1	gene1
2	gene7
3	gene9

```
------    ここから    ------
in_f1 <- "annotation.txt"
in_f2 <- "genelist1.txt"
out_f <- "hoge1.txt"
param <- 1

#ファイルの読み込み
data <- read.table(in_f1, hea
keywords <- readLines(in_f2)
dim(data)

#本番
obj <- is.element(as.character(data[,param]), keywords)
out <- data[obj,]
dim(out)
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)

------    ここまで    ------
```

R Console

```
>
> keywords                          #keywordsの中身を表示
[1] "gene1" "gene7" "gene9"
> length(keywords)                  #keywordsベクトルの要素数を表示
[1] 3
> keywords[3]                       #3番目の要素を表示
[1] "gene9"
> keywords[4]                       #4番目の要素は...ない
[1] NA
> keywords == "gene7"               #"gene7"という文字の位置情報を表示
[1] FALSE  TRUE FALSE
> obj <- keywords == "gene7"        #上記結果をobjに格納
> keywords[obj]                     #objがTRUEとなる要素のみを表示
[1] "gene7"
>
```

論理値ベクトルを理解

genelist1.txt

	A
1	gene1
2	gene7
3	gene9

R Console

```
> hoge <- c("gene7", "gene9")#二つの要素からなるベクトルhogeを作成
> hoge                       #hogeの中身を表示させてるだけ
[1] "gene7" "gene9"
> keywords == hoge           #FALSE TRUE TRUEになることを期待したが...
[1] FALSE FALSE FALSE
 警告メッセージ:
In keywords == hoge :
   長いオブジェクトの長さが短いオブジェクトの長さの倍数になっていません
> is.element(keywords, hoge) #keywords中の各要素は集合hogeに含まれるか否か
[1] FALSE  TRUE  TRUE
>
```

疑問に思ったら、自分の理解できるところからボトムアップ式に試す

```
#本番
obj <- is.element(as.character(data[,param]), keywords)
out <- data[obj,]
dim(out)
write.table(out, out_f, sep="¥t", append=F, quote=F, row.names=F)

------    ここまで    ------
```

Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - **multi-fastaファイルからの情報抽出**
    - 一定の長さ以上のものを抽出。コンティグごとのGC含量計算。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

multi-fastaファイルからの情報抽出

Rでmulti-fastaファイルを読み込んで自在に解析できます

まずはコピペ

・ イントロダクション | NGS | アセンブル後のmulti-fastaファイルからN50などの基本情報を

アセンブルプログラムを実行して得られるmulti-fastaファイルを読み込んで、Total lengthやave
のやり方を示します。ここでは以下の二つを例題として行います：
1.「イントロダクション | 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
2. 130MB程度のRefSeqのhuman mRNAのmulti-fastaファイル(h_rna.fasta)
「ファイル」-「ディレクトリの変更」でファイルを保存したいディレクトリに移動 以下をコピペ

1. hoge4.faファイルの場合：

------ ここから ------

```
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージなどをロード
library(Biostrings)

reads <- readDNAStringSet(in_f, format="fasta")

Total_length <- sum(width(reads))
Number_of_contigs <- length(reads)
Average_length <- mean(width(reads))
Median_length <- median(width(reads))
Max_length <- max(width(reads))
Min_length <- min(width(reads))

#N50計算のところ
sorted_length <- rev(sort(width(reads)))
N50 <- sorted_length[cumsum(sorted_length) >= Total_lengt

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
GC_content <- sum(CG)/sum(ACGT)

#出力用に結果をまとめている
tmp <- NULL
tmp <- rbind(tmp, c("Total length (bp)", Total_length))
tmp <- rbind(tmp, c("Number of contigs", Number_of_contig
tmp <- rbind(tmp, c("Average length", Average_length))
tmp <- rbind(tmp, c("Median length", Median_length))
tmp <- rbind(tmp, c("Max length", Max_length))
tmp <- rbind(tmp, c("Min length", Min_length))
tmp <- rbind(tmp, c("N50", N50))
tmp <- rbind(tmp, c("GC content", GC_content))
```

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
```

RGui (64-bit)

ファイル 編集 閲覧 その他

R Console

```
> #GC含量(GC content)計算のところ
> count <- alphabetFrequency(reads)                    #A,C,G,$
> CG <- rowSums(count[,2:3])                           #C,Gの$
> ACGT <- rowSums(count[,1:4])                         #A,C,G,$
> GC_content <- sum(CG)/sum(ACGT)                      #トータ$
>
> #出力用に結果をまとめている
> tmp <- NULL
> tmp <- rbind(tmp, c("Total length (bp)", Total_length))
> tmp <- rbind(tmp, c("Number of contigs", Number_of_contigs))
> tmp <- rbind(tmp, c("Average length", Average_length))
> tmp <- rbind(tmp, c("Median length", Median_length))
> tmp <- rbind(tmp, c("Max length", Max_length))
> tmp <- rbind(tmp, c("Min length", Min_length))
> tmp <- rbind(tmp, c("N50", N50))
> tmp <- rbind(tmp, c("GC content", GC_content))
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F)#tmpの$
>
>
```

「hoge」フォルダにhoge1.txtが作成されているはず

結果ファイルを眺めて動作確認

ID	Length
contig_1	24
contig_2	103
contig_3	65
contig_4	49

• イントロダクション | NGS | アセンブル後のmulti-fastaファイルからN50などの基本情報を取得

アセンブルプログラムを実行して得られるmulti-fastaファイルを読み込んで、Total lengthやaverage lengthなどの各種情報取得を行うた
のやり方を示します。ここでは以下の二つを例題として行います：
1.「イントロダクション | 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)
2. 130MB程度のRefSeqのhuman mRNAのmulti-fastaファイル(h_rna.fasta)
「ファイル」-「ディレクトリの変更」でファイルを保存したいディレクトリに移動し以下をコピペ。

1. hoge4.faファイルの場合：
------ ここから ------

```
in_f <- "hoge4.fa"                                  #multi-fasta形式の入力ファイルを指定
out_f <- "hoge1.txt"                                #出力ファイル名を指定

#必要なパッケージなどをロード
li                                                  み

re                                                  イルをFASTA形式で読み込み

To                                                  タルの長さ」の情報を取得
Nu                                                  情報を取得
Av                                                  長」の情報を取得
Me                                                  値」の情報を取得
Ma                                                  「最大値」の情報を取得
Mi                                                  「最小値」の情報を取得

#N
so                                                  ートした結果をsorted length
N5                                                  ら足しこんでいってTotal leng

#G
co                                                  配列ごとにカウントした結果を
CG                                                  てCGに格納
AC                                                  算してACGTに格納
GC                                                  情報を取得

#5
tmp <- NULL
tmp <- rbind(tmp, c("Total length (bp)", Total_length))
tmp <- rbind(tmp, c("Number of contigs", Number_of_contigs))
tmp <- rbind(tmp, c("Average length", Average_length))
tmp <- rbind(tmp, c("Median length", Median_length))
tmp <- rbind(tmp, c("Max length", Max_length))
tmp <- rbind(tmp, c("Min length", Min_length))
tmp <- rbind(tmp, c("N50", N50))
tmp <- rbind(tmp, c("GC content", GC_content))
```

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
```

hoge1.txt

ファイル ホー 挿入 ペー 数式

F14

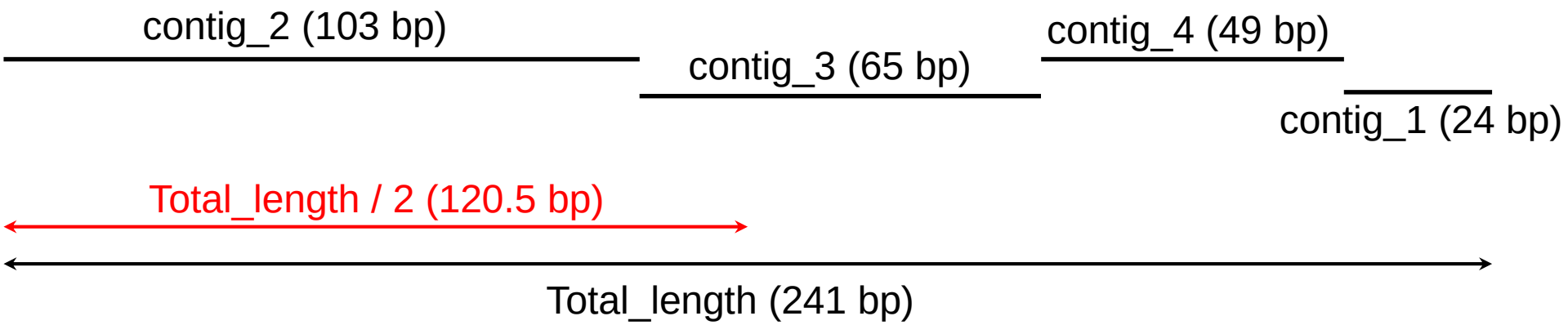
	A	B
1	Total length (bp)	241
2	Number of contigs	4
3	Average length	60.25
4	Median length	57
5	Max length	103
6	Min length	24
7	N50	65
8	GC content	0.577

hoge1 100%

N50

- アセンブルがどれだけうまくいっているかを表す指標の一つ
  - 長いコンティグから足していってTotal_lengthの50%に達したときのコンティグの長さ

ID	Length
contig_1	24
contig_2	103
contig_3	65
contig_4	49



情報抽出手順(の一部)

```
1. hoge4.faファイルの場合:
------    ここから    ------
in_f <- "hoge4.fa"                                  #multi-fasta形式の入力ファイルを指定
out_f <- "hoge1.txt"                                #出力ファイル名を指定

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

reads <- readDNAStringSet(in_f, format="fasta")     #in_fで指定したファイルをFASTA形式で読み込み

Total_length <- sum(width(reads
Number_of_contigs <- length(rea
Average_length <- mean(width(re
Median_length <- median(width(r
Max_length <- max(width(reads))
Min_length <- min(width(reads))
```

R Console

```
> reads
  A DNAStringSet instance of length 4
    width seq                                               names     $
[1]    24 CGGACAGCTCCTCGGCATCCGGAT                          contig_1
[2]   103 GTCTGCCTCAAGCGCCCCAAG...GAGACCCAGCACACCCTGTC     contig_2
[3]    65 TGTAGGAGAAGGGCGGTATCA...TAATTGTATGAGGTCGGGCA     contig_3
[4]    49 CGTGCTGATTCCACACAGCAG...GACCTCTACCTATGAACATG     contig_4
>
> width(reads)
[1]  24 103  65  49
>
> sum(width(reads))
[1] 241
>
```

width関数を使えば配列長情報を取り出せるようだ

情報抽出手順(の一部)

R Console

```
>
> reads[1]
  A DNAStringSet instance of length 1
    width seq                                                names
[1]    24 CGGACAGCTCCTCGGCATCCGGAT                          contig_1
>
> reads[2]
  A DNAStringSet instance of length 1
    width seq                                                names
[1]   103 GTCTGCCTCAAGCGCCCCAAGTGGG...GGCAGAGACCCAGCACACCCTGTC contig_2
>
> width(reads) >= 50
[1] FALSE  TRUE  TRUE FALSE
>
> reads[width(reads) >= 50]
  A DNAStringSet instance of length 2
    width seq                                                names
[1]   103 GTCTGCCTCAAGCGCCCCAAGTGGG...GGCAGAGACCCAGCACACCCTGTC contig_2
[2]    65 TGTAGGAGAAGGGCGGTATCAGCGT...TTACTAATTGTATGAGGTCGGGCA contig_3
>
```

ID	Length
contig_1	24
contig_2	103
contig_3	65
contig_4	49

50 bp以上のコンティグからなる
サブセットの抽出ができそうだ!

参考

・ファイルの読み込み | マップ後 | Bowtie形式 (last modified 2013/01/16)
・ファイルの読み込み | マップ後 | BAM形式 (last modified 2010/7/13)
・ファイルの読み込み | マップ後 | GFF3形式 (last modified 2010/6/9)
・ファイルの読み込み | マップ後 | SOAP形式 (last modified 2011/07/20)
・クオリティチェック | NGS(一般)について (last modified 2012/08/02)
・クオリティチェック | NGS(一般) | qrqc (Quick Read Quality Control) (last modified 2012/02/20)
・フィルタリング | 一般 | fastqファイルからACGTのみからなる配列(重複あり)の抽出 (last modified 2012/07/13)
・フィルタリング | 一般 | fastqファイルからACGTのみからなる配列(重複なし)の抽出 (last modified 2012/07/13)
・フィルタリング | 一般 | multi-fastaファイルのdescription行の記述を整形 (last modified 2012/07/13)
・フィルタリング | 一般 | multi-fastaファイルからACGT以外の文字の出現数でフィルタリング (last modified 2012/07/13)
・フィルタリング | 一般 | multi-fastaファイルから指定した配列長のもののみ抽出 (last modified 2012/04/03)
・フィルタリング | 一般 | multi-fastaファイルから任意のサブセットを抽出 (last modified 2012/07/17)
・フィルタリング | NGS(miRNA) | アダプター配列除去0(基本的なところ) (last modified 2010/5/27)
・フィルタリング

フィルタリング | 一般 | multi-fastaファイルから指定した配列長のもののみ抽出

multi-fasta形式のファイルが手元にあったとして、任意の配列長のもののみ抽出するやり方を示します。
ここでは以下の二つを例題として行います：
1.「イントロダクション | 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)
2. 130MB程度のRefSeqのhuman mRNAのmulti-fastaファイル(h_rna.fasta)

「ファイル」－「ディレクトリの変更」でファイルを保存したいディレクトリに移動し以下をコピペ。

1. hoge4.faファイルの場合：

```
------    ここから    ------
in_f <- "hoge4.fa"                                  #multi-fasta形式の入力ファイルを指定
out_f <- "hoge1.fasta"                              #出力ファイル名を指定
param <- 50                                         #配列長の閾値を指定

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

reads <- readDNAStringSet(in_f, format="fasta")     #in_fで指定したファイルをFASTA形式で読み込み
reads                                               #今現在のreadsオブジェクトを表示
reads <- reads[width(reads) >= param]               #paramで指定した配列長以上のもののみ抽出してrea
reads                                               #今現在のreadsオブジェクトを表示
writeXStringSet(reads, file=out_f, format="fasta")  #out_fで指定したファイル名でreadsというオブジェ

------    ここまで    ------
```

指定した配列長**以下**のものを抽出したいときは「<=」とすればよい

- イントロダクション | NGS | アノテーション情報取得(BioMart and biomaRt)(last modified 2011/08/26)
- イントロダクション | 一般 | 任意のキーワードを含む行を抽出(last modified 2012/03/29) NEW
- イントロダクション | 一般 | ランダムな塩基配列を作成(last modified 2012/04/02) NEW
- イントロダクション | 一般 | 配列取得(last modified 2010/[illegible]7)
- イントロダクション | 一般 | 指定した範囲の配列を取得([illegible] modified 2012/04/03) NEW
- イントロダクション | 一般 | 翻訳配列(translate)を取得(las[illegible] modified 2011/07/27)

イントロダクション | 一般 | 指定した範囲の配列を取得

1.では、12塩基(AGTGACGGTCTT)からなる一つの塩基配列(description行が">kadota")からなるFASTA形式ファイル(sample1.fasta)を入力として、この塩基配列の任意の範囲 (始点が3, 終点が9)の配列を抽出し、得られた部分配列をFASTA形式ファイル(tmp1.fasta)に出力するやり方を示します。
2.では、RefSeqのhuman mRNAのmulti-fasta形式のファイル (h_rna.fasta)が手元にあったとして、任意のRefSeq ID (例:NM_203348.1)の任意の範囲 (例:始点が2, 終点が5)の配列の抽出を行います。
3.では、2.の延長線上で、目的のaccession番号が複数ある場合に対応したものです。[illegible]2列目:start位置, 3列目:end位置」からなるリストファイル (list_sub1.txt)を読み込ませて、目的[illegible]るやり方を示します。
ここでは、得られた部分配列をFASTA形式ファイル(ファイル名:tmp3.fasta)で保存する[illegible]
4.では、3.と同じことを異なるファイル(multi-fastaファイル:hoge4.fa、リストファイル;list[illegible]

「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[illegible]

1. sample1.fastaファイルの場合:

```
------    ここから    ------
in_f <- "sample1.fasta"                    #multi-fasta形式のフ
out_f <- "tmp1.fasta"                      #出力ファイル名を指定
param <- c(3, 9)                           #抽出したい範囲の始点

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

reads <- readDNAStringSet(in_f, format="fasta")   #in_fで指定したファイルをFASTA形式で読み込み
out <- subseq(reads, param[1], param[2])          #paramで指定した始点と終点の範囲の配列を抽出してoutに格納
writeXStringSet(out, file=out_f, format="fasta", width=80)   #outの中身をout_fで指定したファイル名で保存

------    ここまで    ------
```

入力ファイル:sample1.fasta

```
>kadota
AGTGACGGTCTT
```

出力ファイル:tmp1.fasta

```
>kadota
TGACGGT
```

参考

1. sample1.fastaファイルの場合:

```
------    ここから    ------
in_f <- "sample1.fasta"
out_f <- "tmp1.fasta"
param <- c(3, 9)

#必要なパッケージなどをロード
library(Biostrings)

reads <- readDNAStringSet(in_f, format="fasta")
out <- subseq(reads, param[1], param[2])
writeXStringSet(out, file=out_f, format="fasta", width=80)

------    ここまで
```

入力ファイル:sample1.fasta

```
>kadota
AGTGACGGTCTT
```

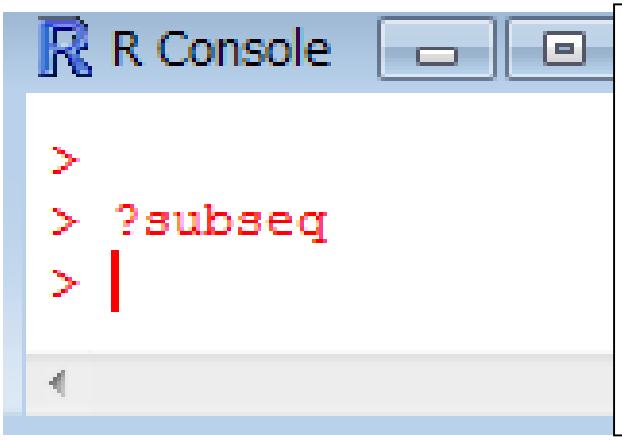
出力ファイル:tmp1.fasta

```
>kadota
TGACGGT
```

```
> reads
  A DNAStringSet instance of length 1
    width seq                                             names
[1]    12 AGTGACGGTCTT                                    kadota
> out
  A DNAStringSet instance of length 1
    width seq                                             names
[1]     7 TGACGGT                                         kadota
> param
[1] 3 9
> param[1]
[1] 3
> param[2]
[1] 9
> subseq(reads, 3, 9)
  A DNAStringSet instance of length 1
    width seq                                             names
[1]     7 TGACGGT                                         kadota
>
```

subseq関数は「塩基配列, start, end」という形式で使うようだ

関数の使用法について



XVector-class {IRanges} R Documentation

XVector objects

Description

The XVector virtual class is a general container for storing an "external vector". It inherits from the Vector, which has a very rich interface.

The following classes derive directly from the XVector class:

...

```
subseq(x4, start=10)
subseq(x4, start=-10)
subseq(x4, start=-20, end=-10)
subseq(x4, start=10, width=5)
subseq(x4, end=10, width=5)
subseq(x4, end=10, width=0)

x3[length(x3):1]
x3[length(x3):1, drop=FALSE]
```

[Package *IRanges* version 1.12.6 Index]

- 「?関数名」で使用法を記したウェブページが開く
- ページの下のほうに、（大抵の場合）使用例が掲載されている
- 使用法既知の関数のマニュアルをいくつか読んで、慣れておく

参考

R Console

```
> reads
  A DNAStringSet instance of length 1
    width seq                                             names
[1]    12 AGTGACGGTCTT                                    kadota
> subseq(reads, start=3, end=9)
  A DNAStringSet instance of length 1
    width seq                                             names
[1]     7 TGACGGT                                         kadota
> subseq(reads, start=3, width=9)
  A DNAStringSet instance of length 1
    width seq                                             names
[1]     9 TGACGGTCT                                       kadota
> subseq(reads, start=3, width=11)
 以下にエラー .Call2("solve_user_SEW", refwidths, start, end
  solving row 1: 'allow.nonnarrowing' is FALSE and the so
> subseq(reads, start=3, width=10)
  A DNAStringSet instance of length 1
    width seq                                             names
[1]    10 TGACGGTCTT                                      kadota
>
```

入力ファイル：sample1.fasta

```
>kadota
AGTGACGGTCTT
```

出力ファイル：tmp1.fasta

```
>kadota
TGACGGT
```

マニュアルの使用例をいくつか試して、ステップアップ

参考

4. hoge4.faファイルで抽出したい情報をリストファイル（list_sub2.txt）で与える場合:

```
------    ここから    ------
in_f1 <- "hoge4.fa"
in_f2 <- "list_sub2.txt"
out_f <- "tmp4.fasta"

library(Bio
reads <- re
list_obj <-

out <- NULL
for(i in 1:
    obj <-
    out <-
}
write.XStri

------
```

#multi-fasta形式のファイルを指定
#リストファイルを指定
#出力ファイル名を指定

で読み込み

レースホル

致する位置を
てobjがTRUE

名で保存

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
```

list_sub2.txt

contig_4	1	10
contig_2	3	8

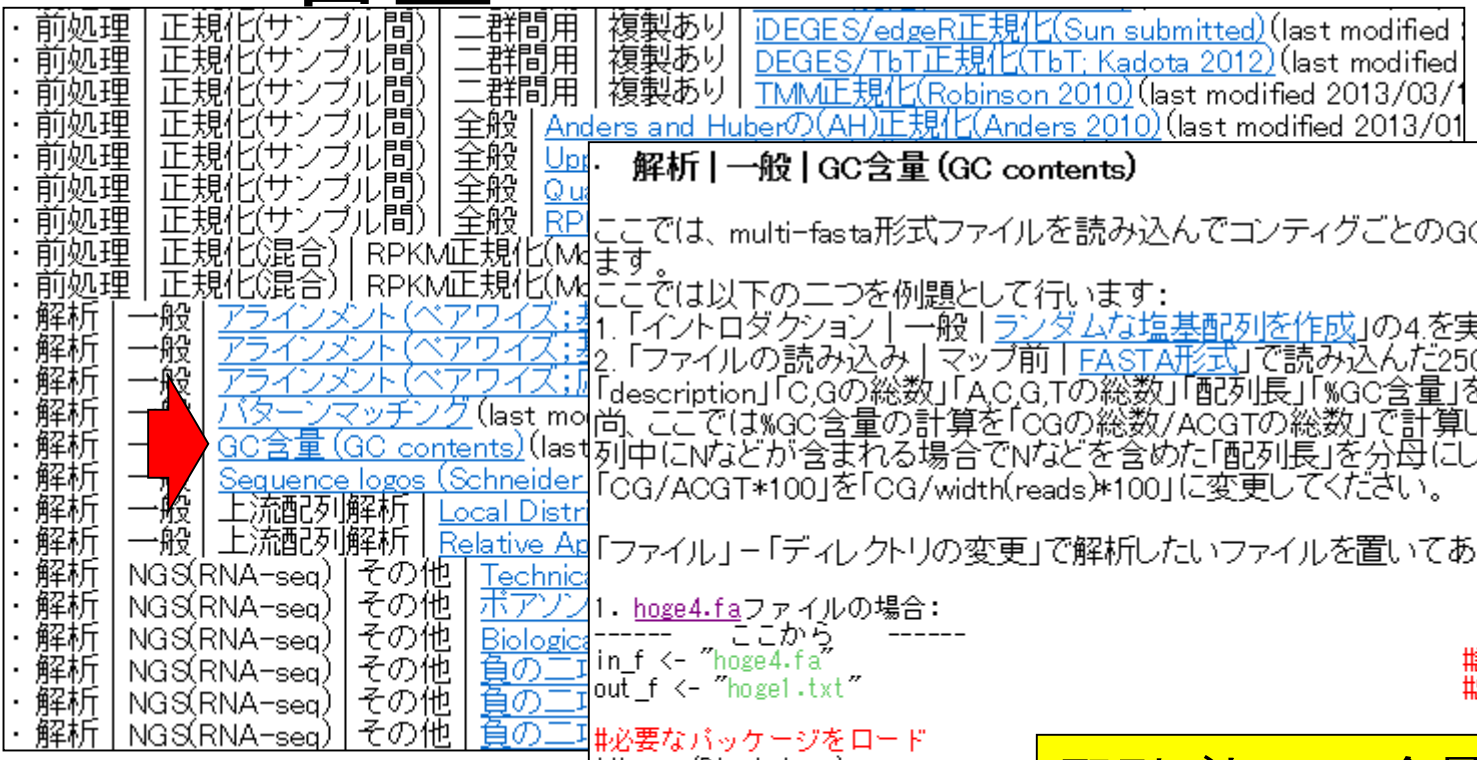
出力ファイル：tmp4.fasta

```
>contig_4
CGTGCTGATT
>contig_2
CTGCCT
```

Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - 一定の長さ以上のものを抽出。**コンティグごとのGC含量計算**。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

GC含量



・ 解析 | 一般 | GC含量 (GC contents)

ここでは、multi-fasta形式ファイルを読み込んでコンティグごとのGC含量 (GC contents)を出力するやり方を示します。
ここでは以下の二つを例題として行います:
1.「イントロダクション | 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)
2.「ファイルの読み込み | マップ前 | FASTA形式」で読み込んだ250 readsからなるtest1.fastaファイル
「description」「C,Gの総数」「A,C,G,Tの総数」「配列長」「%GC含量」をファイルに出力するやり方を例示します。
尚、ここでは%GC含量の計算を「CGの総数/ACGTの総数」で計算していますので、もしGC含量を計算したい配列中にNなどが含まれる場合でNなどを含めた「配列長」を分母にしたい場合にはGC含量を計算をする数式中の「CG/ACGT*100」を「CG/width(reads)*100」に変更してください。

「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. hoge4.faファイルの場合:

```
------    ここから    ------
in_f <- "hoge4.fa"                                  #読み込みたいFASTA形式のファイル名を指定してin_
out_f <- "hoge1.txt"                                #出力ファイル名を指定

#必要なパッケージをロード
library(Biostrings)

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

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)                   #A,C,G,T,..の数を各配列ごとにカウントした結果を
CG <- rowSums(count[,2:3])                          #C,Gの総数を計算してCGに格納
ACGT <- rowSums(count[,1:4])                        #A,C,G,Tの総数を計算してACGTに格納
out <- CG/ACGT*100                                  #%GC含量を計算してoutに格納

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)          #ファイルに出力したい情報を連結してtmpに格納
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報を与えている
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの中身をout_fで指定したファイ

------    ここまで    ------
```

配列ごとのGC含量を計算したいとき

GC含量

R Console

```
> in_f <- "hoge4.fa"
> out_f <- "hoge1.txt"
>
> #必要なパッケージをロード
> library(Biostrings)
>
> #入力ファイルの読み込み
> reads <- readDNAStringSet(in_f, format="fasta")
>
> #GC含量(GC content)計算のところ
> count <- alphabetFrequency(reads)
> CG <- rowSums(count[,2:3])
> ACGT <- rowSums(count[,1:4])
> out <- CG/ACGT*100
>
> #出力用に結果をまとめている
> tmp <- cbind(names(reads), CG, ACGT, width(reads), out)
> colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_conte
> write.table(tmp, out_f, sep="\t", append=F, quote=F, row.names=F, co
>
```

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
```

hoge1.txt

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	contig_1	16	24	24	66.67
3	contig_2	65	103	103	63.11
4	contig_3	33	65	65	50.77
5	contig_4	25	49	49	51.02

hoge1

(スライド7の)シロイヌナズナのGC含量計算は、ファイル名部分を変更して実行しただけです

参考

GC含量

```
------    ここから    ------
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージをロード
library(Biostrings)

#入力ファイルの読み込み
reads <- readDNAStringSet(in_f, format="fasta")

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads
colnames(tmp) <- c("description", "CG", "ACGT",
write.table(tmp, out_f, sep="¥t", append=F, quot

------    ここまで    ------
```

alphabetFrequency関数を適用すると塩基ごとの出現頻度が得られます…が「M, R, W, …」は？

hoge4.fa - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

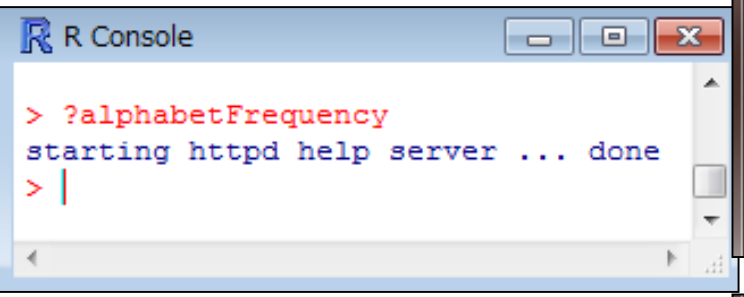
```
>contig_1
CGGACAGCTCCTCGGCATCCGGAT
>contig_2
GTCTGCCTCAAGCGCCCCAAGTGGGTTTGGAGGCCTAACATCGCAAGTCG
ACACTCAGTCCGGCCGTCTGGTTGGCAGGGGCAGAGACCCAGCACACCCT
GTC
>contig_3
TGTAGGAGAAGGGCGGTATCAGCGTCCACTTACACGATCCGTTACTAATT
GTATGAGGTCGGGCA
>contig_4
CGTGCTGATTCCACACAGCAGTAAACGCGGACCTCTACCTATGAACATG
```

R Console

```
>
> reads
  A DNAStringSet instance of length 4
    width seq                                      names
[1]    24 CGGACAGCTCCTCGGCATCCGGAT                 contig_1
[2]   103 GTCTGCCTCAAGCGC...CCAGCACACCCTGTC        contig_2
[3]    65 TGTAGGAGAAGGGCG...GTATGAGGTCGGGCA        contig_3
[4]    49 CGTGCTGATTCCACA...CTACCTATGAACATG        contig_4
> count
      A  C  G  T M R W S Y K V H D B N - +
[1,]  4  9  7  4 0 0 0 0 0 0 0 0 0 0 0 0 0
[2,] 20 34 31 18 0 0 0 0 0 0 0 0 0 0 0 0 0
[3,] 16 13 20 16 0 0 0 0 0 0 0 0 0 0 0 0 0
[4,] 14 15 10 10 0 0 0 0 0 0 0 0 0 0 0 0 0
> ?alphabetFrequency
>
```

参考

GC含量



alphabetFrequency関数がどういう出力結果を返すのか詳細を知りたい場合はValueのところを読むとよい

letterFrequency {Biostrings} R Documentation

Calculate the frequency of letters in a biological sequence, or the consensus matrix of a set of sequences

Description

Given a biological sequence (or a set of biological sequences), the `alphabetFrequency` function computes the frequency of each letter of the relevant alphabet.

Usage

```
alphabetFrequency(x, as.prob=FALSE, ...)
hasOnlyBaseLetters(x)
```

Arguments

x	An XString, XStringSet, XStringViews or MaskedXString object for

Details

`alphabetFrequency`, `letterFrequency`, and `letterFrequencyInSlidingView` are generic functions defined in the Biostrings package

Value

`alphabetFrequency` returns an integer vector when `x` is an XString or MaskedXString object. When `x` is an XStringSet or XStringViews object, then it returns an integer matrix with `length(x)` rows where the `i`-th row contains the frequencies for `x[[i]]`. If `x` is a DNA or RNA input, then the returned vector is named with the letters in the alphabet. If the `baseOnly` argument is `TRUE`, then the returned vector has only 5 elements: 4 elements corresponding to the 4 nucleotides + the 'other' element.

参考

GC含量

Value

`alphabetFrequency` returns an integer vector when `x` is an XString or MaskedXString object. When `x` is an XStringSet or XStringViews object, then it returns an integer matrix with `length(x)` rows where the `i`-th row contains the frequencies for `x[[i]]`. If `x` is a DNA or RNA input, then the returned vector is named with the letters in the alphabet. If the `baseOnly` argument is `TRUE`, then the returned vector has only 5 elements: 4 elements corresponding to the 4 nucleotides + the 'other' element.

M(A/C), R(A/G), W(A/T), S(C/G), …, N(A/C/G/T)という事実は常識?!なので書かれていない…。

R Console

```
> alphabetFrequency(reads)
      A  C  G  T M R W S Y K V H D B N - +
[1,]  4  9  7  4 0 0 0 0 0 0 0 0 0 0 0 0 0
[2,] 20 34 31 18 0 0 0 0 0 0 0 0 0 0 0 0 0
[3,] 16 13 20 16 0 0 0 0 0 0 0 0 0 0 0 0 0
[4,] 14 15 10 10 0 0 0 0 0 0 0 0 0 0 0 0 0
> alphabetFrequency(reads, baseOnly = TRUE)
      A  C  G  T other
[1,]  4  9  7  4     0
[2,] 20 34 31 18     0
[3,] 16 13 20 16     0
[4,] 14 15 10 10     0
>
```

argument（独立変数 or **引数**）とは、出力結果の形式などを指定するオプション。関数ごとに異なる

参考

GC含量

Usage

```
alphabetFrequency(x, as.prob=FALSE, ...)
hasOnlyBaseLetters(x)
uniqueLetters(x)
```

Arguments

x	An XString, XStringSet, XStringVie object for alphabetFrequency, lett uniqueLetters. DNA or RNA input for hasOnlyBase An XString object for letterFreque A character vector, or an XStringSet for consensusMatrix.
as.prob	A consensus matrix (as returned by consensusMatrix), or an XStringSet or XStringViews object for consensusString. If TRUE then probabilities are reported, otherwise counts (the default).

R Console

```
> alphabetFrequency(reads, baseOnly = TRUE)
      A  C  G  T other
[1,]  4  9  7  4     0
[2,] 20 34 31 18     0
[3,] 16 13 20 16     0
[4,] 14 15 10 10     0
> alphabetFrequency(reads, as.prob = TRUE)
          A         C         G         T M R W S Y K V H D B N - +
[1,] 0.1666667 0.3750000 0.2916667 0.1666667 0 0 0 0 0 0 0 0 0 0 0 0 0
[2,] 0.1941748 0.3300971 0.3009709 0.1747573 0 0 0 0 0 0 0 0 0 0 0 0 0
[3,] 0.2461538 0.2000000 0.3076923 0.2461538 0 0 0 0 0 0 0 0 0 0 0 0 0
[4,] 0.2857143 0.3061224 0.2040816 0.2040816 0 0 0 0 0 0 0 0 0 0 0 0 0
> alphabetFrequency(reads, as.prob = TRUE, baseOnly = TRUE)
          A         C         G         T other
[1,] 0.1666667 0.3750000 0.2916667 0.1666667     0
[2,] 0.1941748 0.3300971 0.3009709 0.1747573     0
[3,] 0.2461538 0.2000000 0.3076923 0.2461538     0
[4,] 0.2857143 0.3061224 0.2040816 0.2040816     0
>
```

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
```

相対出現頻度（probability）にすることも可能です。

参考

GC含量

```
------    ここから    ------
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージをロード
library(Biostrings)

#入力ファイルの読み込み
reads <- readDNAStringSet(in_f, format="fasta")

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)
colnames(tmp) <- c("description", "CG", "ACGT", "Length"
write.table(tmp, out_f, sep="¥t", append=F, quote=F, rov

------    ここまで    ------
```

「コンティグごとのCとGの出現頻度の和」は、「行の和」を計算することで得ることができる

R Console

```
> count
      A  C  G  T M R W S Y K V H D B N - +
[1,]  4  9  7  4 0 0 0 0 0 0 0 0 0 0 0 0 0
[2,] 20 34 31 18 0 0 0 0 0 0 0 0 0 0 0 0 0
[3,] 16 13 20 16 0 0 0 0 0 0 0 0 0 0 0 0 0
[4,] 14 15 10 10 0 0 0 0 0 0 0 0 0 0 0 0 0
> dim(count)
[1]  4 17
> count[,2:3]
      C  G
[1,]  9  7
[2,] 34 31
[3,] 13 20
[4,] 15 10
> rowSums(count[,2:3])
[1] 16 65 33 25
>
```

参考

GC含量

```
------    ここから    ------
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージをロード
library(Biostrings)

#入力ファイルの読み込み
reads <- readDNAStringSet(in_f, format="fasta")

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報を与えている
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

R Console

```
> CG
[1] 16 65 33 25
> ACGT
[1]  24 103  65  49
> CG/ACGT
[1] 0.6666667 0.6310680 0.5076923 0.5102041
> CG/ACGT*100
[1] 66.66667 63.10680 50.76923 51.02041
> ATGC
 エラー:  オブジェクト 'ATGC' がありません
>
```

出力ファイルの形式やヘッダー行はどのようにして決めているのか？

hoge1.txt

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	contig_1	16	24	24	66.67
3	contig_2	65	103	103	63.11
4	contig_3	33	65	65	50.77
5	contig_4	25	49	49	51.02

hoge1

参考

GC含量

```
------    ここから    ------
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージをロード
library(Biostrings)

#入力ファイルの読み込み
reads <- readDNAStringSet(in_f, format="fasta")

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)
colnames(tmp) <- c("description", "CG", "ACGT", "Length"
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row

------    ここまで    ------
```

R Console

```
> reads
  A DNAStringSet instance of length 4
    width seq                     names
[1]    24 CGGACAGCT...ATCCGGAT contig_1
[2]   103 GTCTGCCTC...ACCCTGTC contig_2
[3]    65 TGTAGGAGA...GTCGGGCA contig_3
[4]    49 CGTGCTGAT...TGAACATG contig_4
> names(reads)
[1] "contig_1" "contig_2" "contig_3" "contig_4"
> CG
[1] 16 65 33 25
> cbind(names(reads), CG)
                CG
[1,] "contig_1" "16"
[2,] "contig_2" "65"
[3,] "contig_3" "33"
[4,] "contig_4" "25"
>
```

hoge1.txt

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	contig_1	16	24	24	66.67
3	contig_2	65	103	103	63.11
4	contig_3	33	65	65	50.77
5	contig_4	25	49	49	51.02

hoge1

同じ要素数からなるベクトル同士を列（column）方向で結合（**bind**）

参考

GC含量

```
------    ここから    ------
in_f <- "hoge4.fa"                                  #読み込みたいFASTA形式の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

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

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)                   #A,C,G,T,..の数を各配列ごとにカウントした結果をcountに格納
CG <- rowSums(count[,2:3])                          #C,Gの総数を計
ACGT <- rowSums(count[,1:4])                        #A,C,G,Tの総数
out <- CG/ACGT*100                                  #%GC含量を計算

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)          #ファイルに出力したい情報を連結してtmpに格納
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報を与えている
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

cbind関数適用時に、出力したい順番に並べているだけです

hoge1.txt

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	contig_1	16	24	24	66.67
3	contig_2	65	103	103	63.11
4	contig_3	33	65	65	50.77
5	contig_4	25	49	49	51.02

hoge1

参考

GC含量

```
#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)          #ファイルに出力したい情報を連結してtmpに格納
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報を与えている
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの中身をout_fで指定したファイル名で保存。

------    ここまで    ------
```

R Console

```
>
> #GC含量(GC content)計算のところ
> count <- alphabetFrequency(reads)
> CG <- rowSums(count[,2:3])
> ACGT <- rowSums(count[,1:4])
> out <- CG/ACGT*100
>
> #出力用に結果をまとめている
> tmp <- cbind(names(reads), CG, ACGT, width(reads), out)
> tmp
                CG    ACGT          out
[1,] "contig_1" "16" "24"   "24"   "66.6666666666667"
[2,] "contig_2" "65" "103"  "103"  "63.1067961165049"
[3,] "contig_3" "33" "65"   "65"   "50.7692307692308"
[4,] "contig_4" "25" "49"   "49"   "51.0204081632653"
> colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")
> tmp
     description CG   ACGT  Length %GC_contents
[1,] "contig_1"  "16" "24"  "24"   "66.6666666666667"
[2,] "contig_2"  "65" "103" "103"  "63.1067961165049"
[3,] "contig_3"  "33" "65"  "65"   "50.7692307692308"
[4,] "contig_4"  "25" "49"  "49"   "51.0204081632653"
>
```

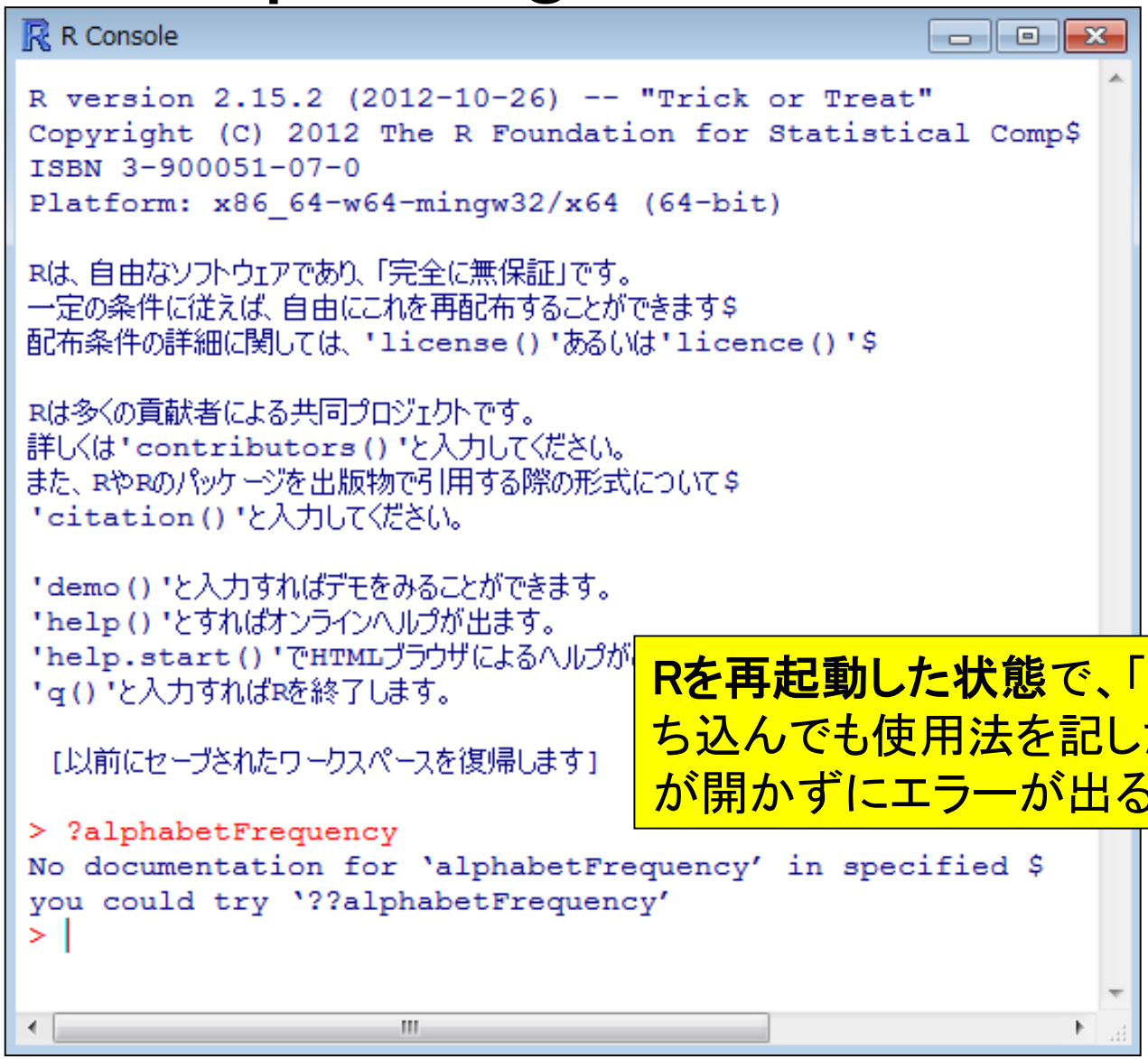
hoge1.txt

	A	B	C	D	E
1	description	CG	ACGT	Length	%GC_contents
2	contig_1	16	24	24	66.67
3	contig_2	65	103	103	63.11
4	contig_3	33	65	65	50.77
5	contig_4	25	49	49	51.02

hoge1

列(column)の名前(names)を
適当に変更して出力しています

パッケージ（package）って何？



パッケージ（package）って何？

```
------    ここから    ------
in_f <- "hoge4.fa"
out_f <- "hoge1.txt"

#必要なパッケージをロード
library(Biostrings)

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

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)
CG <- rowSums(count[,2:3])
ACGT <- rowSums(count[,1:4])
out <- CG/ACGT*100

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT,
colnames(tmp) <- c("description", "CG
write.table(tmp, out_f, sep="¥t", app

------    ここまで    ------
```

「Biostrings」というパッケージをlibrary関数を用いて読み込むことによって、Biostringsパッケージが提供する関数群の中の一つであるalphabetFrequency関数を利用できるんです。

R Console

```
> ?alphabetFrequency
No documentation for 'alphabetFrequency' in specified $
you could try '??alphabetFrequency'
> library(Biostrings)
 要求されたパッケージ BiocGenerics をロード中です

 次のパッケージを付け加えます: ''BiocGenerics''

The following object(s) are masked from 'package:stats$

    xtabs

The following object(s) are masked from 'package:base':

    anyDuplicated, cbind, colnames, duplicated,
    eval, Filter, Find, get, intersect, lapply,
    Map, mapply, mget, order, paste, pmax,
    pmax.int, pmin, pmin.int, Position, rbind,
    Reduce, rep.int, rownames, sapply, setdiff,
    table, tapply, union, unique

 要求されたパッケージ IRanges をロード中です
> ?alphabetFrequency
starting httpd help server ... done
>
```

た結果をcountに格納

に格納

たファイル名で保存。

参考

解析 | 一般 | GC含量 (GC contents)

ここでは、multi-fasta形式ファイルを読み込んでコンティグごとのGC含量 (GC contents)を出力するやり方を示します
ここでは以下の二つを例題として行います:
1.「イントロダクション | 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
2.「ファイルの読み込み | マップ前 | FASTA形式」で読み込んだ250 readsからなる
「description」「C,Gの総数」「A,C,G,Tの総数」「配列長」「%GC含量」をファイルに出力
尚、ここでは%GC含量の計算を「CGの総数/ACGTの総数」で計算していますので、
長」を分母にしたい場合にはGC含量を計算をする数式中の「CG/ACGT*100」を「CG

「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

1. hoge4.faファイルの場合:

```
------    ここから    ------
in_f <- "hoge4.fa"                          #読み込みたいFASTA
out_f <- "hoge1.txt"                        #出力ファイル名を指

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

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

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)           #A,C,G,T,..の数を各
CG <- rowSums(count[,2:3])                  #C,Gの総数を計算し
ACGT <- rowSums(count[,1:4])                #A,C,G,Tの総数を計
out <- CG/ACGT*100                          #%GC含量を計算して

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)   #ファイルに出力し
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名情報
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T)#tmpの

------    ここまで    ------
```

2. test1.fastaファイルの場合:

```
------    ここから    ------
in_f <- "test1.fasta"                       #読み込みたいFASTA
out_f <- "hoge2.txt"                        #出力ファイル名を指

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

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

#GC含量(GC content)計算のところ
count <- alphabetFrequency(reads)           #A,C,G,T,..の数を各
CG <- rowSums(count[,2:3])                  #C,Gの総数を計算し
ACGT <- rowSums(count[,1:4])                #A,C,G,Tの総数を計
out <- CG/ACGT*100                          #%GC含量を計算して

#出力用に結果をまとめている
tmp <- cbind(names(reads), CG, ACGT, width(reads), out)   #ファイルに出
colnames(tmp) <- c("description", "CG", "ACGT", "Length", "%GC_contents")#列名
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F, col.names=T

------    ここまで    ------
```

BioconductorのBiostringsのwebページ

Bioconductor
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Search:

Home | Install | Help | Develo

Home » Bioconductor 2.12 » Software Packages » Biostrings

Biostrings

String objects representing biological sequences, and matching algorithms

Bioconductor version: Release (2.12)

Memory efficient string containers, string matching algorithms, and other utilities, for fast manipulation of large biological sequences or sets of sequences.

Author: H. Pages, P. Aboyoun, R. Gentleman, and S. DebRoy

Maintainer: H. Pages <hpages at fhcrc.org>

To install this package, start R and enter:

```
source("http://bioconductor.org/biocLite.R")
biocLite("Biostrings")
```

パッケージを個別にインストールする場合

To cite this package in a publication, start R and enter:

```
citation("Biostrings")
```

Documentation

PDF	R Script	A short presentation of the basic classes defined in Bios
PDF	R Script	Biostrings Quick Overview
PDF	R Script	Handling probe sequence information
PDF	R Script	Multiple Alignments
PDF	R Script	Pairwise Sequence Alignments
PDF		Reference Manual
Text		NEWS

使い方の解説記事はPDFのところをクリック。例えば…

Details

biocViews	DataImport, DataRepresentation, Genetics, Infrastructure, SequenceMatching, Sequencing, Software

参考

Biostrings Quick Overview

Hervé Pagès
Fred Hutchinson Cancer Research Center
Seattle, WA

April 3, 2013

Please note that *most* but *not all* the functionalities provided by the Biostrings package are listed in document.

Function	Description
length	Return the number of sequences in an object.
names	Return the names of the sequences in an object.
[	Extract sequences from an object.
head, tail	Extract the first or last sequences from an object.
rev	Reverse the order of the sequences in an object.
c	Put in a single object the sequences from 2 or more objects.
width, nchar	Return the sizes (i.e. number of letters) of all the sequences in object.
==, !=	Element-wise comparison of the sequences in 2 objects.
match, %in%	Analog to match and %in% on character vectors.
duplicated, unique	Analog to duplicated and unique on character vectors.
sort, order	Analog to sort and order on character vectors, except that ordering of DNA or Amino Acid sequences doesn't depend on locale.
split, relist	Analog to split and relist on character vectors, except that result is a *DNAStringSetList* or *AAStringSetList* object.

Table 1: Low-level manipulation of *DNAStringSet* or *AAStringSet* objects.

Function	Description
subseq, subseq<-	Extract or replace subsequences in a set of sequences.
reverse complement reverseComplement	Compute the reverse, complement, or reverse-complement, of a s of DNA sequences.
translate	Translate a set of DNA sequences into a set of Amino Acid s quences.
chartr	T
replaceLetterAt	R

Table 2: Basic transformations of sequences.

Function	Description
alphabetFrequency letterFrequency	Tabulate the letters (all the letters in the alphabet for alphabetFrequency, only the specified letters for letterFrequency) of a sequence or set of sequences.
letterFrequencyInSlidingView	Specialized version of letterFrequency that tallies the requested letter frequencies for a fixed-width view that is conceptually slid along the input sequence.
consensusMatrix	Computes the consensus matrix of a set of sequences.
dinucleotideFrequency trinucleotideFrequency oligonucleotideFrequency	Fast 2-mer, 3-mer, and k-mer counting for DNA or RNA.
nucleotideFrequencyAt	Tallies the short sequences formed by extracting the nucleotides found at a set of fixed positions from each sequence of a set of DNA or RNA sequences.

Table 3: Counting / tabulating.

Function	Description
matchPattern countPattern	Find/count all the occurrences of a given pattern (typically short) in a reference sequence (typically long). Support mismatches and indels.
vmatchPattern vcountPattern	Find/count all the occurrences of a given pattern (typically short) in a set of reference sequences. Support mismatches and indels.
matchPDict countPDict whichPDict	Find/count all the occurrences of a set of patterns in a reference sequence. (whichPDict only identifies which patterns in the set have at least one match.) Support a small number of mismatches.
vmatchPDict vcountPDict vwhichPDict	[Note: vmatchPDict not implemented yet.] Find/count all the occurrences of a set of patterns in a set of reference sequences. (whichPDict only identifies for each reference sequence which patterns in the set have at least one match.) Support a small number of mismatches.
pairwiseAlignment	Solve (Needleman-Wunsch) global alignment, (Smith-Waterman) ls-free) overlap alignment problems.
	rences of a Position Weight Matrix in a
	nking patterns from sequences.
	n a reference sequence i.e. matches speci- attern, and a maximum distance between
matchProbePair	Find all the amplicons that match a pair of probes in a reference sequence.

「Biostrings」パッケージ中の関数を使いこなせると、他の自然言語処理系プログラミング言語（perlやruby）を改めて勉強しなくても必要な解析の大部分が可能です

Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - 一定の長さ以上のものを抽出。コンティグごとのGC含量計算。
- **Rで（比較）トランスクリプトーム解析**
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

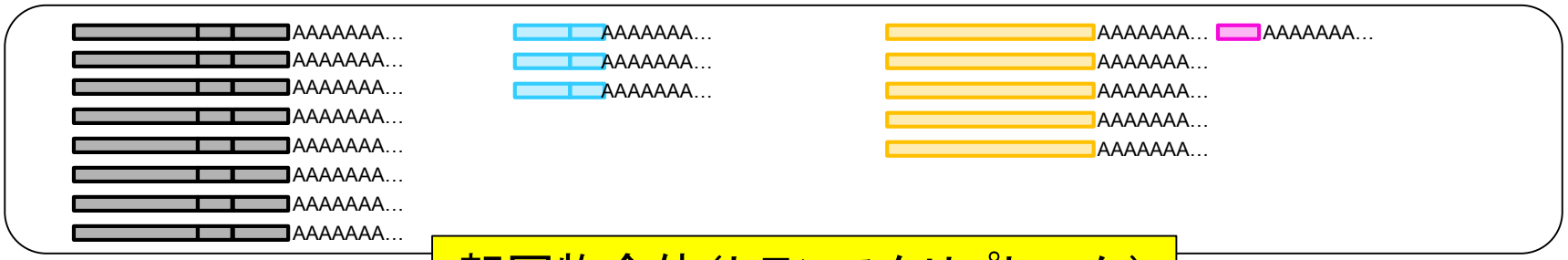
トランスクリプトームとは

- ある状態のあるサンプル（例：目）のあるゲノムの領域

ヒト

遺伝子全体（ゲノム）

・どの染色体上のどの領域にどの遺伝子があるかは調べる個体（例：ヒト）が同じなら不変（目だろうが心臓だろうが…）



転写物全体（トランスクリプトーム）

・遺伝子1は沢山転写されている（発現している）
・遺伝子4はごくわずかしか転写されてない
・…

トランスクリプトームとは

- ある状態のあるサンプル（例：目）のあるゲノムの領域

遺伝子全体（ゲノム）

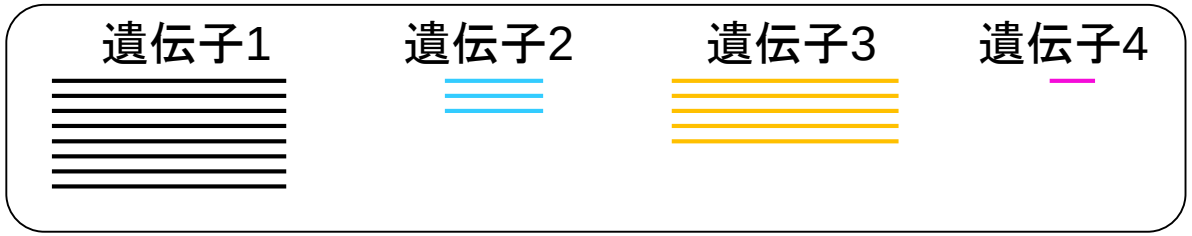
・どの染色体上のどの領域にどの遺伝子があるかは調べる個体（例：ヒト）が同じなら不変（目だろうが心臓だろうが…）

転写物全体（トランスクリプトーム）

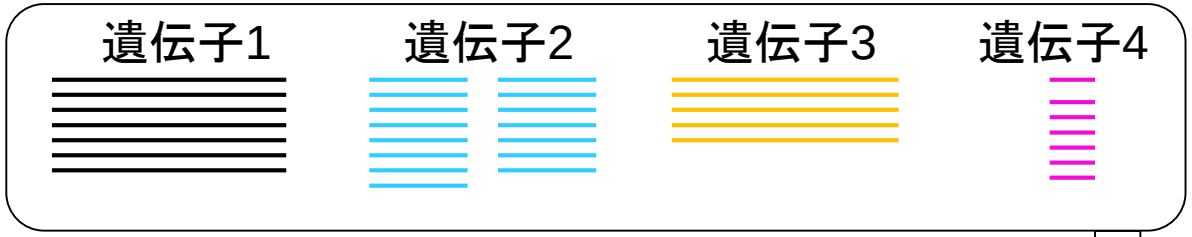
・遺伝子2は光刺激に応答して発現亢進
・遺伝子4も光刺激に応答して発現亢進

トランスクリプトーム情報を得る手段

- 光刺激前（T1）の目のトランスクリプトーム



- 光刺激後（T2）の目のトランスクリプトーム



これがいわゆる
「遺伝子発現行列」

	T1	T2
遺伝子1	8	7
遺伝子2	3	15
遺伝子3	5	5
遺伝子4	1	7
…	…	…

・マイクロアレイ
・電気泳動に基づく方法
・配列決定に基づく方法

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

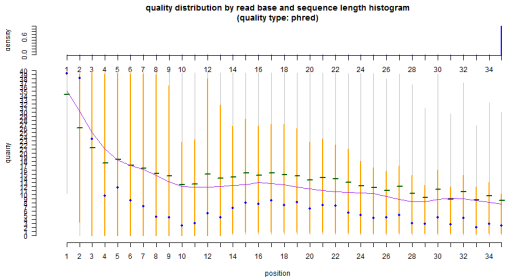
↓

マッピング

↓

データ解析

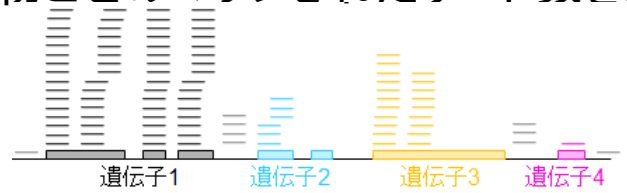
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

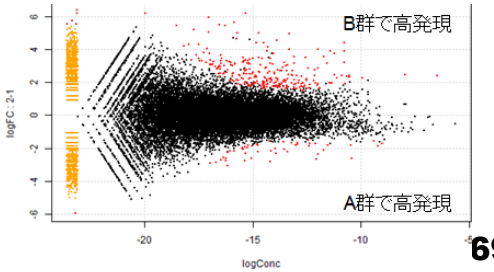
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など



雑感

- 一口にトランスクリプトーム解析といっても目的や手段は多様。
  - トランスクリプトーム配列取得
    - ゲノム配列既知の場合、Cufflinksなどを用いて遺伝子構造推定（アノテーション）
    - ゲノム配列未知の場合、Trinityなどのトランスクリプトーム用アセンブラを実行
  - 遺伝子（or isoform）ごとの発現量推定
    - RSEMなどを利用して発現量情報を得る
    - ある特定のサンプル内での遺伝子間の発現量の大小関係を知りたい
    - Length biasやGC含量biasなどの各種補正がポイント
    - 遺伝子レベルの発現量　⇔　isoformレベルの発現量の正確な推定
  - 比較するサンプル間で発現変動している遺伝子（or isoform）の同定
    - TCCパッケージ（など）を利用して発現変動遺伝子（DEG）を得る
    - Sequence depthやサンプル間で発現している遺伝子のcomposition biasの補正がポイント
    - （GO解析など）DEG結果を用いる多くの下流解析結果に影響を及ぼす

研究目的別留意点

- ある特定のサンプル内での**遺伝子間**の発現量の大小関係を知りたい場合
  - 「配列長」由来bias：長いほど沢山sequenceされる
  - 「GC含量」由来bias：カウント数の分布がGC含量依存的である
- **サンプル間**比較（sample A vs. Bなど）で、発現変動遺伝子（DEG）を調べたい場合
  - 「sequence depthの違い」：総リード数がx倍違うと全体的にx倍変動…
  - 「組成の違い」：サンプル特異的高発現遺伝子の存在で比較困難に…

配列長を考慮した発現量推定のイメージ

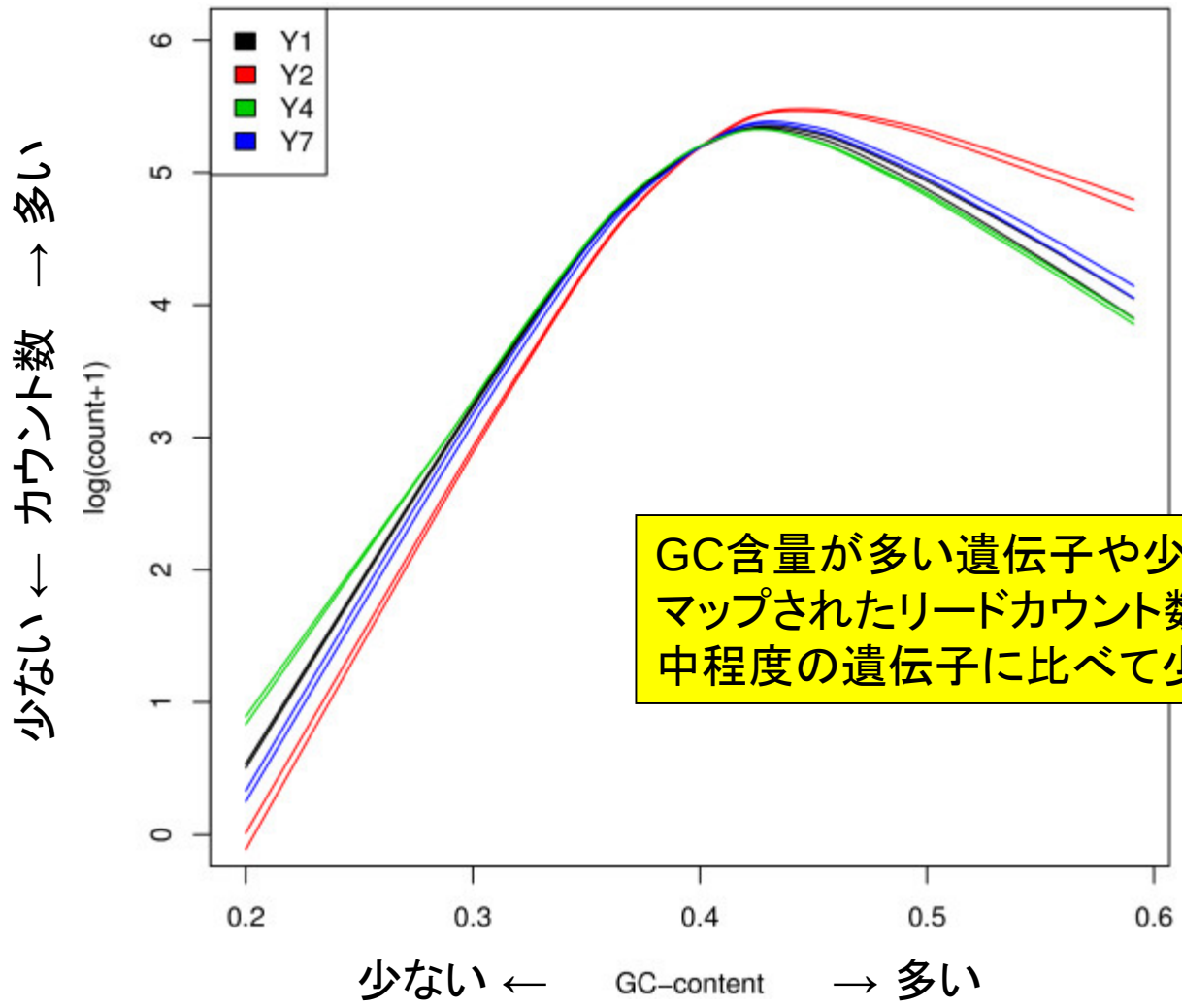
- gene1: 3 exons (middle length), 14 reads mapped (**low** coverage)
- gene2: 3 exons (middle length), 56 reads mapped (**high** coverage)
- gene3: 2 exons (**short** length), 12 reads mapped (middle coverage)
- gene4: 2 exons (**long** length), 31 reads mapped (middle coverage)

マップされたリード分布　　生リードカウント結果　　補正度の発現量

・長さが同じならリード数の多い方が発現量高い（gene 1 vs. 2）
・長いほどマップされるリード数が多くなる効果を補正する必要がある（gene 3 vs. 4）

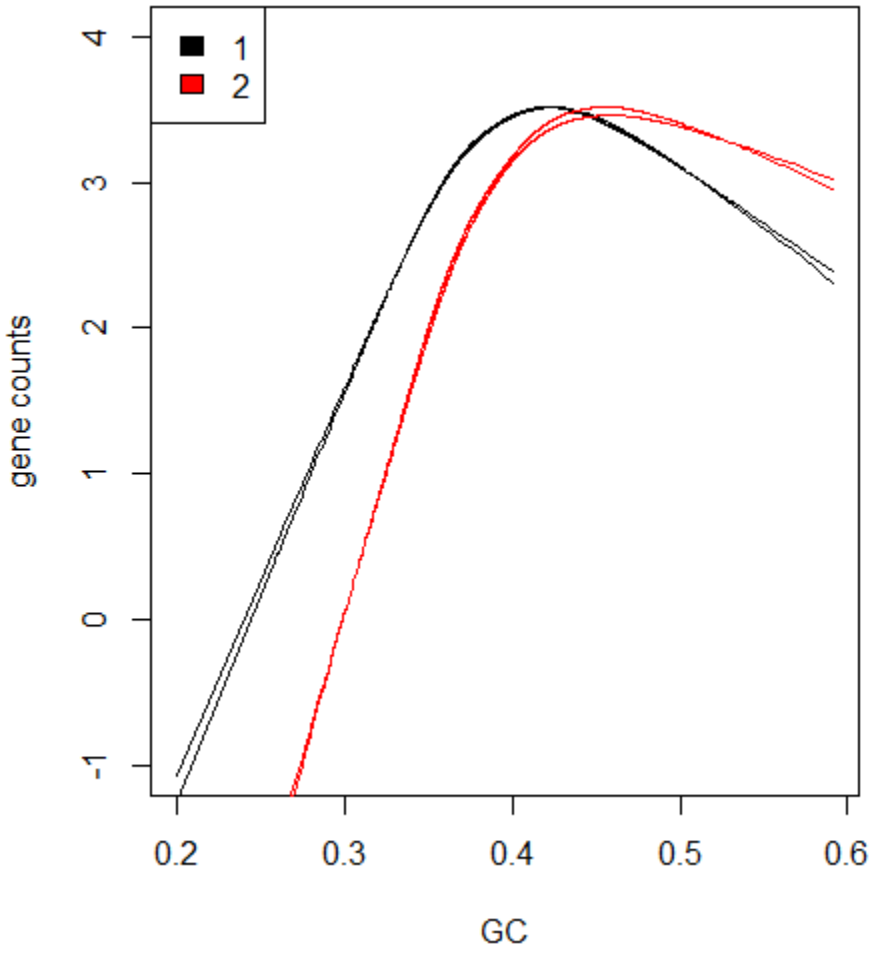
一つのサンプル内で転写物（遺伝子）間の発現レベルの大小を比較したい場合には配列長を考慮すべきである

GC biasの実例

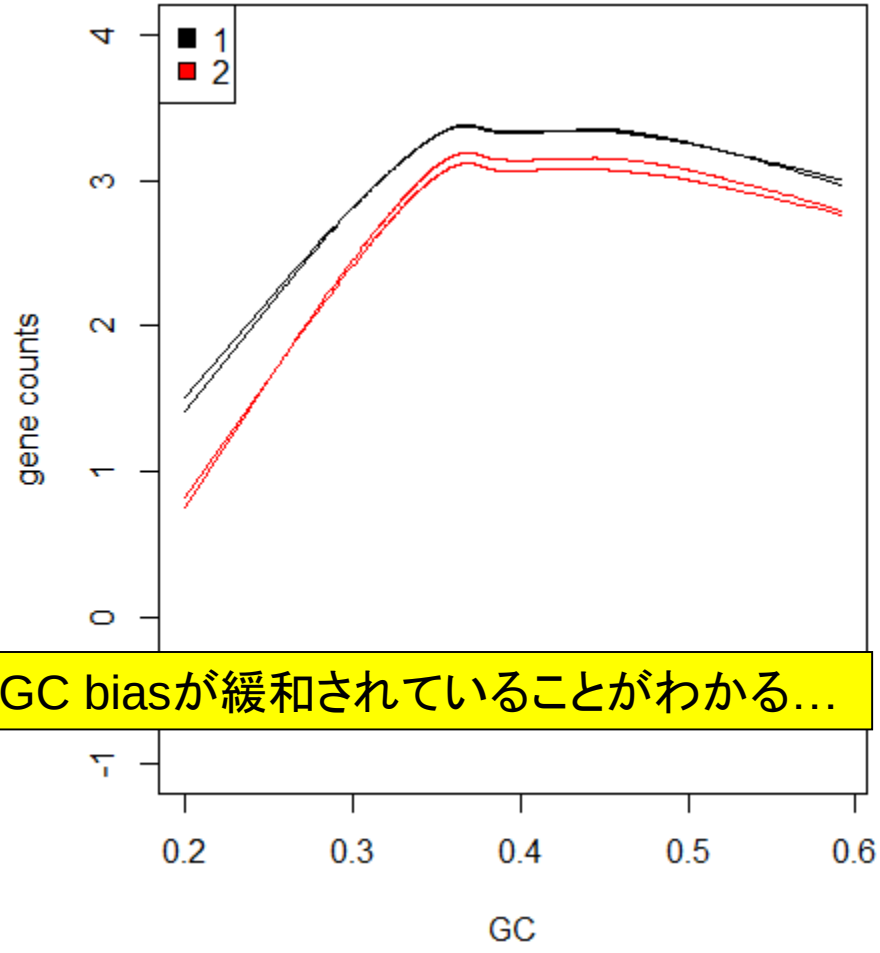


GC含量が多い遺伝子や少ない遺伝子上にマップされたリードカウント数は、GC含量が中程度の遺伝子に比べて少ない傾向にある

GC bias補正（EDASeqパッケージ）



Quantile
正規化



GC biasが緩和されていることがわかる…

比較トランスクリプトーム解析の流れ

複数のFASTQファイル

↓

リファレンス配列の作成

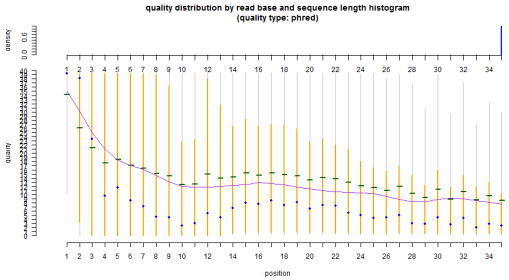
↓

マッピング

↓

データ解析

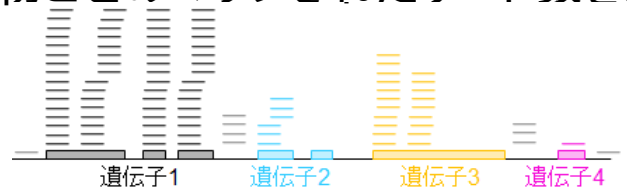
クオリティチェック



アセンブル結果（multi-fasta）ファイルから平均長やトータルの長さなどの基本情報を抽出

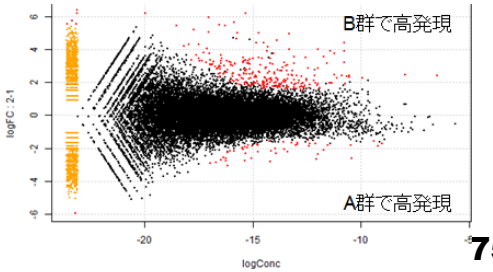
Total length (bp)	2993
Number of contigs	4
Average length	748.3
Median length	784
Max length	888
Min length	537
N50	886
GC content	0.524

マッピング結果（BED形式）ファイルを入力として、転写物ごとのマップされたリード数をカウント



遺伝子1	40
遺伝子2	6
遺伝子3	20
遺伝子4	1

発現変動遺伝子のリストアップや、作図など

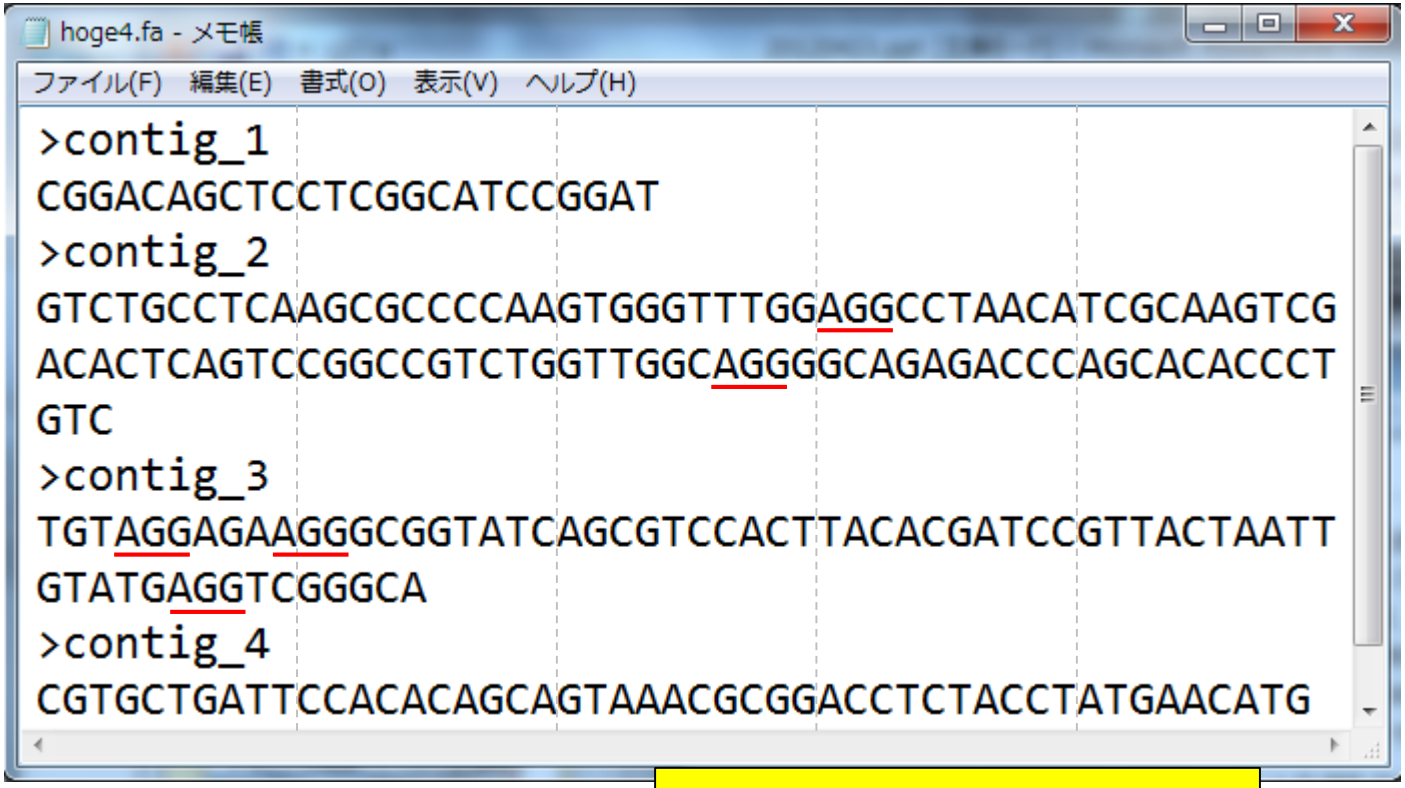


Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - 一定の長さ以上のものを抽出。コンティグごとのGC含量計算。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - **リードマッピング　→　数値化（カウントデータの取得）**
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

マッピング　＝　（大量高速）文字列検索

- マップされる側の配列：4コンティグ（or 4遺伝子 or 4染色体）
- マップする側のNGS由来塩基配列データ：”AGG”



出力ファイル：hoge2.txt

	start	end
contig_2	31	33
contig_2	77	79
contig_3	4	6
contig_3	10	12
contig_3	56	58

Rでやってみよう

パターンマッチング

- 前処理 | 75percentile正規化(第3四分位数を揃える)(last modified 2011/08/09)
- 前処理 | GAM正規化(Zheng 2011)(last modified 2011/07/25)
- 解析 | 一般 | アラインメント(ペアワイズ;基本編1) (last modified 2010/6/8)
- 解析 | 一般 | アラインメント(ペアワイズ;基本編2) (last modified 2010/6/8)
- 解析 | 一般 | アラインメント(ペアワイズ;応用編) (last modified 2010/6/8)
- 解析 | 一般 | パターンマッチング (last modified 2012/04/13) NEW
- 解析 | 一般 | GC含量 (GC content) (last modified 2010/7/1)
- 解析 | 一般 | Sequence logos (Schneider 1990) (last modified 2012/04/04) NEW
- 解

• **解析 | 一般 | パターンマッチング**

読み込んだリファレンス配列(reference sequence or subject sequence)から(短い)配列パターンを探す場合に利用します。
ここでは、以
1. Dihydrofo
Finger Nucl
2. 1. と同じ
3. 2. と同じ
4. multi-fas
5. multi-fas
6. multi-fas
7. multi-fas
(記述の仕

4. multi-fastaファイルhoge4.faを入力として、"AGG"でキーワード探索を行う場合：

```
------    ここから    ------
in_f <- "hoge4.fa"                                          #読み込みたいFASTA形式のファイル
out_f <- "hoge2.txt"                                        #出力ファイル名を指定してout_fに
param <- "AGG"                                              #調べたい配列パターンを指定して

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

#本番
out <- vmatchPattern(pattern=param, subject=seq)            #paramで指定した配列と100%マッチ
hoge <- cbind(start(unlist(out)), end(unlist(out)))         #一致領域の(start, end)の位置情報
colnames(hoge) <- c("start", "end")                         #行列hogeに列名を付加
rownames(hoge) <- names(unlist(out))                        #行列hogeに行名を付加
tmp <- cbind(rownames(hoge), hoge)                          #ファイルに出力したい情報を連結
write.table(tmp, out_f, sep="¥t", append=F, quote=F, row.names=F)#tmpの中身をout_fで指定したファ

------    ここまで    ------
```

4. multi-fastaファイルhoge4.faを入力として、"AGG"でキーワード探索を行う場合：

```
------    ここから    ------
in_f <- "hoge4.fa"                                  #読み込みたいFASTA形式のファイル
out_f <- "hoge2.txt"                                #出力ファイル名を指定してout_fに
param <- "AGG"                                      #調べたい配列パターンを指定して

#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
seq <- readDNAStringSet(in_f, format="fasta")       #in_fで指定したファイルの

#本番
out <- vmatchPattern(pattern=param, subject=seq)    #paramで指定した配列と100%
hoge <- cbind(start(unlist(out)), end(unlist(out)))  #一致領域の(start, end)の
colnames(hoge) <- c("start", "end")                 #行列hogeに列名を付加
```

出力ファイル：hoge2.txt

	start	end
contig_2	31	33
contig_2	77	79
contig_3	4	6
contig_3	10	12
contig_3	56	58

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
```

複数のキーワード探索も可能です

6. multi-fastaファイルhoge4.faをリファレンス配列（マップされる側）として、4リードからなるdata_reads.t

```
------    ここから    ------
in_f1 <- "hoge4.fa"
in_f2 <- "data_reads.txt"
out_f <- "hoge4.txt"

#必要なパッケージをロード
library(Biostrings)

#入力ファイルの読み込み
seq <- readDNAStringSet(in_f1, format="fasta")
reads <- readDNAStringSet(in_f2, format="fasta")

#本番
out <- c("in_f2", "in_f1", "start", "end")
for(i in 1:length(reads)){
    tmp <- vmatchPattern(pattern=as.character(reads[i]), subject=se
}
wri
---
```

• 解析 | 一般 | パターンマッチング

data_reads.txt

```
>seq1
TTT
>seq2
GGG
>seq3
ACT
>seq4
ACA
```

出力ファイル：hoge4.txt

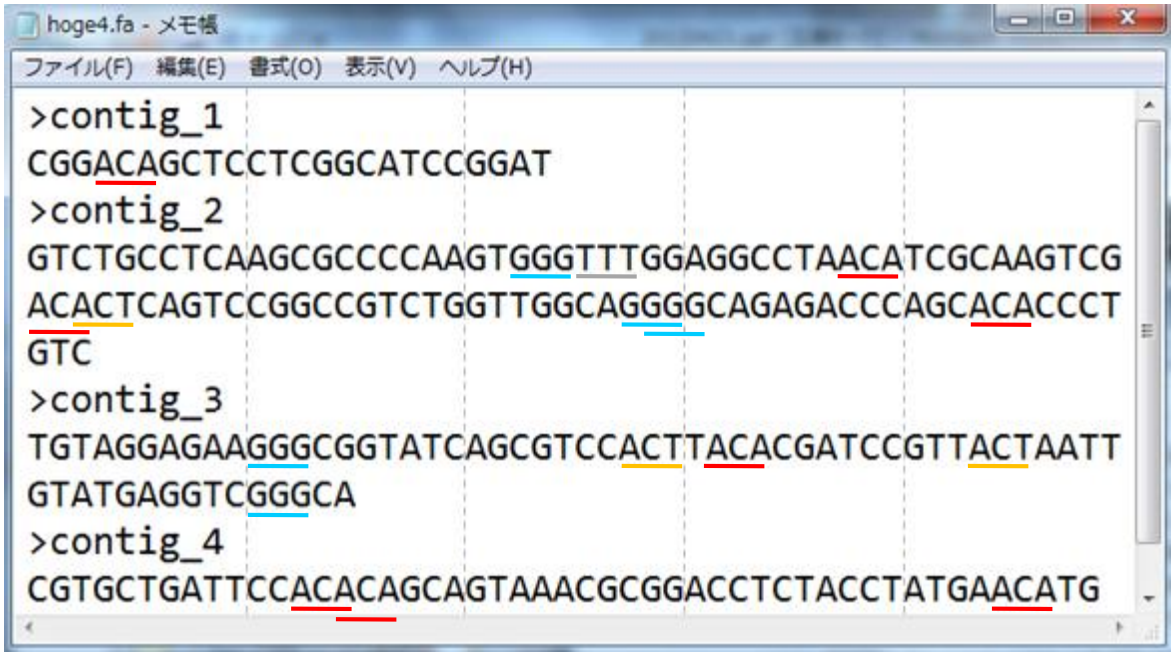
in_f2	in_f1	start	end
TTT	contig_2	26	28
GGG	contig_2	23	25
GGG	contig_2	78	80
GGG	contig_2	79	81
GGG	contig_3	11	13
GGG	contig_3	61	63
ACT	contig_2	53	55
ACT	contig_3	28	30
ACT	contig_3	44	46
ACA	contig_1	4	6
ACA	contig_2	38	40
ACA	contig_2	51	53
ACA	contig_2	94	96
ACA	contig_3	32	34
ACA	contig_4	13	15
ACA	contig_4	15	17
ACA	contig_4	45	47

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
```

出力結果ファイルと発現量の関係



data_reads.txt

```
>seq1
TTT
>seq2
GGG
>seq3
ACT
>seq4
ACA
```

出力ファイル：hoge4.txt

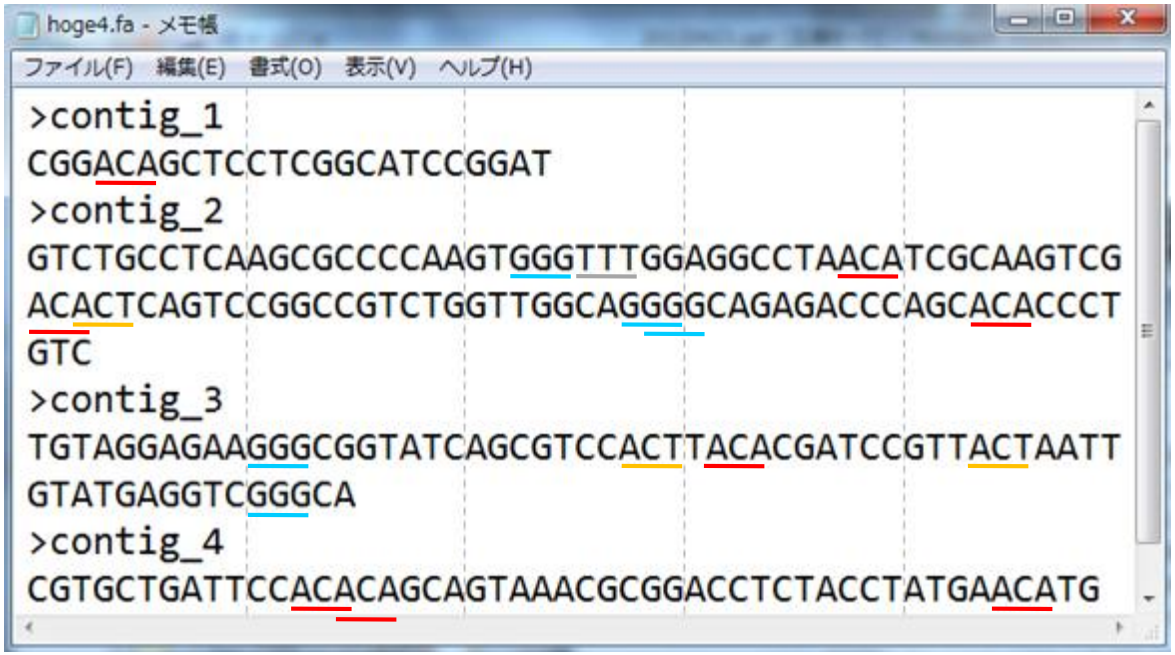
in_f2	in_f1	start	end
TTT	contig_2	26	28
GGG	contig_2	23	25
GGG	contig_2	78	80
GGG	contig_2	79	81
GGG	contig_3	11	13
GGG	contig_3	61	63
ACT	contig_2	53	55
ACT	contig_3	28	30
ACT	contig_3	44	46
ACA	contig_1	4	6
ACA	contig_2	38	40
ACA	contig_2	51	53
ACA	contig_2	94	96
ACA	contig_3	32	34
ACA	contig_4	13	15
ACA	contig_4	15	17
ACA	contig_4	45	47

multi mapper（複数個所にマップされるリード）の取り扱いは？

よく見かけるカウントデータ取得条件

- basic alignerの一つであるBowtieプログラムを利用して、リファレンス配列(ゲノム or トランスクリプトーム)の一カ所とのみ(最大2塩基ミスマッチまで許容して)一致するリード(uniquely mapped reads or unique mapper)数をカウント
  - Marioni et al., *Genome Res*., **18**:1509-1517, 2008
  - Bullard et al., *BMC Bioinformatics*, **11**:94, 2010
  - Risso et al., *BMC Bioinformatics*, **12**:480, 2011
  - ReCount (Frazee et al., *BMC Bioinformatics*, **12**:449, 2011)
  - …

Unique mapperのみにすると…



data_reads.txt

```
>seq1
TTT
>seq2
GGG
>seq3
ACT
>seq4
ACA
```

出力ファイル：hoge4.txt

in_f2	in_f1	start	end
TTT	contig_2	26	28
GGG	contig_2	23	25
GGG	contig_2	78	80
GGG	contig_2	79	81
GGG	contig_3	11	13
GGG	contig_3	61	63
ACT	contig_2	53	55
ACT	contig_3	28	30
ACT	contig_3	44	46
ACA	contig_1	4	6
ACA	contig_2	38	40
ACA	contig_2	51	53
ACA	contig_2	94	96
ACA	contig_3	32	34
ACA	contig_4	13	15
ACA	contig_4	15	17
ACA	contig 4	45	47

contig_1	1
contig_2	8
contig_3	5
contig_4	3

⇒

contig_1	0
contig_2	1
contig_3	0
contig_4	0

- フィルタリング｜NGS｜末端部分のトリミング (last modified 2012/09/12)
- 前処理｜について(last modified 2010/12/16)
- 前処理｜Trinity出力ファイルからFPKM値を取得(last modified 2011/10/20)
- 前処理｜トランスクリプトーム配列へのマップ後のファイルからマップされたリード数をカウント(BED形式ファイル)
- 前処理｜Ensembl Geneの長さを計算する(last modified 2011/08/15)
- 前処理｜転写物の配列長が長いほどマップされたリード数が多くなることを確認(last modified 2012/09/11)
- 前処理｜転写物の配列長が長いほど発現変動遺伝子とされる確率が上昇することを確認(last modified 2011/0
- 前処理｜発現レベルの定量化について(last modified 2013/01/22) NEW
- 前処理｜正
- 前処理｜正

• 前処理｜トランスクリプトーム配列へのマップ後のファイルからマップされたリード数をカウント(BED形式ファイル)

手元に「**multi-fasta形式のマップされる側の塩基配列ファイル**(例: h_rna.fastaやhoge4.fa)」と Bowtie (Langmead et al., 2009)などのマッピングプログラムを用いて得られた「**どこにマップされたかを表すBED形式ファイル**(例: SRR002324_t.bed、sample_1.bed、sample_2.bed)」があるとします。
発現変動遺伝子(DEG)同定の目的で用いられるTCCやedgeR などのRパッケージは遺伝子(あるいは転写物)ごとに何個のリードがマップされたかという「カウント行列(count matrix)」を入力とすることを前提としています。(ざっくり言えば)BED形式ファイルは、トランスクリプトーム配列にマップした場合、「どのtranscript(or 遺伝子)の(1列目)、どの位置から(2列目; start)、どの位置まで(3列目; end)にリードがマップされた」という情報からなります。それゆえ「行数がマップされたリード数」に相当し、「遺伝子ごとの出現数がカウント数」に相当するわけです。
したがって、基本的にはBED形式ファイルを読み込んで「遺伝子名に相当する列」の部分の情報のみを用いて「どの遺伝子名が何回出現したか」をカウントすればいいだけなんですが... 実際には、二つのサンプル(or 二つのBED形式ファイル)を比較する場合などには「片方のBEDファイル中に一度も出現しない遺伝子」が存在しうるため、「multi-fasta形式のマップされる側の塩基配列ファイル」も同時に読み込んでその遺伝子リストの並びで出力してやったほうが都合がいいのでここではそうしています。

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

1．マップされる側の配列（hoge4.fa）中のIDの並びでBED形式ファイル(sample_1.bed)中の出現回数をカウントする場合：

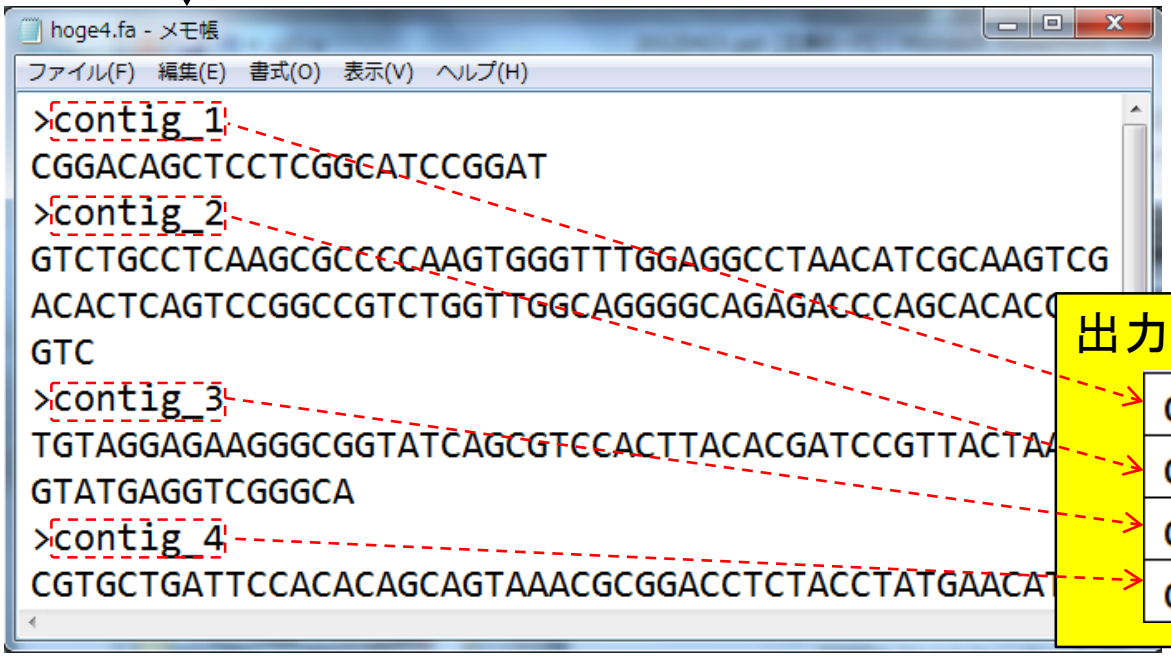
```
------    ここから    ------
in_f1 <- "sample_1.bed"                          #読み込みたいBED形式フ
in_f2 <- "hoge4.fa"                              #マップに用いたリファレ
out_f <- "output1.txt"                           #出力ファイル名を指定して out_f に格納

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

#入力ファイルの読み込みと必要な情報の抽出など
```

1.をやってみましょう。

入力ファイルと目的のおさらい

- 入力ファイル1：sample_1.bed
  - BED形式ファイル。1列目の情報のみを用いてコンティグ（遺伝子ID）ごとのカウント（出現回数）情報取得のために利用。
- 入力ファイル2：hoge4.fa
  - マップに用いたリファレンス配列。multi-fasta形式ファイル。Description行のコンティグ名（ID）の並びで結果を出力させるために利用。

in_f1	start	end
contig_2	26	28
contig_2	23	25
contig_2	78	80
contig_2	79	81
contig_3	11	13
contig_3	61	63
contig_2	53	55
contig_3	28	30
contig_3	44	46
contig_1	4	6
contig_2	38	40
contig_2	51	53
contig_2	94	96
contig_3	32	34
contig_4	13	15
contig_4	15	17
contig_4	45	47



出力ファイル：output1.txt

contig_1	1
contig_2	8
contig_3	5
contig_4	3

Contents

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - 一定の長さ以上のものを抽出。コンティグごとのGC含量計算。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - **発現変動解析**
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

比較解析用Rパッケージ（or スクリプト）

- *DEGSeq* (Wang *et al*., *Bioinformatics*, **26**: 136-138, 2010)
- *edgeR* (Robinson *et al*., *Bioinformatics*, **26**: 139-140, 2010)
- *GPseq* (Srivastava and Chen, *Nucleic Acids Res*., **38**: e170, 2010)
- *baySeq* (Hardcastle and Kelly, *BMC Bioinformatics*, **11**: 422, 2010)
- *DESeq* (Anders and Huber, *Genome Biol*., **11**: R106, 2010)
- *NBPSeq* (Di *et al*., *SAGMB*, **10**: article24, 2011)
- *TPSM* (Auer and Doerge, *SAGMB*, **10**: article26, 2011)
- *BBSeq* (Zhou *et al*., *Bioinformatics*, **27**: 2672-2678, 2011)
- *NOISeq* (Tarazona *et al*., *Genome Res*., **21**: 2213-2223, 2011)
- *PoissonSeq* (Li *et al*., *Biostatistics*, **13**: 523-538, 2012)
- *SAMseq* (Li and Tibshirani, *Stat Methods Med Res*., 2011 Nov 28)
- *BitSeq* (Glaus *et al*., *Bioinformatics*, **28**: 1721-1728, 2012)
- *DEXSeq* (Anders et al., *Genome Res*., **22**: 2008-2017, 2012)
- *ShrinkBayes* (Van DE Wiel *et al*., *Biostatistics*, **14**: 113-128, 2013)
- *sSeq* (Yu *et al*., *Bioinformatics*, **29**: 1275-1282, 2013)
- *TCC* (Sun *et al*., submitted)

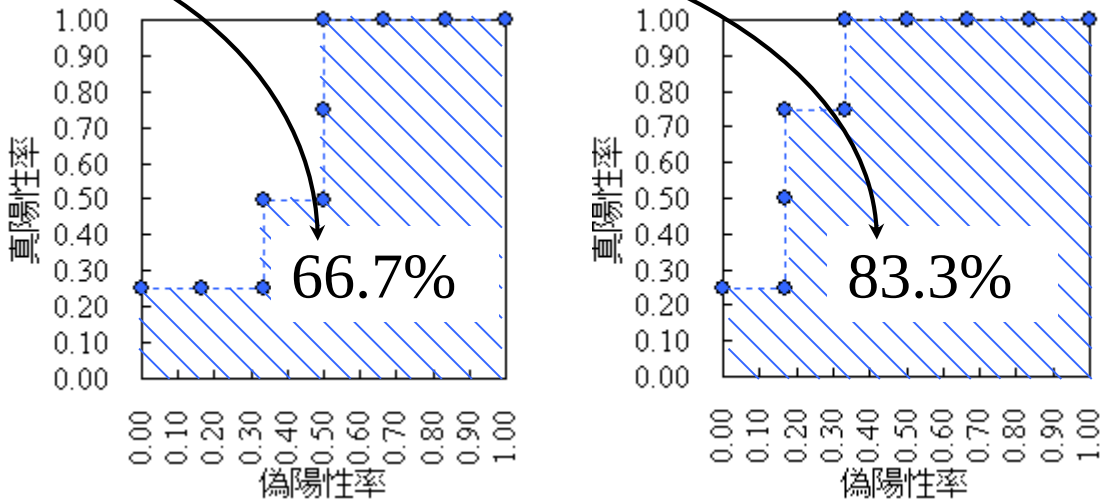
黒字のものたち（+α）の比較結果は…

- *DEGSeq* (Wang *et al*., *Bioinformatics*, **26**: 136-138, 2010)
- *edgeR* (Robinson *et al*., *Bioinformatics*, **26**: 139-140, 2010)
- *GPseq* (Srivastava and Chen, *Nucleic Acids Res*., **38**: e170, 2010)
- *baySeq* (Hardcastle and Kelly, *BMC Bioinformatics*, **11**: 422, 2010)
- *DESeq* (Anders and Huber, *Genome Biol*., **11**: R106, 2010)
- *NBPSeq* (Di *et al*., *SAGMB*, **10**: article24, 2011)
- *TPSM* (Auer and Doerge, *SAGMB*, **10**: article26, 2011)
- *BBSeq* (Zhou *et al*., *Bioinformatics*, **27**: 2672-2678, 2011)
- *NOISeq* (Tarazona *et al*., *Genome Res*., **21**: 2213-2223, 2011)
- *PoissonSeq* (Li *et al*., *Biostatistics*, **13**: 523-538, 2012)
- *SAMseq* (Li and Tibshirani, *Stat Methods Med Res*., 2011 Nov 28)
- *BitSeq* (Glaus *et al*., *Bioinformatics*, **28**: 1721-1728, 2012)
- *DEXSeq* (Anders et al., *Genome Res*., **22**: 2008-2017, 2012)
- *ShrinkBayes* (*ShrinkSeq*; Van DE Wiel *et al*., *Biostatistics*, **14**: 113-128, 2013)
- *sSeq* (Yu *et al*., *Bioinformatics*, **29**: 1275-1282, 2013)
- *TCC* (Sun *et al*., submitted)

よりよい方法とは？

- その方法を用いて発現変動の度合いでランキングしたときに、真の発現変動遺伝子（DEG）がより上位にランキングされる（感度・特異度高い）

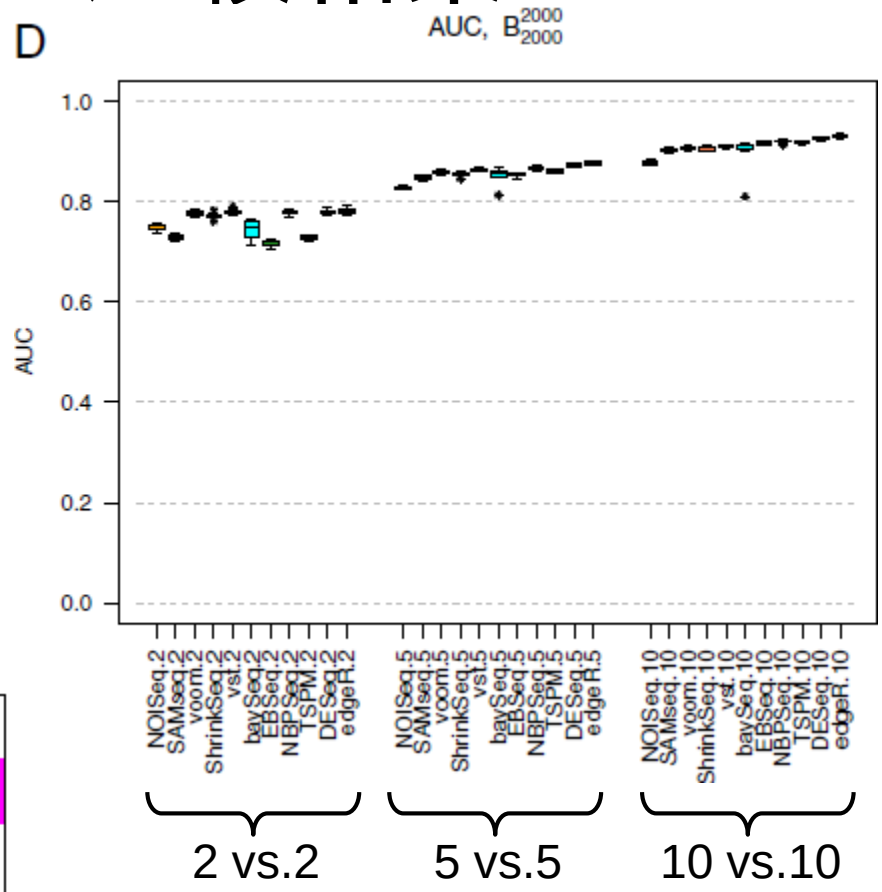
ランキング法				
rank	方法1		方法2	
1	gene8	真	gene8	真
2	gene5	偽	gene5	偽
3	gene4	偽	gene3	真
4	gene3	真	gene2	真
5	gene7	偽	gene7	偽
6	gene1	真	gene1	真
7	gene2	真	gene4	偽
8	gene9	偽	gene9	偽
9	gene10	偽	gene10	偽
10	gene6	偽	gene6	偽

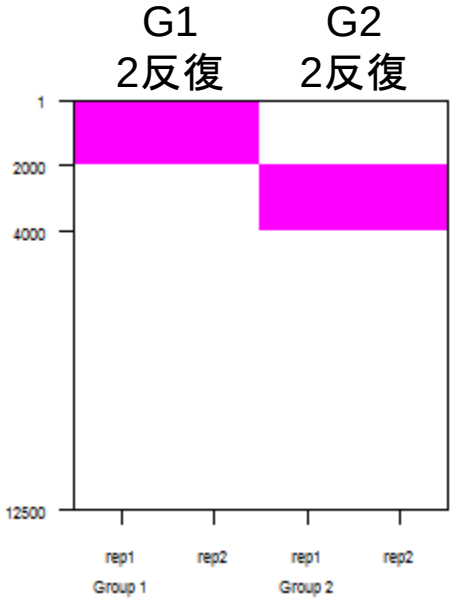


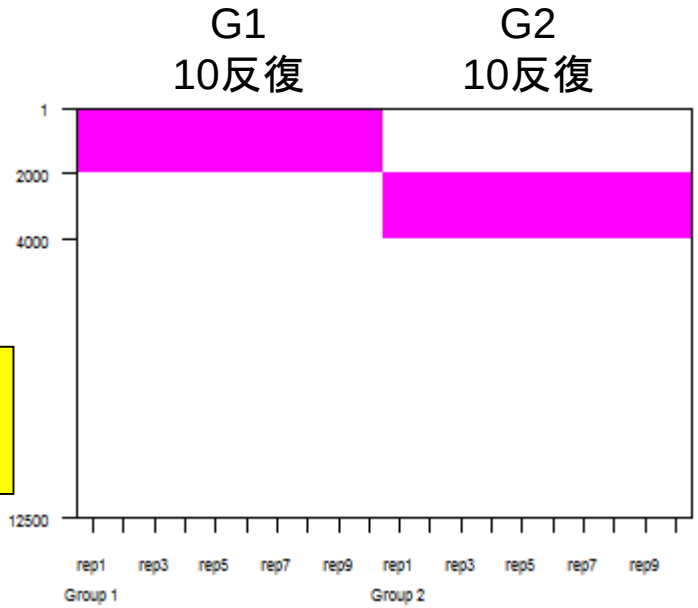
Area Under the ROC Curve (ROC曲線の下部面積：AUC)
バイオインフォマティクス分野でよく用いられる評価基準です

AUC値の比較結果

シミュレーション条件：G1 vs. G2
全遺伝子数：12500
発現変動遺伝子(DEG)数：4000
　G1で高発現：2000
　G2で高発現：2000
unbiased DE situation



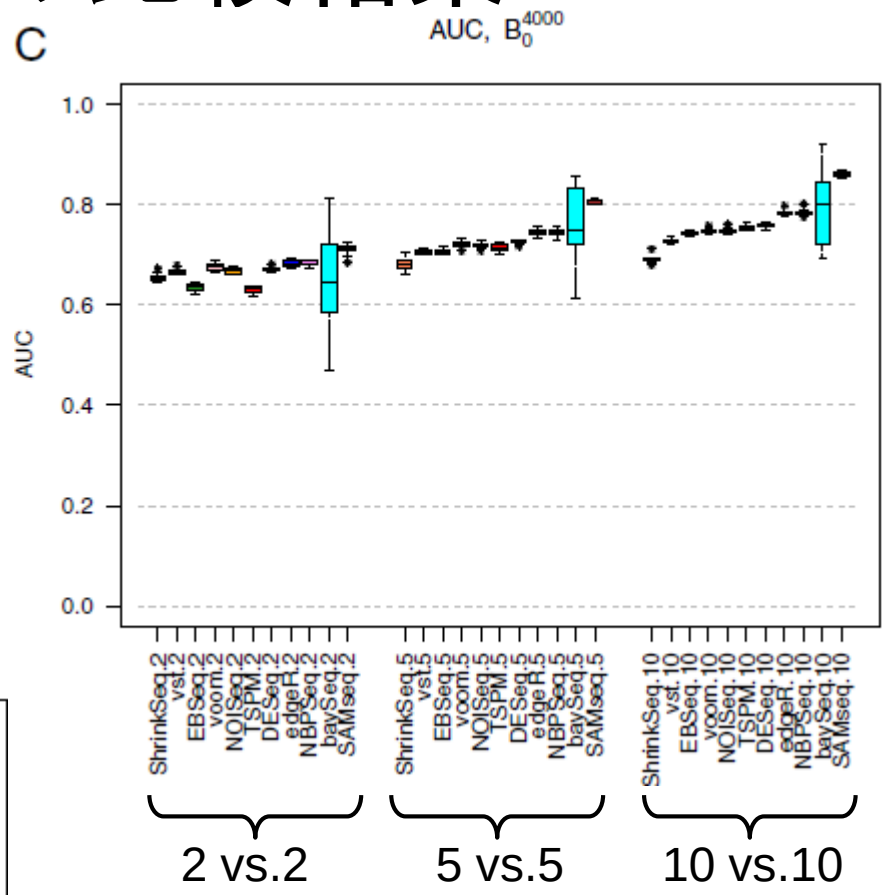


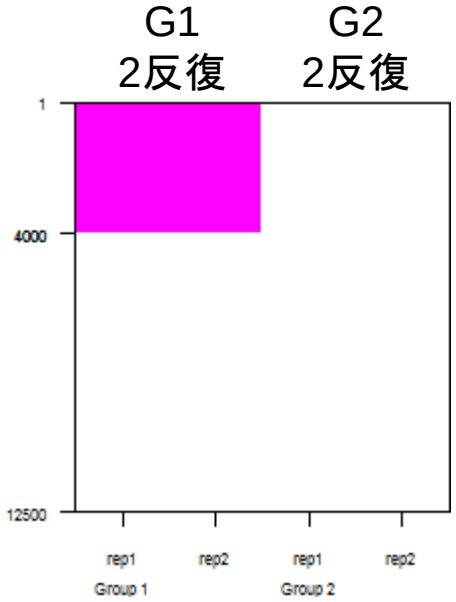


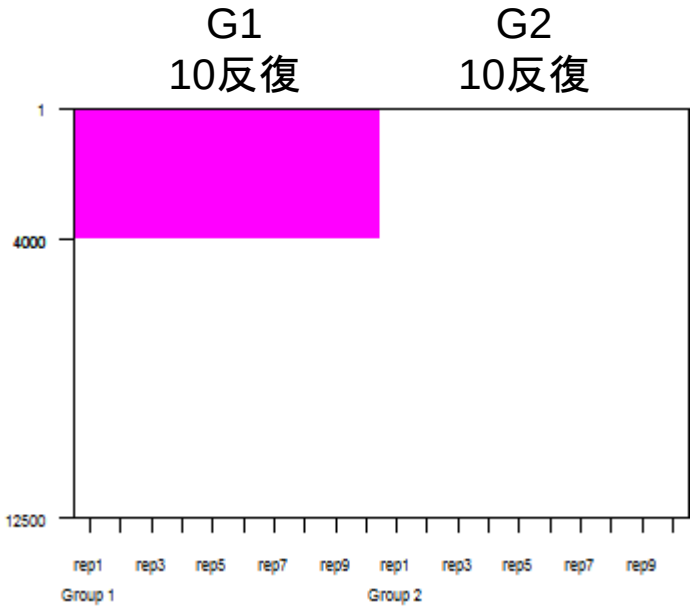
反復実験数を増やすほど精度は上がる
（これが言いたいわけではない…）

AUC値の比較結果

シミュレーション条件：G1 vs. G2
全遺伝子数：12500
発現変動遺伝子(DEG)数：4000
　G1で高発現：4000
　G2で高発現：0
biased DE situation







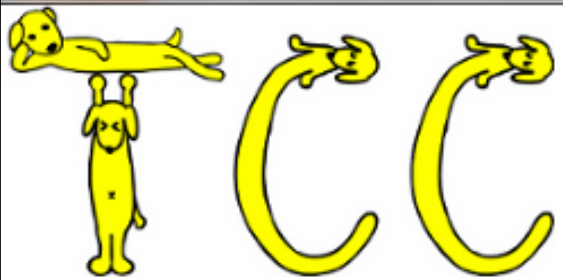
グループ（群）間でDEG数の組成に偏りがあると精度が大幅に低下する

理由：データ正規化法がDEG数の組成に偏りがないことを想定しているため

比較

- *DEGSeq* (
- *edgeR* (Ro
- *GPseq* (Sri
- *baySeq* (H
- *DESeq* (An
- *NBPSeq* (
- *TPSM* (Au
- *BBSeq* (Zh
- *NOISeq* (T
- *PoissonSe*
- *SAMseq* (L
- *BitSeq* (Gla
- *DEXSeq* (A
- *ShrinkBaye*
- *sSeq* (Yu *et al.*, *Bioinformatics*, **29**: 1275-1282, 2013)
- ***TCC* (Sun *et al.*, submitted)**

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



Sun *et al.*, submitted

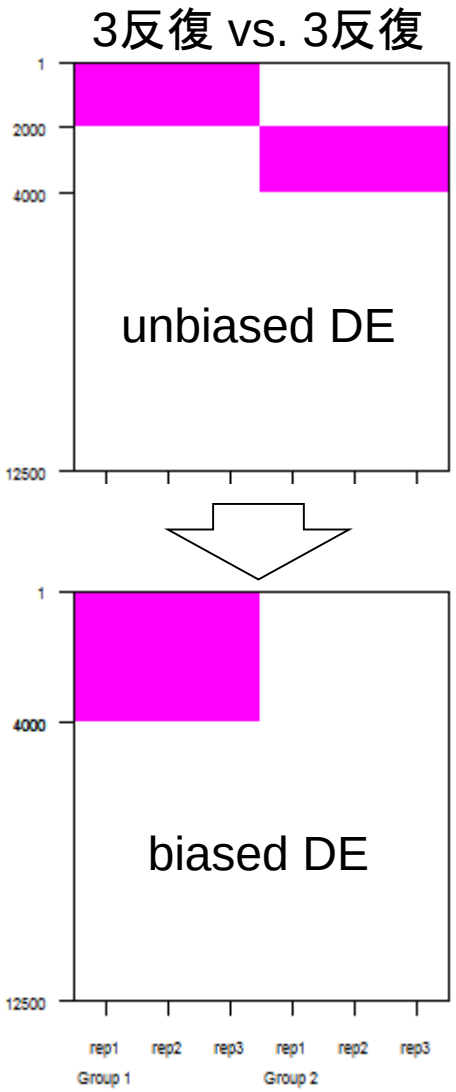
TCC: an R package for comparing tag count data with robust normalization strategies

The R package, **TCC** provides users with a robust and accurate framework to perform differential expression analysis of tag count data. Differential expression analysis of tag count data (such as RNA-seq) from high-throughput sequencing technologies is a fundamental means of studying gene expression. We recently developed a multi-step normalization method (TbT; Kadota et al., 2012) for two-group RNA-seq data with replicates. The strategy is to remove data that are potential differentially expressed genes (DEGs) before performing the data normalization. We demonstrated that the DEG elimination strategy (called DEGES) for data normalization is essential for obtaining a well-ranked gene list in which true DEGs are top-ranked and non-DEGs are bottom ranked. **TCC** provides integrated analysis pipelines with improved data normalization steps, compared with other packages such as edgeR, DESeq, and baySeq, by appropriately combining their functionalities.

Important note! (last modified: May 20, 2013)

While the older version (ver. 1.1.3) of this package is currently available at the CRAN repository, we are now moving it from CRAN to Bioconductor. This webpage is temporal until the next release (perhaps, ver. 1.2.0) of TCC is available upon Bioconductor. The latest version available on this webpage is **ver. 1.1.99**.

AUC値の比較結果（*edgeR* vs. *TCC*）



edgeR 90.84%　　*TCC* 90.83%

edgeR 82.96%　　*TCC* 89.86%

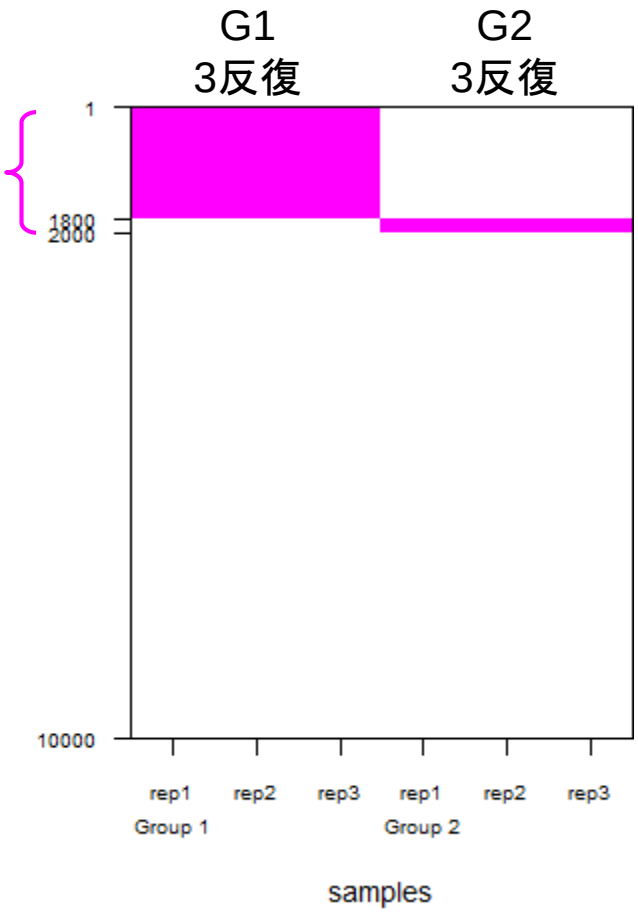
*TCC*パッケージ中のシミュレーション解析でも*edgeR*の性能低下、および*TCC*の頑健性の高さが確認できます

TCCでの解析手順(複製あり;二群間比較)

- 仮想データ(10,000 genes×6 samples)
  - 2,000 DEGs (20%がDEG)
    - Group1 (G1)で高発現:gene1~1800 (**90%**)
    - Group2 (G2)で高発現:gene1801~2000 (**10%**)

data_hypodata_3vs3.txt

	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3
gene_1	36	56	144	2	1	0
gene_2	84	152	124	52	37	28
gene_3	592	840	800	151	257	200
gene_4	0	8	4	1	1	3
...						



(Rで)塩基配列解析(主に次世代シーケンサーのデータ) (last modified 2013/05/23, since 2010)

What's new?
Rパッケージ TCCの最新版は1.1.99です。6月6日に開催されるNAIST植物グローバル教育プロジェクト・平成25年度ワークショップでは、R(ver. 3.0.1)とTCC(ver. 1.1.99)をベースに話をする予定です。Rのインストールと起動を実行したあとにTCC(ver. 1.1.99)のインストールも忘れずに行っておいてください。(2013/05/23)NEW

- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(fixed dispersion) (last modified 2011/10/31)
- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(random dispersion) (last modified 2011/10/31)
- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(tagwise dispersion) (last modified 2012/10/18)
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | について (last modified 2013/05/21) NEW
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製なし | DESeq with iDEGES/DESeq (Sun submitted) (last modified 2013/01/28)**recommended**
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製なし | DESeq (Anders 2010) (last modified 2013/01/...
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR with iDEGES/edgeR (Sun submitted) (last modified 2013/05/22)**recommended**NEW
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR with DEGES/TbT (Kadota 2012) (last ...fied 2013/01/28)
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR coupled with TbT normalization (上と同じで昔の記述) (last modified 2012/09/11)
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | NBPSeq coupled with TbT normalization (Kadota 2012) (last modified 2012/08/24)
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | DESeq coupled with TbT normalization (Kadota 2012) (last modified 2012/06/07)

・解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR with iDEGES/edgeR (Sun_submitted)

iDEGES/edgeR正規化(Sun submitted)を実行したのち、edgeRパッケージ中のexact testで発現変動遺伝子(Differentially expressed Genes; DEGs)検出を行うやり方を示します。全てTCC (≥ ver. 1.1.99)パッケージ内で完結します。
ちなみに、param_DEmethodのところでの"edger"は「DEG検出の手段としてedgeRパッケージ中の方法を利用する」ことに相当しますが、以下に示す他の二つの選択肢もあります:
・DEG検出はDESeqパッケージ中のnegative binomial testにしたい場合:"deseq"
・DEG検出はbaySeqパッケージ中のempirical Bayesにしたい場合:"bayseq"
また、param_FDRでは0.05を指定していますが、これは「**DEGと判定したものの中で**どの程度の数の偽物(本当はDEGではないのにDEGと判定されたもの)を許容するか？」というfalse discovery rate (FDRと略す)の閾値を指定するところです。0.05だと5%偽物が含まれることを許容することに相当し、一般にこの数を低く設定するほどDEGと判定する数が少なくなります。
ちなみによくp値の閾値(有意水準; significance level)と混同しがちですが、例えばp-value < 0.05は「**non-DEGの中で**(本当はDEGではないにもかかわらず)DEGと判定してしまうのが5%程度になるような閾値」に相当します。つまり、「分子は同じですが、分母が異なる」ということです。なぜ、p値を使わずにFDR(p値に対応する正しい用語は本当はq-value)がよく用いられるかというと、例えばある10000遺伝子からなるランダムデータの二群間比較を行って、p < 0.05の閾値を満たすものをDEGと判定するとしましょう。p値の定義からいって、大体5%(10000×0.05 = 500個)程度がDEGと判定されます。この得られた500個が本当のDEG...ではないですよね？ つまり、実データの場合で、「有意水準を満たすDEG数」が「搭載遺伝子数×設定した有意水準」以下であればそのデータセット中にDEGはないという判定がくだされるべき、という常識的な判断を下せなければなりません。逆に、10000遺伝子からなる実データの二群間比較を行って、p < 0.05の閾値を満たすものが1200個あったとしましょう。この中には偶然にDEGと判定されたものが500個程度含まれると考えるべきですが、残りの700個程度は本物と考えていいですよね。言い換えると1200個中(500/1200)×100 = 41.6667%がfalse positiveということであり、「FDR p < 0.416667を満たす遺伝子数は1200個あった」という言い方ができます。実用上は、上記事柄が理解できていれば十分でしょう。
つまり
「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13の10,000 genes×6 samplesの「複製あり」タグカウントデータ(data_hypodata_3vs3.txt)
Biological replicatesを模倣したシミュレーションデータ(G1群3サンプル vs. G2群3サンプル)です。
gene_1～gene_2000までがDEG (最初の1800個がG1群で高発現、残りの200個がG2群で高発現)
gene_2001～gene_10000までがnon-DEGであることが既知です。

```
------     ここから     ------
in_f <- "data_hypodata_3vs3.txt"                    #読み込みたい発現データファイルを指定してin_fに格納
out_f <- "data_hypodata_3vs3_iDEGESedgeR_DE.txt"    #出力ファイル名を指定
param_A <- 3                                        #A群(G1群)のサンプル数を指定
param_B <- 3                                        #B群(G2群)のサンプル数を指定
paramI <- 3                                         #正規化におけるTMM-(edgeR-TMM)nパイプラインのnの値(iterationの回数)を指定(デフォル
param_DEmethod <- "edger"                           #(正規化後に実行するDEG検出法を指定
param_FDR <- 0.05                                   #DEG検出時のfalse discovery rate (FDR)閾値を指定
```

1. サンプルデータ13の10,000 genes×6 samplesの「複製あり」タグカウントデータ(data_hypodata_3vs3.txt)
Biological replicatesを模倣したシミュレーションデータ(G1群3サンプル vs. G2群3サンプル)です。
gene_1～gene_2000までがDEG（最初の1800個がG1群で高発現、残りの200個がG2群で高発現）
gene_2001～gene_10000までがnon-DEGであることが既知です。

```
------     ここから     ------
in_f <- "data_hypodata_3vs3.txt"                    #読み込みたい発現データファイルを指定してin_fに格納
out_f <- "data_hypodata_3vs3_iDEGESedgeR_DE.txt"    #出力ファイル名を指定
param_A <- 3                                        #処群(G1群)のサンプル数を指定
param_B <- 3
paramI <- 3
param_DEmethod <- "edger
param_FDR <- 0.05

#必要なパッケージなどを
library(TCC)

#発現データの読み込みとT
data <- read.table(in_f,
data.cl <- c(rep(1, para
tcc <- new("TCC", data,

#iDEGES/edgeR正規化の実
tcc <- calcNormFactors(t

#DEG検出の実行と結果の抽
tcc <- estimateDE(tcc, t
result <- getResult(tcc,

#結果をまとめたものをファイルに出力
tmp <- cbind(rownames(tcc$count), tcc$c
write.table(tmp, out_f, sep="¥t", appen

#以下は（こんなこともできますという）お
#正規化後のデータでM-A plotを描画
plot(tcc, FDR=param_FDR)

------     ここまで     ------
```

切り取り(T)
コピー(C)
貼り付け
すべて選択(
印刷(I)...
印刷プレビ
Bing でマッ
Bing で翻訳
Google で検
電子メール
すべてのア
OneNote に

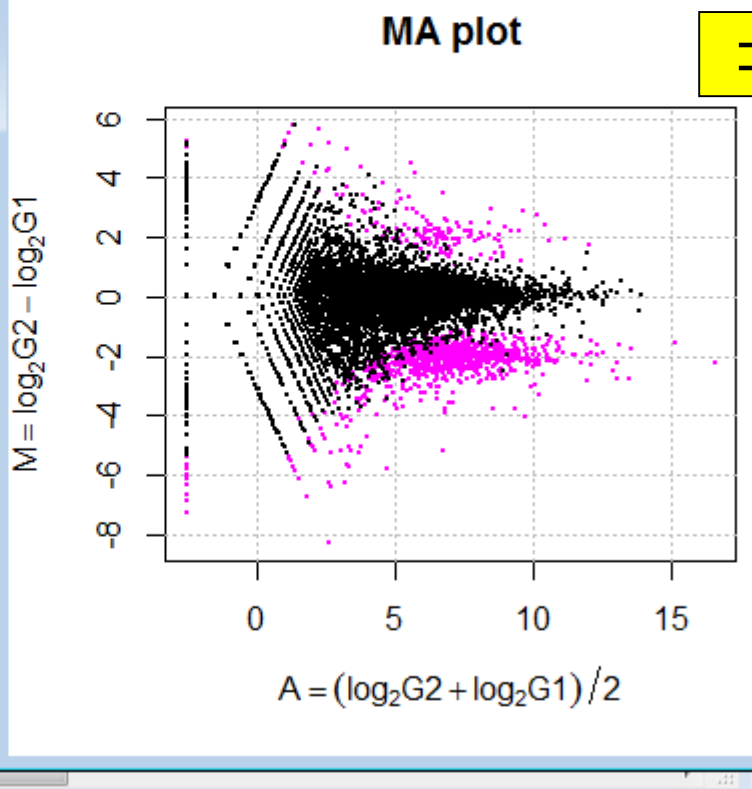
RGui (64-bit)

ファイル　履歴　サイズ変更　ウインドウ

R Console

```
>
> #iDEGES/edgeR正規化の実行
> tcc <- calcNormFactors(tcc, norm
+                                 itei
TCC::INFO: Calculating normalizati
TCC::INFO: (iDEGES pipeline : tmm
TCC::INFO: Done.
>
> #DEG検出の実行と結果の抽出
> tcc <- estimateDE(tcc, test.meth
TCC::INFO: Identifying DE genes us
TCC::INFO: Done.
> result <- getResult(tcc, sort=FA
>
> #結果をまとめたものをファイルに出力
> tmp <- cbind(rownames(tcc$count)
> write.table(tmp, out_f, sep="\t"
>
> #以下は(こんなこともできますという)おまけ
> #正規化後のデータでM-A plotを描画
> plot(tcc, FDR=param_FDR)
>
>
```

R Graphics: Device 2 (ACTIVE)



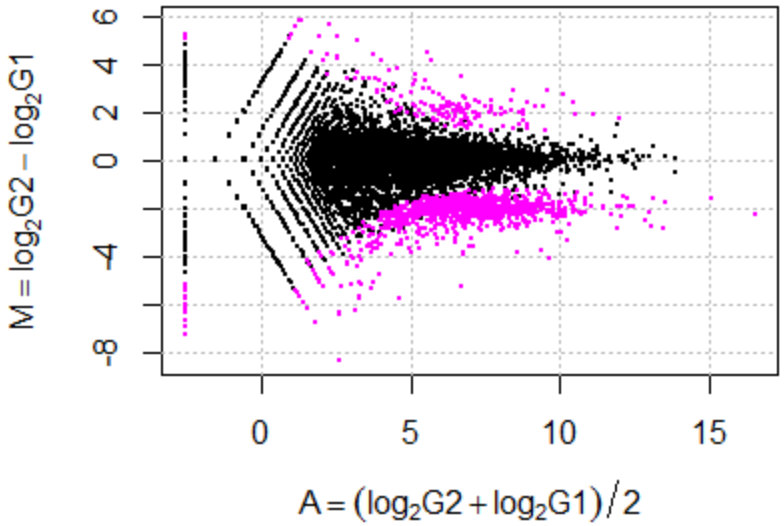
コピペ

出力ファイルの説明

入力データ

p-valueとその順位

rowname	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3	gene_id	a.value	m.value	p.value	q.value	rank	estimatedDEG
gene_1	36	56	144	2	1	0	gene_1	3.1484	-6.262	1.72E-10	1.14E-07	15	1
gene_2	84	152	124	52	37	28	gene_2	6.0969	-1.5883	0.00066	0.00731	900	1
gene_3	592	840	800	151	257	200	gene_3	8.6019	-1.8415	2.91E-06	6.81E-05	427	1
gene_4	0	8	4	1	1	3	gene_4	1.3706	-1.2342	0.4809	1	4523	0
gene_5	32	32	0	1	1	0	gene_5	1.9168	-4.9662	0.00328	0.03098	1060	1
gene_6	4	0	24	4	10	0	gene_6	2.7217	-0.9582	0.55429	1	5047	0
gene_7	344	240	236	76	67	71	gene_7	7.1252	-1.9029	1.52E-06	4.15E-05	367	1
gene_8	1264	784	1060	212	183	179	gene_8	8.7979	-2.4011	5.79E-10	2.26E-07	25	1
gene_9	92	88	84	21	22	33	gene_9	5.5612	-1.7621	0.00037	0.00437	841	1
gene_10	64	48	96	24	13	12	gene_10	5.0718	-2.0502	0.00064	0.00713	898	1
...													



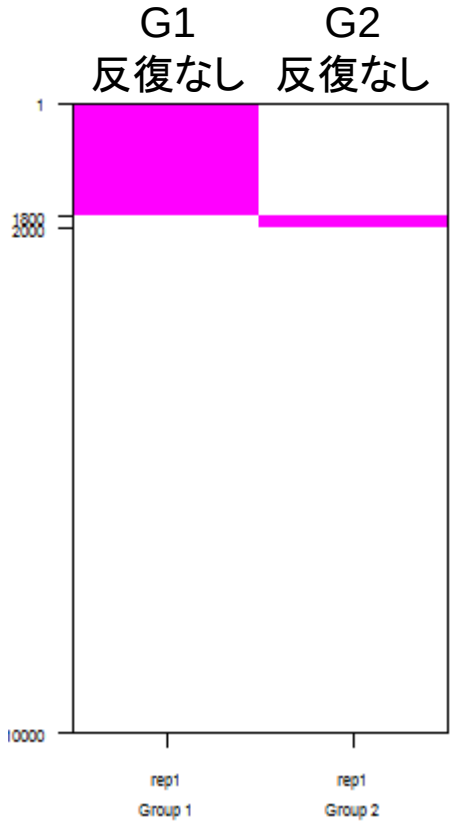
M-A plotのA値とM値

q-value

(param_FDRで指定した)FDR閾値(<0.05)を満たすDEG。*q*-value < 0.05のものが1の値をとる。

TCCでの解析手順(複製なし;二群間比較)

- 仮想データ(10,000 genes×2 samples)
  - 2,000 DEGs (20%がDEG)
    - Group1 (G1)で高発現:gene1~1800 (**90%**)
    - Group2 (G2)で高発現:gene1801~2000 (**10%**)



data_hypodata_1vs1.txt

	G1_rep1	G2_rep1
gene_1	36	2
gene_2	84	52
gene_3	592	151
gene_4	0	1
...		

data_hypodata_3vs3.txt

	G1_rep1	G1_rep2	G1_rep3	G2_rep1	G2_rep2	G2_rep3
gene_1	36	56	144	2	1	0
gene_2	84	152	124	52	37	28
gene_3	592	840	800	151	257	200
gene_4	0	8	4	1	1	3
...						

(Rで)塩基配列解析(主に次世代シーケンサーのデータ) (last modified 2013/05/23, since 2010)

What's new?
Rパッケージ TCCの最新版は1.1.99です。6月6日に開催されるNAIST植物グローバル教育プロジェクト・平成25年度ワークショップでは、R(ver. 3.0.1)とTCC(ver. 1.1.99)をベースに話をする予定です。Rのインストールと起動を実行したあとにTCC(ver. 1.1.99)のインストールも忘れずに行っておいてください。(2013/05/23)NEW

- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(fixed dispersion) (last modified 2011/10/31)
- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(random dispersion) (last modified 2011/10/31)
- 解析 | NGS(RNA-seq) | その他 | 負の二項分布に従うシミュレーションデータを作成する(tagwise dispersion) (last modified 2012/10/18)
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | について (last modified 2013/05/21) NEW
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製なし | DESeq with iDEGES/DESeq (Sun submitted) (last modified 2013/01/28)**recommended**
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製なし | DESeq (Anders 2010) (last modified 2013/01/
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR with iDEGES/edgeR (Sun submitted) (last modified 2013/05/22)**recommended**NEW
- 解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製あり | edgeR with DEGES/TbT (Kadota 2012) (last modified 2013/01/28)

解析 | NGS(RNA-seq) | 発現変動遺伝子 | 二群間 | 複製なし | DESeq with iDEGES/DESeq (Sun_submitted)

iDEGES/DESeq正規化(Sun submitted)を実行したのち、DESeqパッケージ中のnegative binomial testで発現変動遺伝子(Differentially expressed Genes; DEGs)検出を行うやり方を示します。全てTCC (≥ ver. 1.1.99)パッケージ内で完結します。
ちなみに、param_DEmethodのところでの"deseq"は「(正規化後に実行する)DEG検出の手段としてDESeqパッケージ中の方法を利用する」ことに相当しますが、以下に示す別の選択肢もあります:
・DEG検出はbaySeqパッケージ中のempirical Bayesにしたい場合:"bayseq"

「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14の10,000 genes×2 samplesの「複製なし」タグカウントデータ(data_hypodata_1vs1.txt)
シミュレーションデータ(G1群1サンプル vs. G2群1サンプル)です。
gene_1～gene_2000までがDEG (最初の1800個がG1群で高発現、残りの200個がG2群で高発現)
gene_2001～gene_10000までがnon-DEGであることが既知です。

```
------     ここから     ------
in_f <- "data_hypodata_1vs1.txt"                                  #読み込みたい発現データファイルを指定してin_fに格納
out_f <- "data_hypodata_1vs1_iDEGESDESeq_DE.txt"                  #出力ファイル名を指定
param_A <- 1                                                      #A群(G1群)のサンプル数を指定
param_B <- 1                                                      #B群(G2群)のサンプル数を指定
paramI <- 3                                                       #DESeq-(DESeq-DESeq)nパイプラインのnの値(iterationの回数)を指
param_DEmethod <- "deseq"                                         #(正規化後に実行する) DEG検出法を指定
param_FDR <- 0.05                                                 #DEG検出時のfalse discovery rate (FDR)閾値を指定

#必要なパッケージなどをロード
library(TCC)                                                      #パッケージの読み込み

#発現データの読み込みとTCCクラスオブジェクトの作成
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")#発現データファイルの読み込み
data.cl <- c(rep(1, param_A), rep(2, param_B))                    #A群を1、B群を2としたベクトルdata.clを作成
tcc <- new("TCC", data, data.cl)                                  #TCCクラスオブジェクトtccを作成

#iDEGES/DESeq正規化の実行
tcc <- calcNormFactors(tcc, norm.method="deseq", test.method="deseq",#正規化を実行した結果をtccに格納
                       iteration=paramI, FDR=0.1, floorPDEG=0.05)#正規化を実行した結果をtccに格納

#DEG検出の実行と結果の抽出
tcc <- estimateDE(tcc, test.method=param_DEmethod, FDR=param_FDR)#DEG検出を実行した結果をtccに格納
```

TCC: an R package for comparing tag count data with robu strategies

The R package, **TCC** provides users with a robust and accurate framework to perform d tag count data. Differential expression analysis of tag count data (such as RNA-seq) fro technologies is a fundamental means of studying gene expression. We recently develop method (TbT; Kadota et al., 2012) for two-group RNA-seq data with replicates. The stra potential differentially expressed genes (DEGs) before performing the data normalizatio elimination strategy (called DEGES) for data normalization is essential for obtaining a w DEGs are top-ranked and non-DEGs are bottom ranked. **TCC** provides integrated analy normalization steps, compared with other packages such as edgeR, DESeq, and baySe functionalities.

Important note! (last modified: May 22, 2013)

While the older version (ver. 1.1.3) of this package is currently available at the CRAN re from CRAN to Bioconductor. This webpage is temporal until the next release (perhaps, Bioconductor. The latest version available on this webpage is **ver. 1.1.99**.

Installation

To install the latest version (ver. 1.1.99) of this package, download the source file and e starting R:

```
install.packages("TCC_1.1.99.tar.gz", repos = NULL, type = "source")
```

Note that you need to enter the following commands if those packages have not been ir

```
source("http://bioconductor.org/biocLite.R")
biocLite(c("edgeR", "baySeq", "DESeq", "ROC"))
```

Docum

User's Guide (vignette) | R script | Manual

Jun 6 2013

Contents

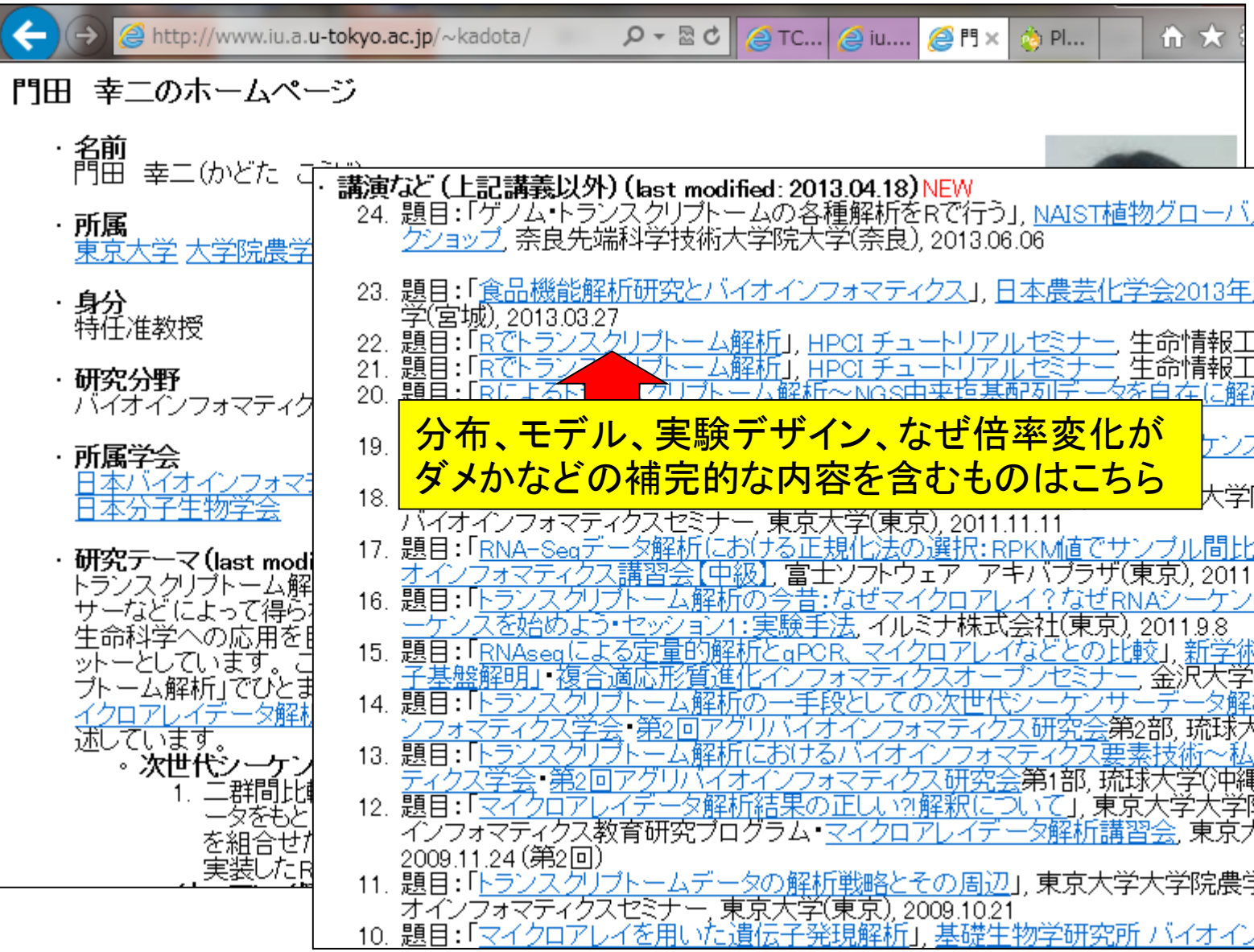
二群間比較以外にも対応しています

まとめ

Rでいろいろできます

- Rでゲノム解析
  - アノテーションファイル（行列形式）からの情報抽出
    - 条件を満たす行のみの抽出。ハッシュとかgrep周辺がRで可能
  - multi-fastaファイルからの情報抽出
    - 一定の長さ以上のものを抽出。コンティグごとのGC含量計算。
- Rで（比較）トランスクリプトーム解析
  - 研究目的別留意点（サンプル内比較、サンプル間比較）
  - リードマッピング　→　数値化（カウントデータの取得）
  - 発現変動解析
    - 既存のRパッケージの弱点（unbiased DE：〇、biased DE：×）
    - *TCC*パッケージを用いて二群間比較（複製あり、複製なし）

スライドPDFはウェブから取得可能です



門田　幸二のホームページ

・**名前**
門田　幸二(かどた　こうじ)

・**所属**
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

・**身分**
特任准教授

・**研究分野**
バイオインフォマティクス、特にトランスクリプトーム解析

・**所属学会**
日本バイオインフォマティクス学会
日本分子生物学会

・**研究テーマ(last modified: 2013.05.23)**
トランスクリプトーム解析手法の開発。本ユニットでは、マイクロアレイ、一次元
サーなどによって得られる 様々なトランスクリプトームデータの解析や新規解析
生命科学への応用を目指します。「数式を並べ立てた難解な方法を凌駕する"
ットーとしています。これまでの主な研究成果を三つのカテゴリーで分けていま
プトーム解析」でひとまとめにできます。また、実験系の方でも気軽に研究成果
イクロアレイデータ解析」と「(Rで)塩基配列解析」上にも下記開発手法中の一部
述しています。
  - **次世代シーケンサー(NGS)解析関連:**
    1. 二群間比較用データ正規化法。RNA-seqに代表されるダイナミック
    ータをもとに二群間比較を行うときに、既存のRパッケージ中のデフ
    を組合せたほうがよりよい結果が得られることを示した(**TbT**: Kadot
    実装したRパッケージTCC。ver. 1.1.99が最新です！)

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

Agricultural Bioinformatics Research Unit

受講生の方へ　研究者の方へ

ホーム > 教育プログラム > 各講義のページ > 11.農学生命情報科学特論I

11.農学生命情報科学特論I

概要

次世代シーケンサーの普及により、以前は主にゲノム解析系で必要とされていた（塩基）配列解析のためのスキルがトランスクリプトーム解析においても要求される時代になりつつあります。本科目では、様々な局面で応用可能な配列解析系のスキルアップを目指し、RNAシークエンス（RNA-Seq）に基づく（非モデル生物の）トランスクリプトーム解析を題材とした実習を含む講義を行います。

担当教員

門田幸二 （東大・農・アグリバイオ
西 達也 （株式会社ジナリス、東大・農・アグリバイオ / 特任教授）

より詳細はこの講義科目で話します。外部の方も受講可能です。

お知らせ

講義日程が6月26日から6月27日なっています。ご注意ください。（平成25年4月5日更新）。
6/27, 7/3, 7/4の講義では、Rの様々なパッケージを利用します。持ち込み用PC利用希望者はRのインストールと起動を参考にして必要なパッケージをインストールしておいてください。

講義日程（平成25年度）

1. 平成25年06月19日17:15-20:30 (PC不使用) **講師**：西達也
2. 平成25年06月27日17:15-20:30 (PC使用) **講師**：門田幸二 **日程変更しました！**
3. 平成25年07月03日17:15-20:30 (PC使用) **講師**：門田幸二
4. 平成25年07月04日17:15-20:30 (PC使用) **講師**：門田幸二

謝辞

共同研究者

清水 謙多郎　先生（東京大学・大学院農学生命科学研究科）

西山 智明　先生（金沢大学・学際科学実験センター）

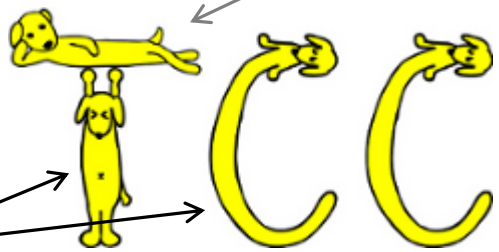
孫 建強　氏（東京大学・大学院農学生命科学研究科・大学院生）

グラント

- 基盤研究(C)（H24-26年度）:「シークエンスに基づく比較トランスクリプトーム解析のためのガイドライン構築」(代表)
- 新学術領域研究(研究領域提案型)(H22年度-):「非モデル生物におけるゲノム解析法の確立」(分担;研究代表者:西山智明)

挿絵やTCCのロゴなど

（妻の）門田 雅世さま作



（有能な秘書の）三浦 文さま作