

(Rで)塩基配列解析 基本的な利用法 Macintosh版

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/>

到達目標:このスライドに書かれている程度のことは自在にできるようにしてエラーへの対処法を身につける。

1. 必要なパッケージのインストールが正しくできているかどうかの自力での判定、および個別のパッケージのインストール
2. 作業ディレクトリの変更
3. テキストエディタで自在に入出力ファイル名の変更(どんなファイル名のものがどこに生成されるかという全体像の把握)
4. 「ありがちなミス」のところで示しているエラーメッセージとその原因をきっちり理解

前提条件

インストールができていても、実際にはできていなかったという事例が散見されます。パッケージ名のスペルミスもよく見受けられます。いくつかのパッケージについて適切にインストールされているか確認しておきましょう。

(Rで)塩基配列解析

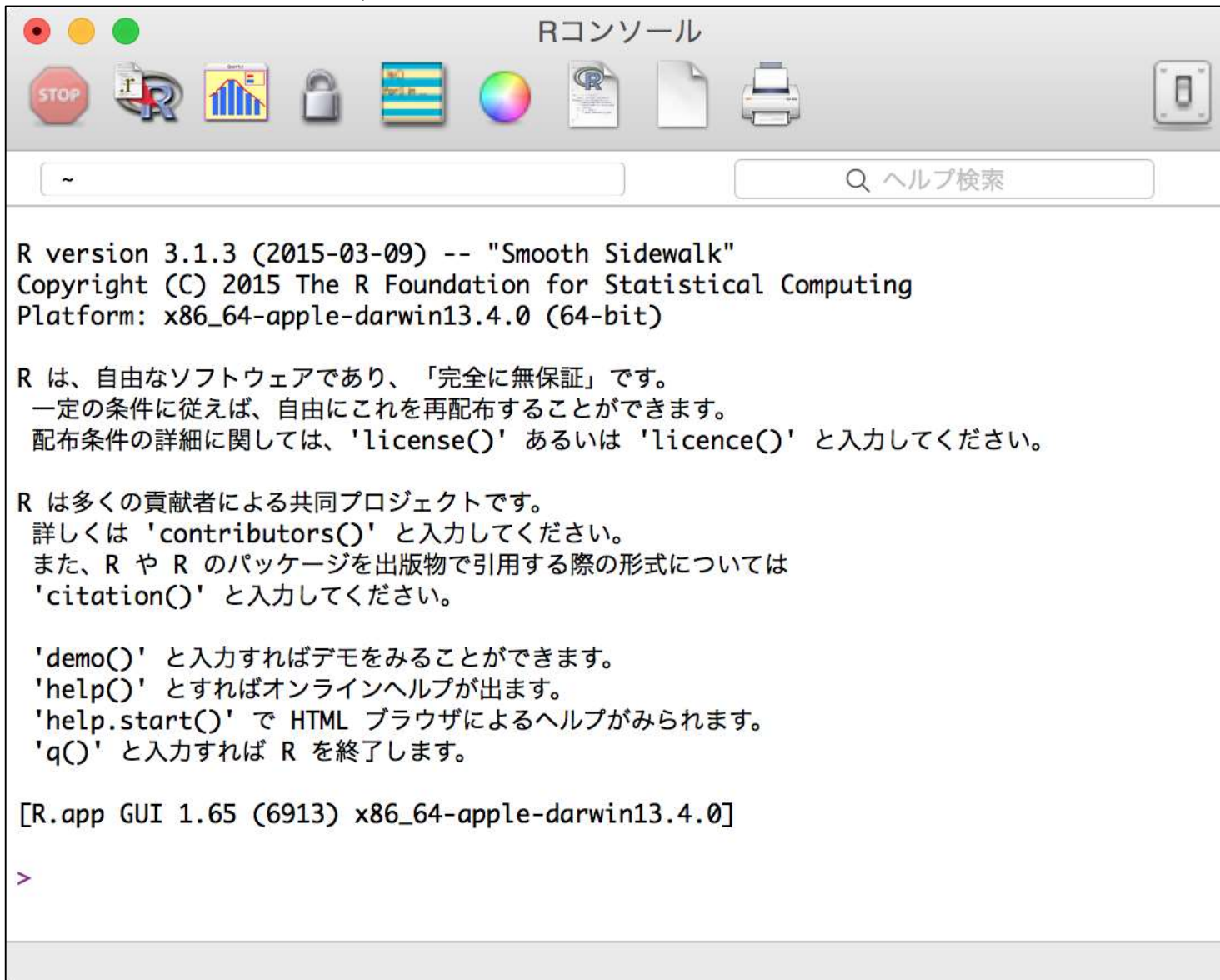
～NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～
(last modified 2015/04/03, since 2010)

What's new?

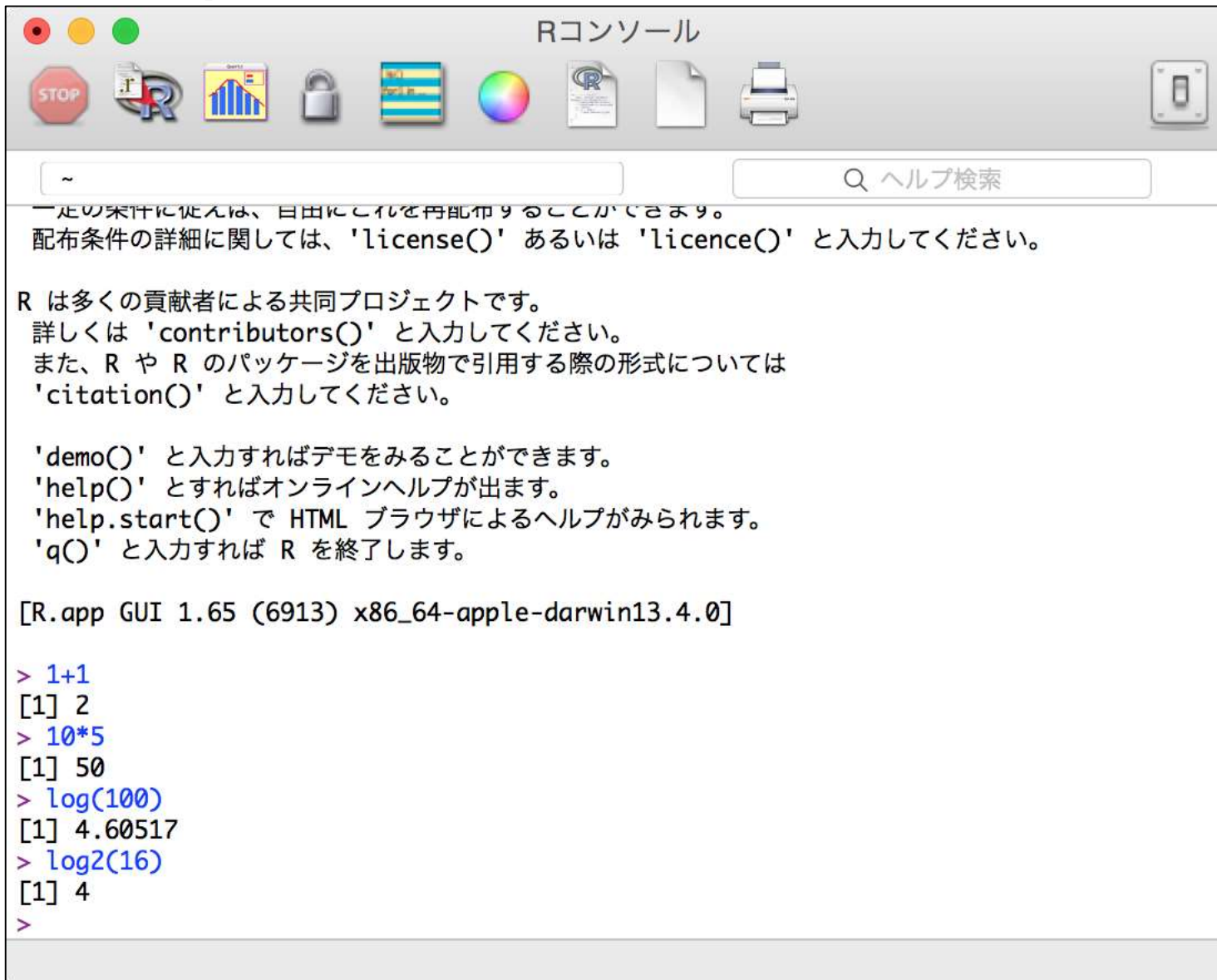
- このウェブページは [インストール | についての推奨手順 \(Windows2015.04.01版とMacintosh2015.04.02版\)](#) に従ってフリーソフトRと必要なパッケージをインストール済みであるという前提で記述しています。初心者の方は [基本的な利用法 \(Windows2015.04.03版とMacintosh2015.04.03版\)](#) で自習してください。本ウェブページを体系的にまとめた [書籍](#) もあります。(2015/04/03) **NEW**
 - 私の所属する [アグリバイオインフォマティクス教育研究プログラム](#) では、平成27年度もバイオインフォ関連講義を行います。例年東大以外の企業の方、研究員、学生が2-3割程度受講しております。受講ガイダンスは4月6日17:15- 於東大農です。(2015/03/31) **NEW**
 - R本体およびパッケージのインストール手順のところを更新しました。詳細は [インストール | について](#) をご覧ください。(2015/04/02) **NEW**
 - [MBCluster.Seq](#) パッケージを用いた遺伝子間クラスタリングのやり方を一通り示しました。(2015/03/14) **NEW**
 - [参考資料 \(講義、講習会、本など\)](#) の項目を更新しました。(2015/03/09) **NEW**
-
- [はじめに](#) (last modified 2015/03/31) **NEW**
 - [参考資料 \(講義、講習会、本など\)](#) (last modified 2015/03/09) **NEW**
 - [過去のお知らせ](#) (last modified 2015/03/31) **NEW**
 - [インストール | について](#) (last modified 2015/04/02) **NEW**
 - [インストール | R本体 | 最新版 | Win用](#) (last modified 2015/03/22) 推奨 **NEW**

[トップページへ](#)

Rの起動



基本的な利用法



Rコンソール

~ ヘルプ検索

一定の条件に従えば、自由にこれを再配布することができます。
配布条件の詳細に関しては、`'license()'` あるいは `'licence()'` と入力してください。

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

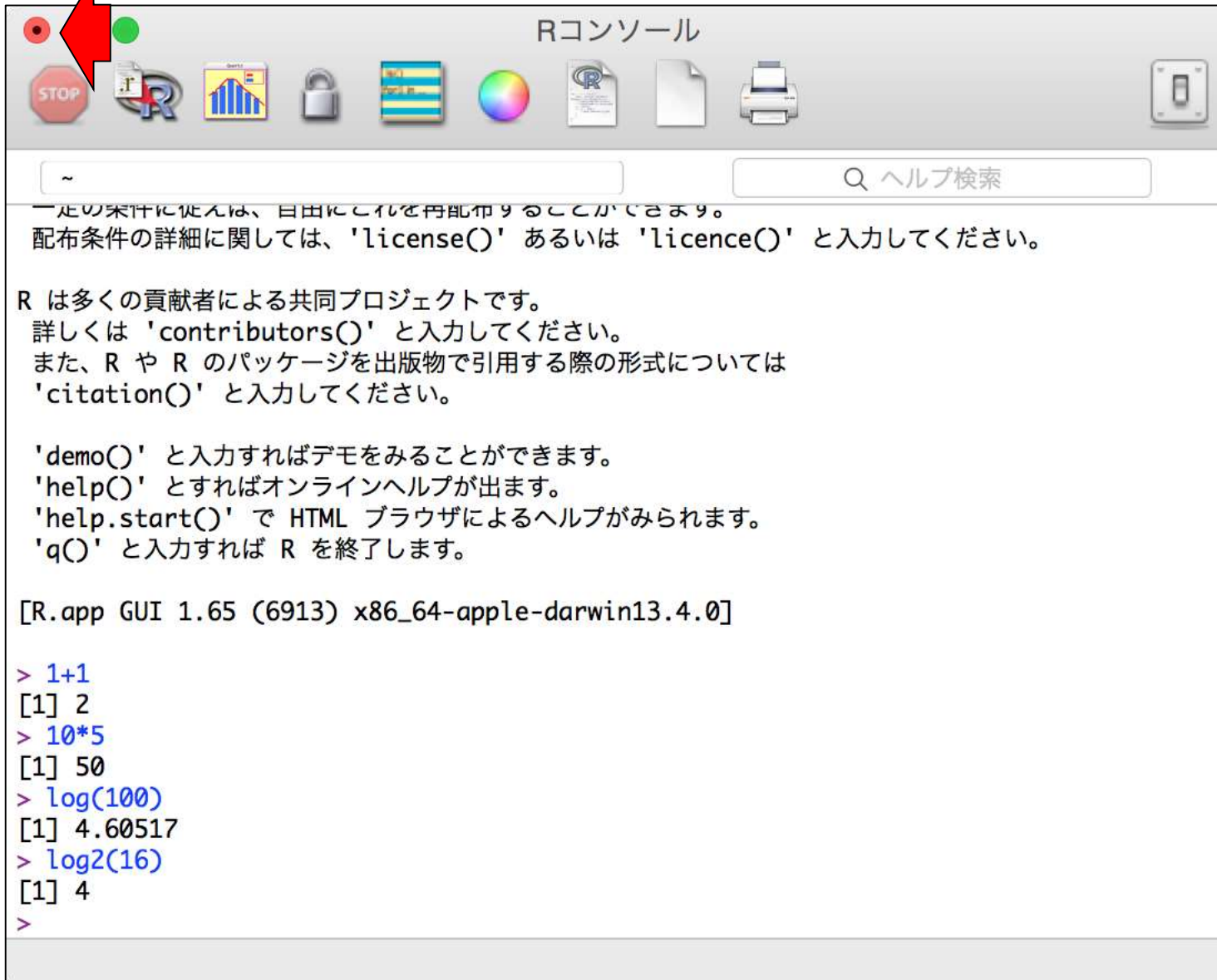
`'demo()'` と入力すればデモをみることができます。
`'help()'` とすればオンラインヘルプが出ます。
`'help.start()'` で HTML ブラウザによるヘルプがみられます。
`'q()'` と入力すれば R を終了します。

[R.app GUI 1.65 (6913) x86_64-apple-darwin13.4.0]

```
> 1+1
[1] 2
> 10*5
[1] 50
> log(100)
[1] 4.60517
> log2(16)
[1] 4
>
```

通常のソフトウェアと同様、左上の赤丸ボタンを押せばよい。

Rの終了



一定の条件に従えば、自由にこれを再配布することができます。
配布条件の詳細に関しては、'license()' あるいは 'licence()' と入力してください。

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

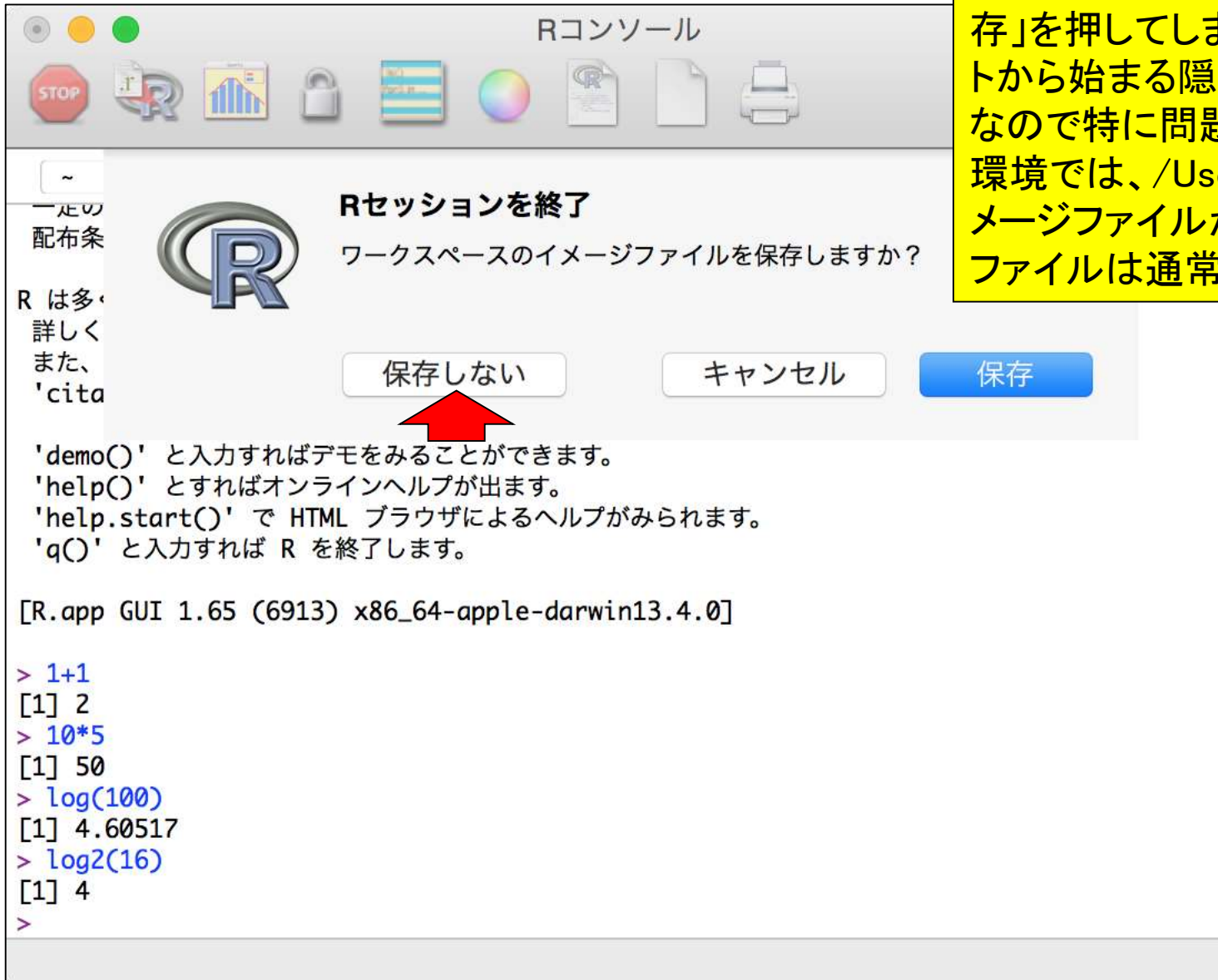
'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。

```
[R.app GUI 1.65 (6913) x86_64-apple-darwin13.4.0]

> 1+1
[1] 2
> 10*5
[1] 50
> log(100)
[1] 4.60517
> log2(16)
[1] 4
>
```

Rの終了

「ワークスペースのイメージファイルを保存しますか?」というダイアログが出るが、最初のうちは「保存しない」でよい。(間違っても「保存」を押してしまっても.Rapp.historyというドットから始まる隠しファイルが作成されるだけなので特に問題はない。ユーザ名kadotaの環境では、/Users/kadota/.Rapp.historyにイメージファイルが自動作成される。尚、隠しファイルは通常は表示されない。



Rコンソール

Rセッションを終了
ワークスペースのイメージファイルを保存しますか?

保存しない キャンセル 保存

一定の配布条
R は多、詳しくまた、'cita
'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。

[R.app GUI 1.65 (6913) x86_64-apple-darwin13.4.0]

```
> 1+1  
[1] 2  
> 10*5  
[1] 50  
> log(100)  
[1] 4.60517  
> log2(16)  
[1] 4  
>
```


塩基配列を入力として、その翻訳されたアミノ酸配列を取得することができます

解析基礎1: 翻訳配列取得

- ・ [イントロ](#) | [一般](#) | [任意の位置の塩基を置換](#) (last modified 2013/09/12)
- ・ [イントロ](#) | [一般](#) | [指定した範囲の配列を取得](#) (last modified 2014/03/08)
- ・ [イントロ](#) | [一般](#) | [指定したID\(染色体やdescription\)の配列を取得](#) (last modified 2014/03/10)
- ・ [イントロ](#) | [一般](#) | [翻訳配列\(translate\)を取得\(基礎\)](#) | [Biostrings](#) (last modified 2015/03/09) **NEW**
- ・ [イントロ](#) | [一般](#) | [翻訳配列\(translate\)を取得\(応用\)](#) | [seqinr\(Chapman 2005\)](#) (last modified 2015/03/09)
- ・ [イントロ](#) | [一般](#) | [相補鎖\(complement\)を取得](#) (last modified 2013/06/14)
- ・ [イントロ](#) | [一般](#) | [逆相補鎖\(reverse complement\)を取得](#)
- ・ [イントロ](#) | [一般](#) | [逆鎖\(reverse\)を取得](#) (last modified 2013/06/14)
- ・ [イントロ](#) | [一般](#) | [2連続塩基の出現頻度情報を取得](#) (last modified 2013/06/14)
- ・ [イントロ](#) | [一般](#) | [3連続塩基の出現頻度情報を取得](#) (last modified 2013/06/14)
- ・ [イントロ](#) | [一般](#) | [任意の長さの連続塩基の出現頻度情報を取得](#) (last modified 2013/06/14)
- ・ [イントロ](#) | [一般](#) | [Tips](#) | [任意の拡張子でファイルを保存](#)
- ・ [イントロ](#) | [一般](#) | [Tips](#) | [拡張子は同じで任意の文字を指定](#)
- ・ [イントロ](#) | [一般](#) | [配列取得](#) | [ゲノム配列](#) | [公共DBから](#)

イントロ | 一般 | 翻訳配列(translate)を取得(基礎) | Biostrings **NEW**

[Biostrings](#)パッケージを用いて塩基配列を読み込んでアミノ酸配列に翻訳するやり方を示します。翻訳のための遺伝コード(genetic code)は、Standard Genetic Codeだそうです。もちろん生物種?!!によって多少違い(variants)があるようで、"Standard", "SGC0", "Vertebrate Mitochondrial", "SGC1"などいろいろ選べるようです。「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ー。

1. FASTA形式ファイル(sample1.fasta)の場合:

multi-FASTAではないsingle-FASTA形式ファイルです。

```
in_f <- "sample1.fasta" #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.fasta" #出力ファイル名を指定してout_fに格納

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

#入力ファイルの読み込み
fasta <- readDNAStringSet(in_f, format="fasta") #in_fで指定したファイルの読み込み
fasta #確認してるだけです

#本番
fasta <- translate(fasta) #アミノ酸配列に翻訳した結果をfastaに格納
fasta #確認してるだけです

#ファイルに保存
writeXStringSet(fasta, file=out_f, format="fasta", width=50) #fastaの中身を指定したフ
```


ファイルの保存

①解析したいファイル sample1.fastaをhogeフォルダ中に保存。②勝手に.txtという拡張子が追加されてしまいますので、戻しておきましょう。

1. FASTA形式ファイル(sample1.fasta)の場合:

multi-FASTAではないsingle-FASTA形式

```
in_f <- "sample1.fasta"
out_f <- "hoge1.fasta"
```

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

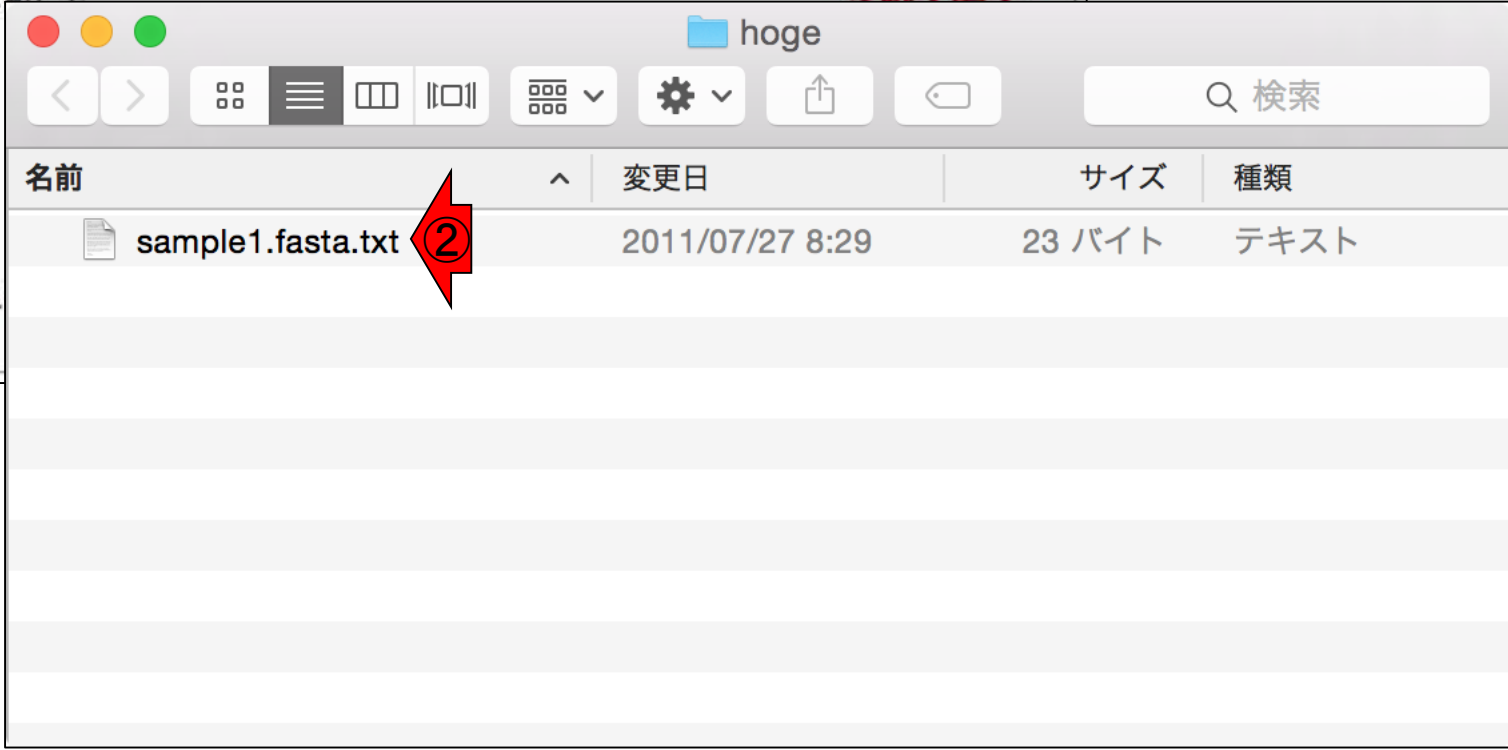
```
#入力ファイルの読み込み
fasta <- readDNASTringSet(in_f,
                           fasta)
```

```
#本番
fasta <- translate(fasta)
fasta
```

```
#ファイルに保存
writeXStringSet(fasta, file)
```

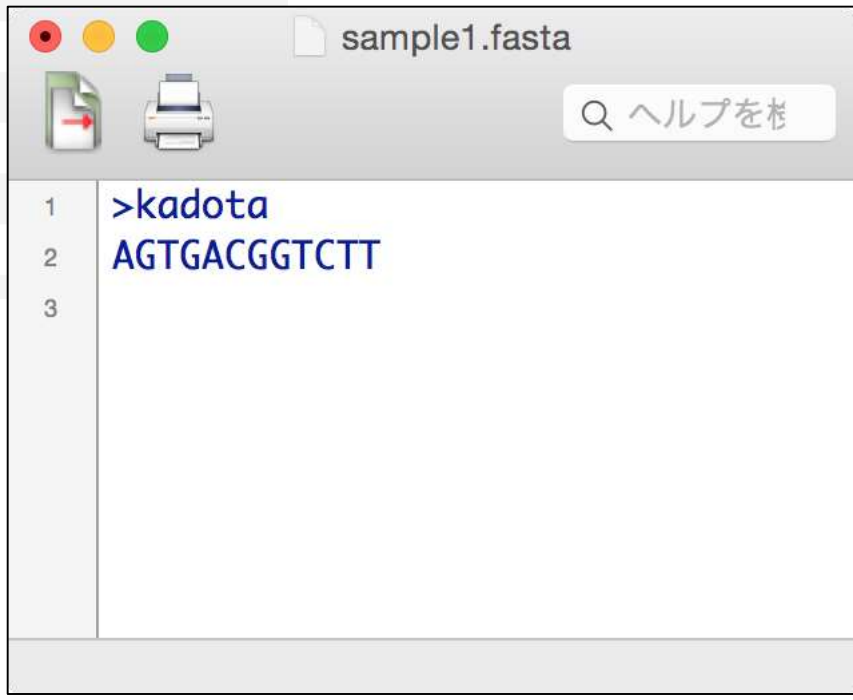
- リンクを新規タブで開く
- リンクを新規ウインドウで開く
- リンク先のファイルをダウンロード
- リンク先のファイルを別名でダウンロード...
- リンクをブックマークに追加...
- リンクをリーディングリストに追加
- リンクをコピー

格納
格納
の読み込み



ファイルの保存

基本Rで取り扱うので、エディタもRのものを利用したほうが無難です。



作業ディレクトリの変更

R起動直後のデフォルトの作業ディレクトリは、ユーザ名kadotaの環境では、「/Users/kadota」です。その一方で、今解析したいファイルはデスクトップ上にあるhogeなので、作業ディレクトリをそこに変更する必要があります。「getwd()」は、現在の作業ディレクトリを表示させるコマンドです。

Rコンソール

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

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

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

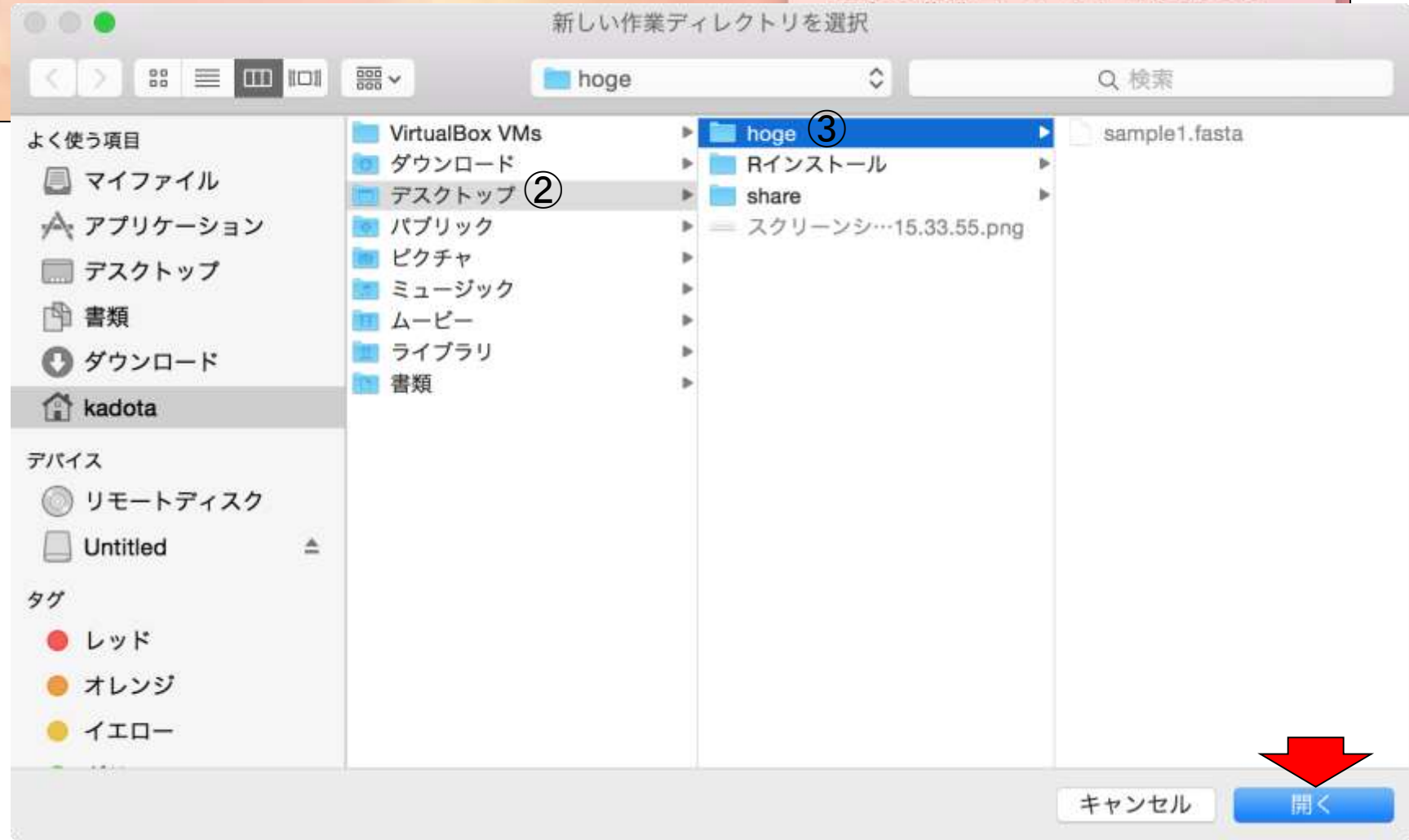
[R.app GUI 1.65 (6913) x86_64-apple-darwin13.4.0]

[履歴が次のファイルから読み込まれました /Users/kadota/.Rapp.history]

```
> getwd()
[1] "/Users/kadota"
> |
```

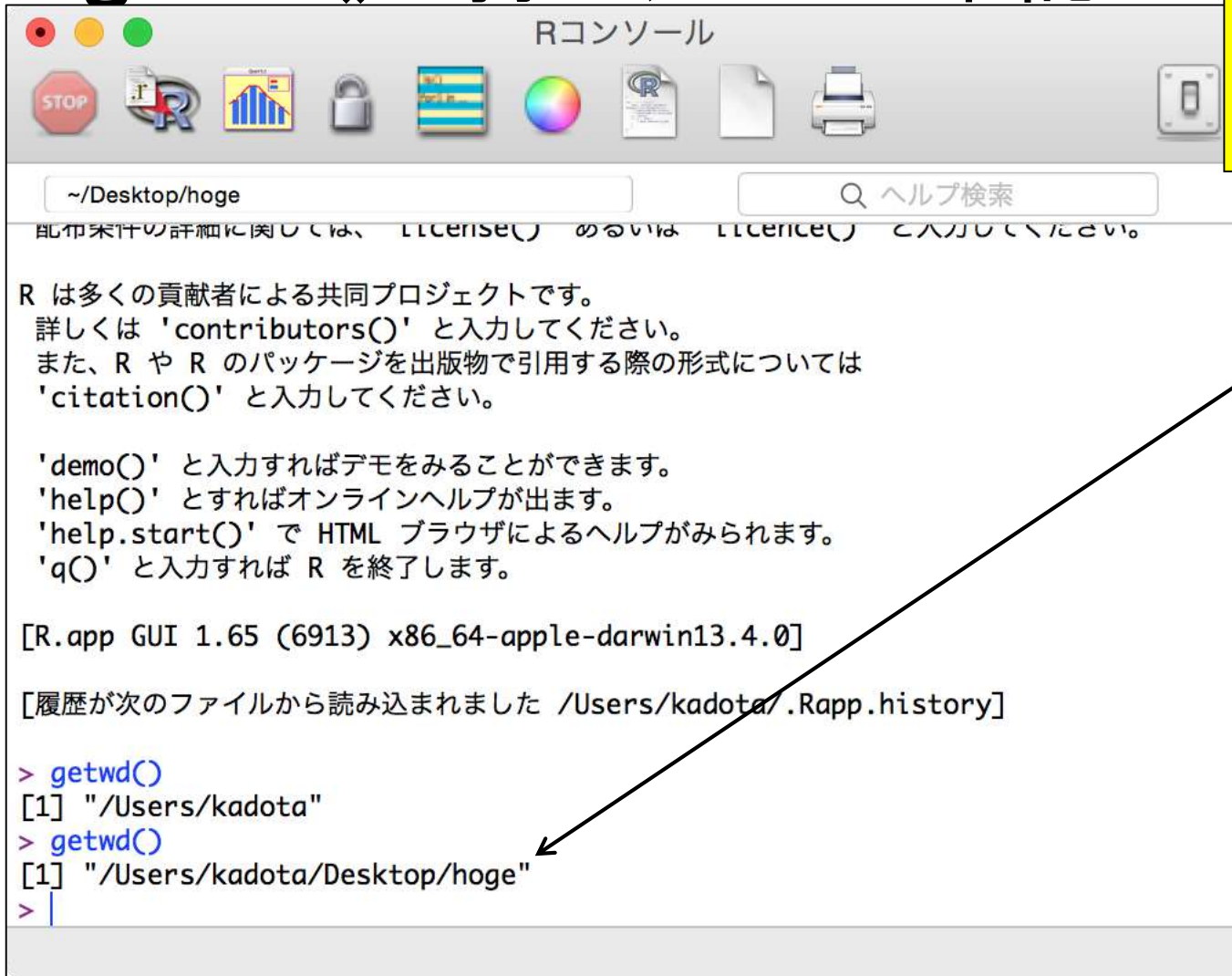
デスクトップのhogeを指定して「開く」を押す。

作業ディレクトリの変更



getwd()と打ち込んで確認

当たり前ですが、解析したいディレクトリ(またはフォルダ)を正しく指定できていなければエラーに遭遇します。また、解析したいファイルが存在しない状態でもエラーが出ます。



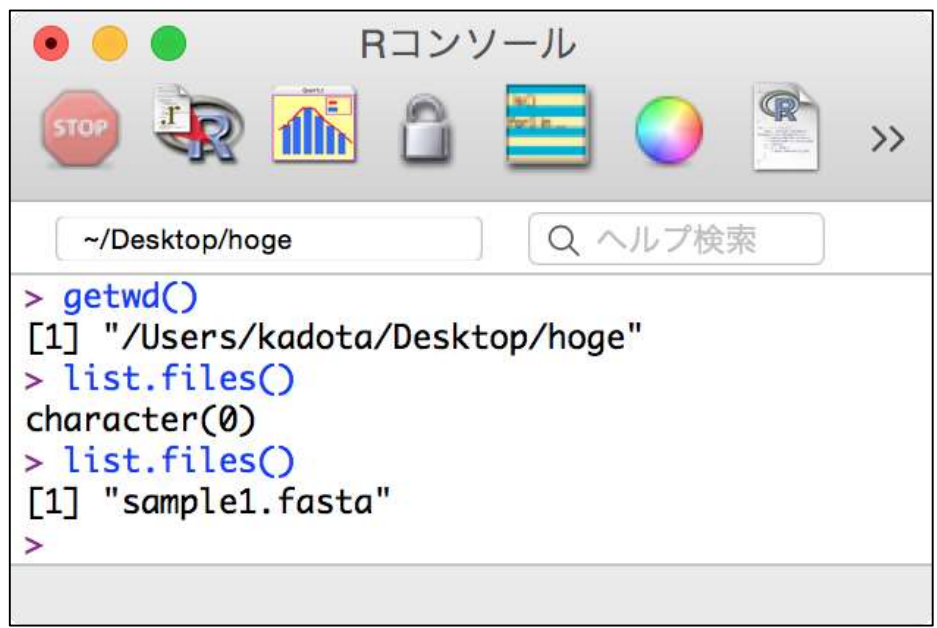
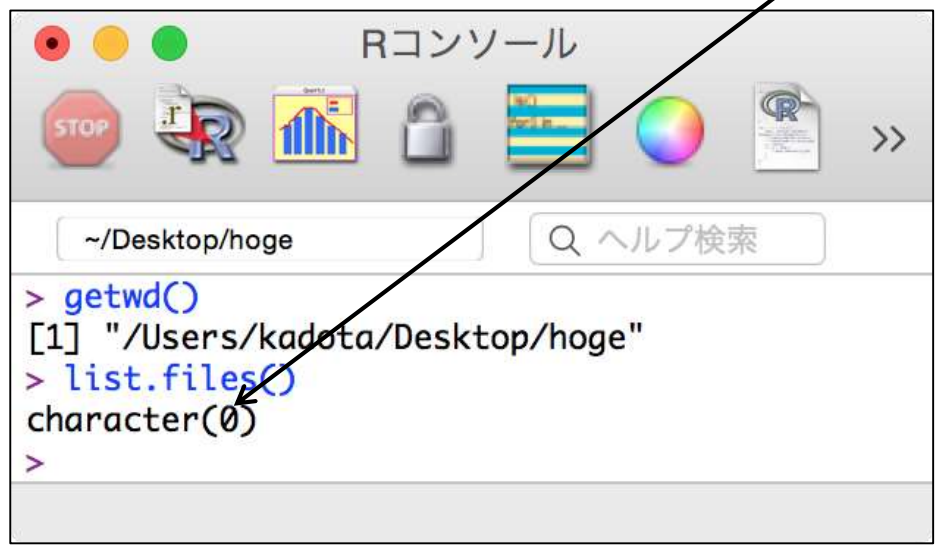
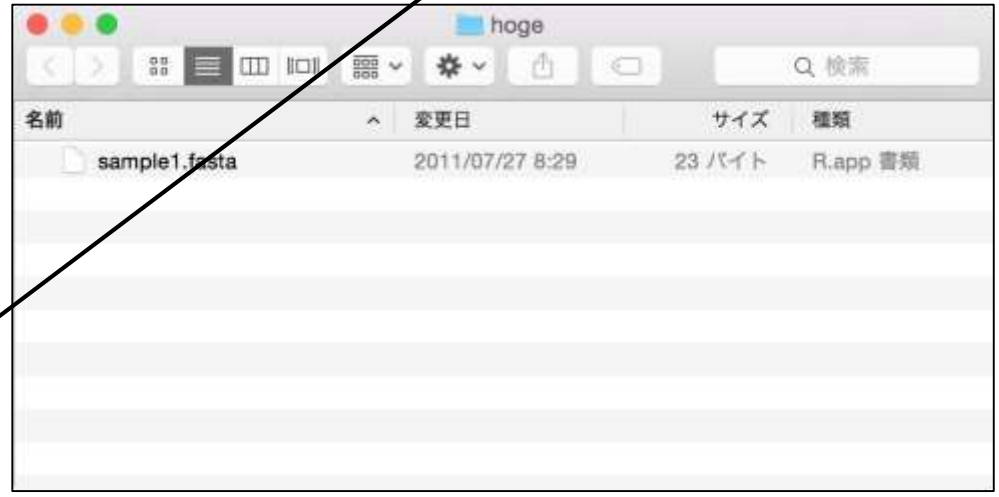
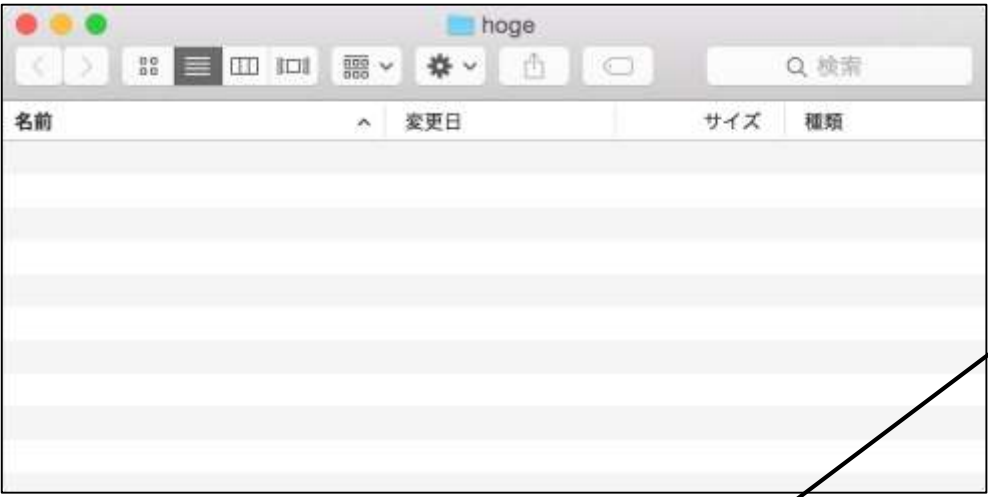
```
Rコンソール
~/Desktop/hoge
配布条件の詳細については、 license() あるいは License() と入力してください。
R は多くの貢献者による共同プロジェクトです。
詳しくは 'contributors()' と入力してください。
また、R や R のパッケージを出版物で引用する際の形式については
'citation()' と入力してください。
'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。
[R.app GUI 1.65 (6913) x86_64-apple-darwin13.4.0]
[履歴が次のファイルから読み込まれました /Users/kadota/.Rapp.history]
> getwd()
[1] "/Users/kadota"
> getwd()
[1] "/Users/kadota/Desktop/hoge"
> |
```



実際のhogeフォルダとR操作画面の関係

ファイル保存前

ファイル保存後



①一連のコマンド群をコピーして
②R Console画面上でペースト。

基本はコピペ

1. FASTA形式ファイル(sample1.fasta)の場合：

multi-FASTAではないsingle-FASTA形式ファイルです。

```

in_f <- "sample1.fasta"
out_f <- "hoge1.fasta"

#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
fasta <- readDNASTringSet(in_f,
                           fasta)

#本番
fasta <- translate(fasta)

#ファイルに保存
writeXStringSet(fasta, file=out_f, format="fasta")

```

#入力ファイル名を指定してin_fに格納
#出力ファイル名を指定してout_fに格納

"in_f <- \"sample1.fasta\" ...\"を調べる
Googleで検索

コピー ①

スピーチ

Googleで検索
スポークン

Rコンソール

~/Desktop/hoge

ヘルプ検索

```

> getwd()
[1] "/Users/kadota"
> getwd()
[1] "/Users/kadota/Desktop/hoge"
> list.files()
character(0)
> list.files()
[1] "sample1.fasta"
>

```

カット

コピー

ペースト ②

フォント ▶

スペルと文法 ▶

自動置換 ▶

変換 ▶

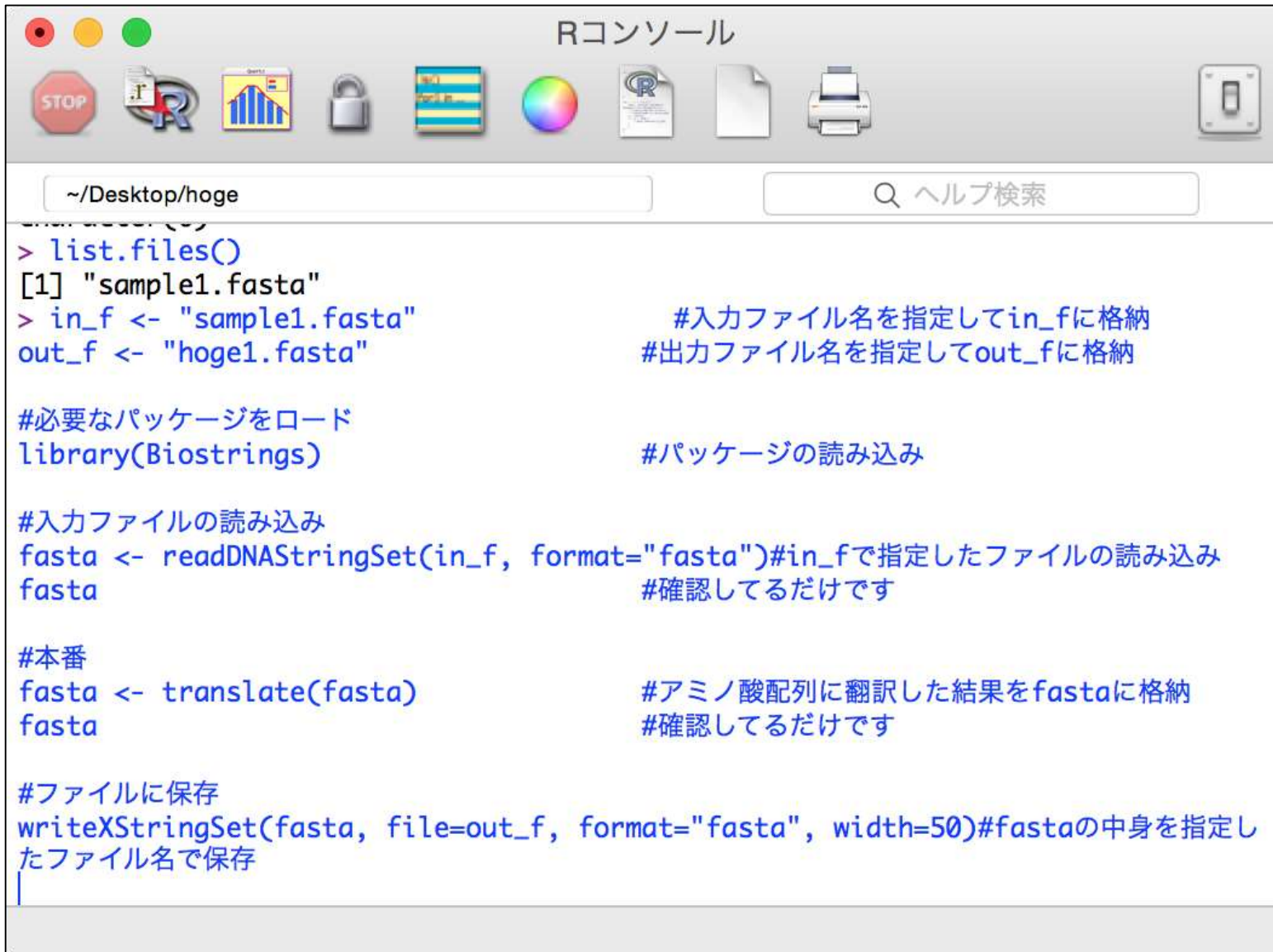
スピーチ ▶

レイアウトの方向 ▶

現在位置の関数のヘルプを表示 ^H

ペースト直後の状態。「リターンキー」を押す。

基本はコピペ



```
Rコンソール
~/Desktop/hoge
> list.files()
[1] "sample1.fasta"
> in_f <- "sample1.fasta"           #入力ファイル名を指定してin_fに格納
out_f <- "hoge1.fasta"             #出力ファイル名を指定してout_fに格納

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

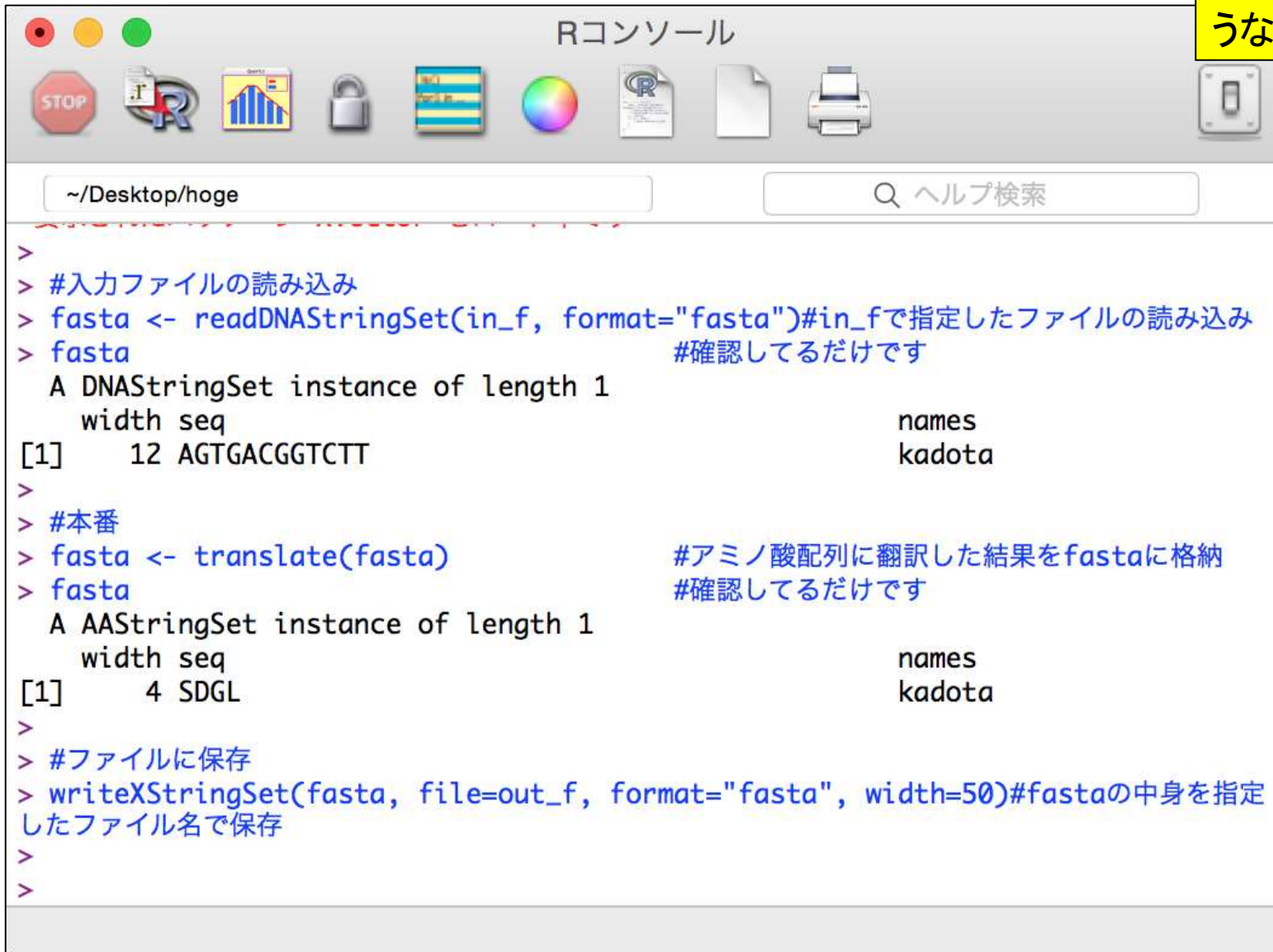
#入力ファイルの読み込み
fasta <- readDNASTringSet(in_f, format="fasta")#in_fで指定したファイルの読み込み
fasta                                #確認してるだけです

#本番
fasta <- translate(fasta)          #アミノ酸配列に翻訳した結果をfastaに格納
fasta                                #確認してるだけです

#ファイルに保存
writeXStringSet(fasta, file=out_f, format="fasta", width=50)#fastaの中身を指定したファイル名で保存
```

基本はコピー

「リターンキー」を押すとコピーしたコードが実行される。無事実行が終わると、このような画面になる。



```
>
> #入力ファイルの読み込み
> fasta <- readDNASTringSet(in_f, format="fasta")#in_fで指定したファイルの読み込み
> fasta                                     #確認してるだけです
A DNASTringSet instance of length 1
  width seq                                     names
[1]    12 AGTGACGGTCTT                          kadota
>
> #本番
> fasta <- translate(fasta)                  #アミノ酸配列に翻訳した結果をfastaに格納
> fasta                                     #確認してるだけです
A AAStringSet instance of length 1
  width seq                                     names
[1]     4 SDGL                                  kadota
>
> #ファイルに保存
> writeXStringSet(fasta, file=out_f, format="fasta", width=50)#fastaの中身を指定
したファイル名で保存
>
>
```

出力ファイル名として指定したhoge1.fastaが生成されていることが分かります

実行結果

```

Rコンソール
~/Desktop/hoge
次のパッケージを付け加えます: 'BiocGenerics'

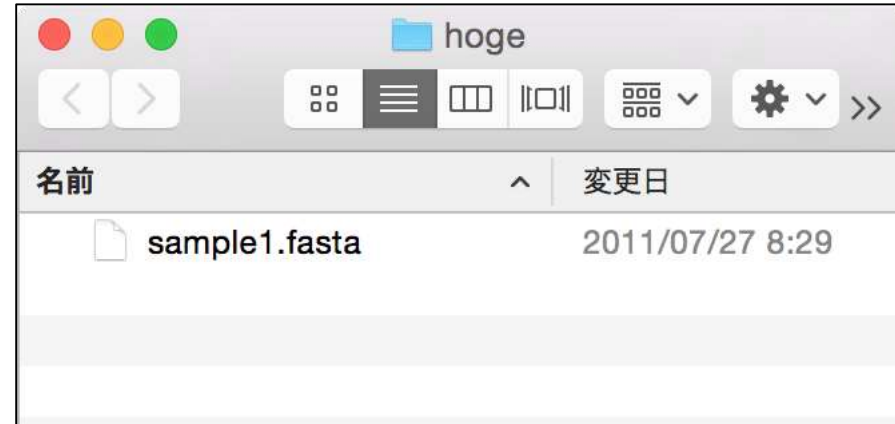
The following objects are masked from 'package:parallel':
  clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
  clusterExport, clusterMap, parApply, parCapply, parLapply,
  parLapplyLB, parRapply, parSapply, parSapplyLB

The following object is masked from 'package:stats':
  xtabs

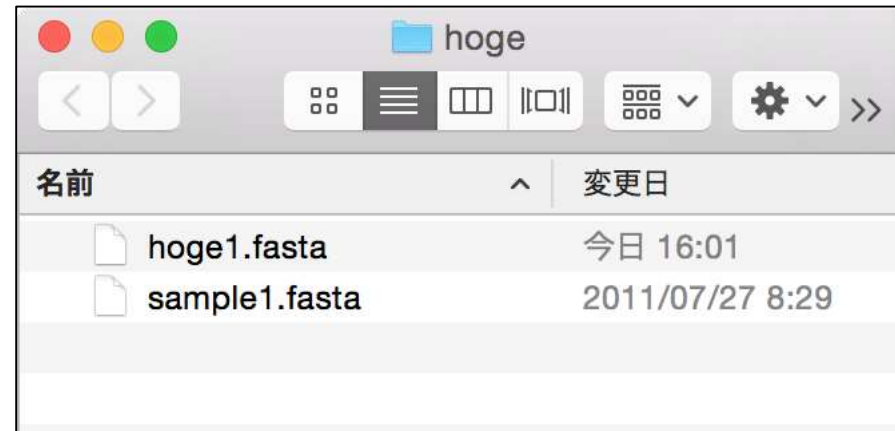
The following objects are masked from 'package:base':
  anyDuplicated, append, as.data.frame, as.vector, cbind,
  colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
  intersect, is.unsorted, lapply, Map, mapply, match, mget,
  order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
  rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
  table, tapply, union, unique, unlist, unsplit

要求されたパッケージ S4Vectors をロード中です
要求されたパッケージ stats4 をロード中です
要求されたパッケージ IRanges をロード中です
要求されたパッケージ XVector をロード中です
>
> #入力ファイルの読み込み
> fasta <- readDNAStringSet(in_f, format="fasta")#in_fで指定したファイルの読み込み
> fasta #確認するだけです
A DNASTringSet instance of length 1
  width seq                      names
[1] 12 AGTGACGGTCTT                kadota
>
> #本番
> fasta <- translate(fasta) #アミノ酸配列に翻訳した結果をfastaに格納
> fasta #確認するだけです
A AAStringSet instance of length 1
  width seq                      names
[1] 4 SDGL                        kadota
>
> #ファイルに保存
> writeXStringSet(fasta, file=out_f, format="fasta", width=50)#fastaの中身を指定
したファイル名で保存
  
```

実行前のhogeフォルダ



実行後のhogeフォルダ



実行結果

「list.files()」で表示される結果」と
「実行後のhogeフォルダの中身」
は当然同じです

```

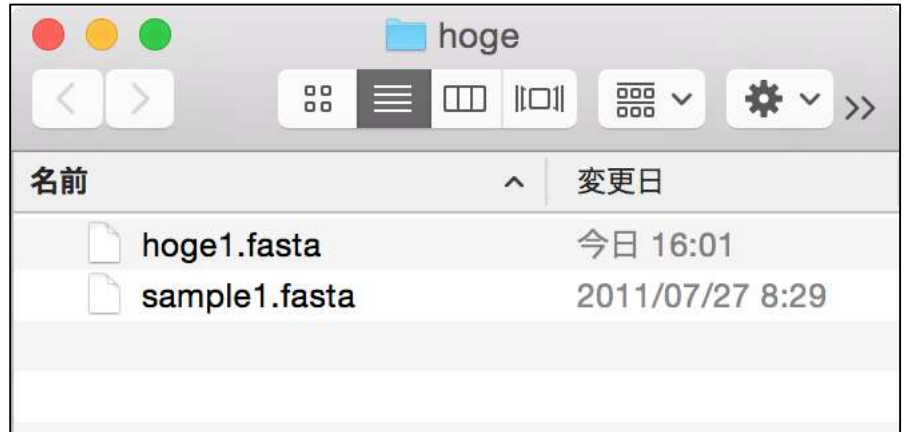
Rコンソール
~/Desktop/hoge
結果をFASTAに格納
> fasta #確認してるだけです
A AAStringSet instance of length 1
width seq names
[1] 4 SDGL kadota
> #ファイルに保存
> writeXStringSet(fasta, file=out_f, format="fasta",
width=50)#fastaの中身を指定したファイル名で保存
>
> list.files()
[1] "hoge1.fasta" "sample1.fasta"

```

実行前のhogeフォルダ



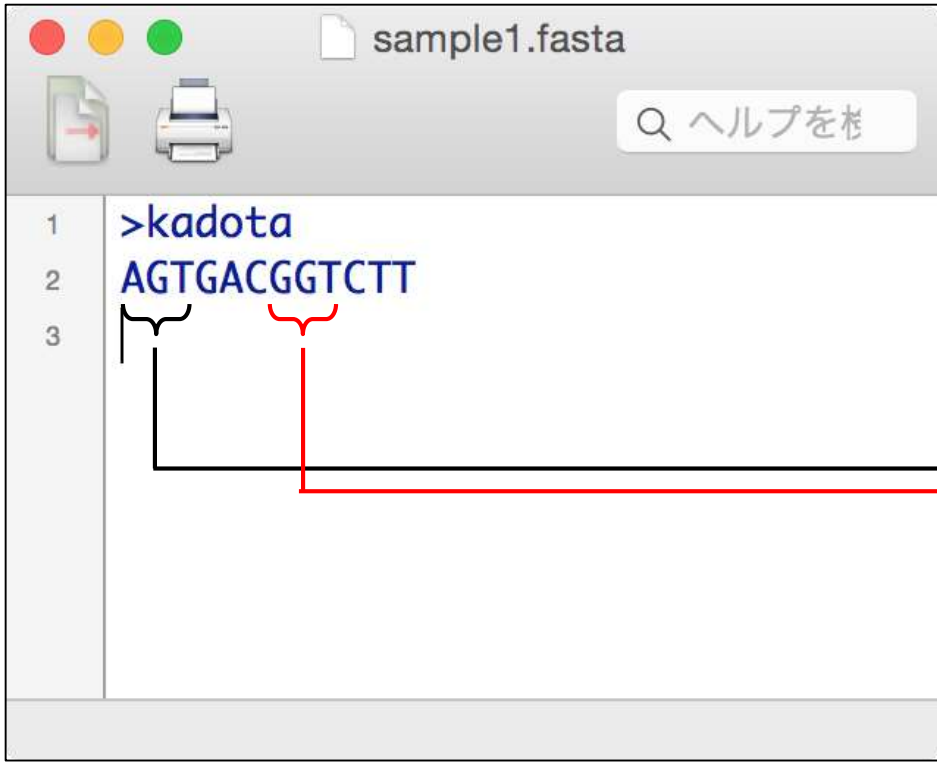
実行後のhogeフォルダ



入力ファイル中の塩基配列は、3の倍数の12塩基長、ACGTのみからなるので何のエラーもない

実行結果

入力: 塩基配列ファイル (sample1.fasta)



sample1.fasta

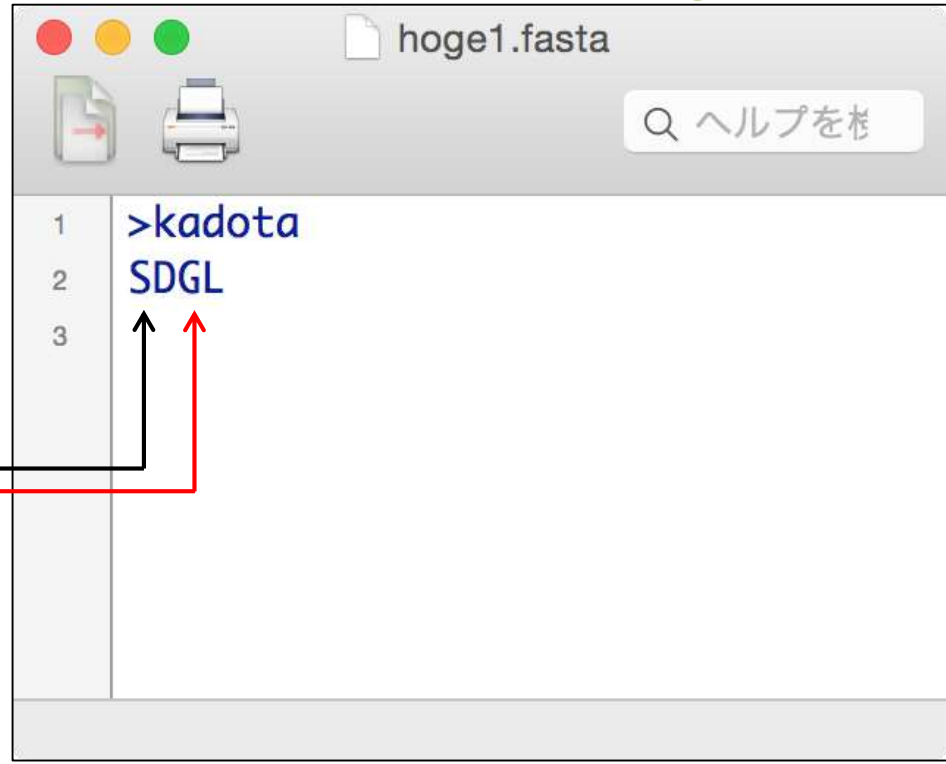
```
1 >kadota
2 AGTGACGGTCTT
3
```

The screenshot shows a text editor window for 'sample1.fasta'. The content is as follows:

1	>kadota
2	AGTGACGGTCTT
3	

Red brackets are drawn under the sequence 'AGTGACGGTCTT' to show its division into three codons: AGT, GAC, and GGT. A red line connects the first codon 'AGT' to the amino acid 'S' in the output window, and another red line connects the second codon 'GAC' to the amino acid 'D' in the output window.

出力: アミノ酸配列ファイル (hoge1.fasta)



hoge1.fasta

```
1 >kadota
2 SDGL
3
```

The screenshot shows a text editor window for 'hoge1.fasta'. The content is as follows:

1	>kadota
2	SDGL
3	

Black arrows point from the first and second lines of the output to the amino acid 'S' and 'D' respectively. Red arrows point from the first and second lines of the output to the amino acid 'G' and 'L' respectively.

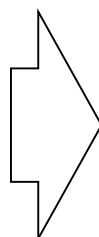
解析基礎2

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

入力: アノテーションファイル (annotation.txt)

	A	B	C	D
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

出力: hoge1.txt



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

入力: リストファイル (genelist1.txt)

	A
1	gene1
2	gene7
3	gene9

解析基礎2

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 (genelist.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

- ・ イントロ | 一般 | [ランダムに行を抽出](#) (last modified 2014/07/17)
- ・ イントロ | 一般 | [任意の文字列を行の最初に挿入](#) (last modified 2014/07/17)
- ・ イントロ | 一般 | [任意のキーワードを含む行を抽出\(基礎\)](#) (last modified 2014/04/11)
- ・ イントロ | 一般 | [ランダムな塩基配列を生成](#) (last modified 2014/06/16)
- ・ イントロ | 一般 | [任意の長さの可能な全ての塩基配列を作成](#) (last modified 2015/02/19)
- ・ イントロ | 一般 | [任意の位置の塩基を置換](#) (last modified 2013/09/12)
- ・ イントロ | 一般 | [指定した範囲の配列を取得](#) (last modified 2014/06/16)
- ・ イントロ | 一般 | [指定したID\(染色体やdescription\)を取得](#) (last modified 2014/06/16)
- ・ イントロ | 一般 | [翻訳配列\(translate\)を取得\(基礎\)](#) (last modified 2014/06/16)
- ・ イントロ | 一般 | [翻訳配列\(translate\)を取得\(応用\)](#) (last modified 2014/06/16)

イントロ | 一般 | 任意のキーワードを含む行を抽出(基礎)

例えばタブ区切りテキストファイルが手元があり、この中からリストファイル中の文字列を含む行を抽出するやり方を示します。Linux (UNIX) の grep コマンドのようなものであり、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ー。

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 ([genelist.txt](#)) 中のものが含まれる行全体を出力したい場合:

```
in_f1 <- "annotation.txt" #入力ファイル名を指定してin_f1に格納(アノテーションファイル)
in_f2 <- "genelist.txt" #入力ファイル名を指定してin_f2に格納(リストファイル)
out_f <- "hoge1.txt" #出力ファイル名を指定してout_fに格納
param <- 1 #アノテーションファイル中の検索したい列番号を指定

#入力ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="\t", quote="") #in_f1で指定したファイルの読み込み
keywords <- readLines(in_f2) #in_f2で指定したファイルの読み込み
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) #条件を満たすかどうかを判定した結果
out <- data[obj,] #objがTRUEとなる行のみ抽出した結果をoutに格納
dim(out) #オブジェクトoutの行数と列数を表示

#ファイルに保存
write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F) #outの中身を指定したファイルに保存
```

解析基礎2

作業ディレクトリは「デスクトップ - hoge」。hogeフォルダ中にannotation.txtとgenelist1.txtが存在するという前提。

イントロ | 一般 | 任意のキーワードを含む行を抽出(基礎)

例えばタブ区切りテキストファイルが手元があり、この中からリストファイル中の文字列を示します。Linux (UNIX)のgrepコマンドのようなものであり、perlのハッシュのようなもの「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

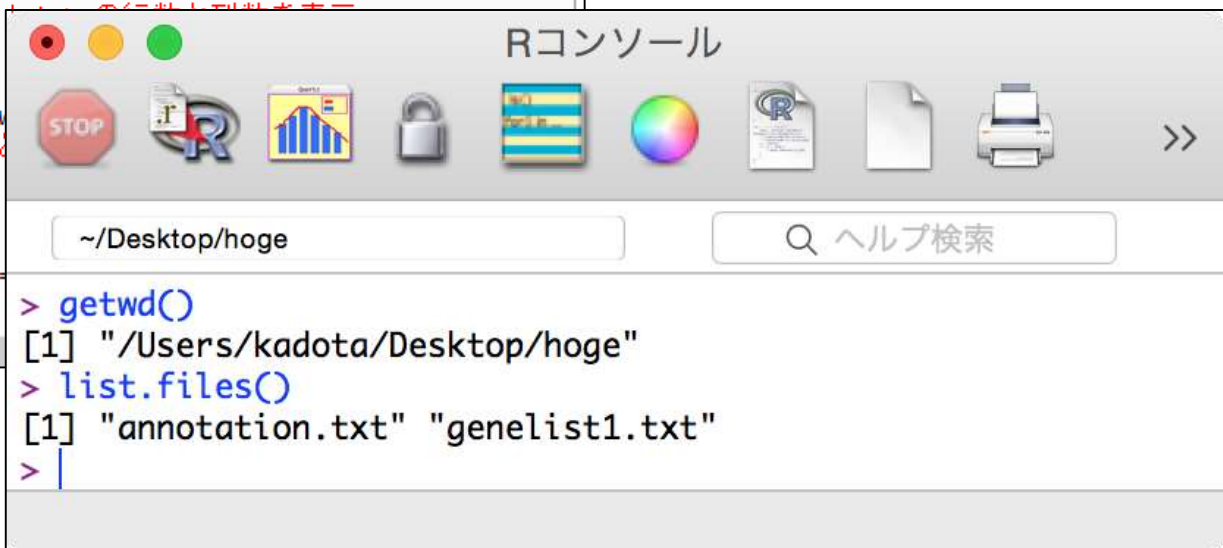
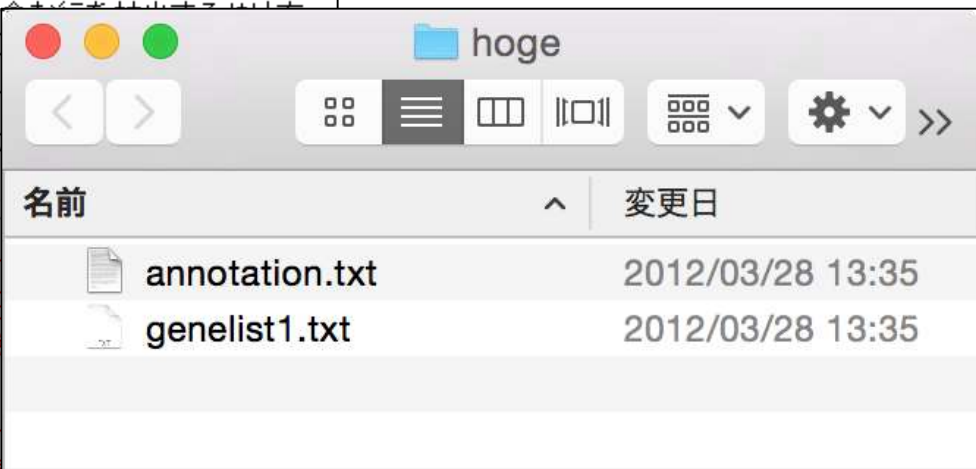
1. 目的のタブ区切りテキストファイル(annotation.txt)中の第1列目をキーとして、リストのものに含まれる行全体を出力したい場合:

```
in_f1 <- "annotation.txt" #入力ファイル名を指定してin_
in_f2 <- "genelist1.txt" #入力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してout_
param <- 1 #アンテーションファイル中の
```

```
#入力ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="\t", quote="") #in_f1
keywords <- readLines(in_f2) #in_f2で指定したファイルの読
dim(data) #オブジェクト
```

```
#本番
obj <- is.element(as.character(data[,param]), keywords) #objがTRUE
out <- data[obj,] #オブジェクト
dim(out)
```

```
#ファイルに保存
write.table(out, out_f, sep="\t", append=F, quote=)
```



基本はコピペ

- ①一連のコマンド群をコピーして
- ②R Console画面上でペースト。
- ③「リターンキー」を押す

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：

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

#入力ファイルの読み込み

```
data <- read.table(in_f1,
keywords <- readLines(in_f2)
dim(data)
```

#本番

```
obj <- is.element(as.character(data[,1]), keywords)
out <- data[obj,]
dim(out)
```

#ファイルに保存

```
write.table(out, out_f, sep="\t", as.is=TRUE)
```

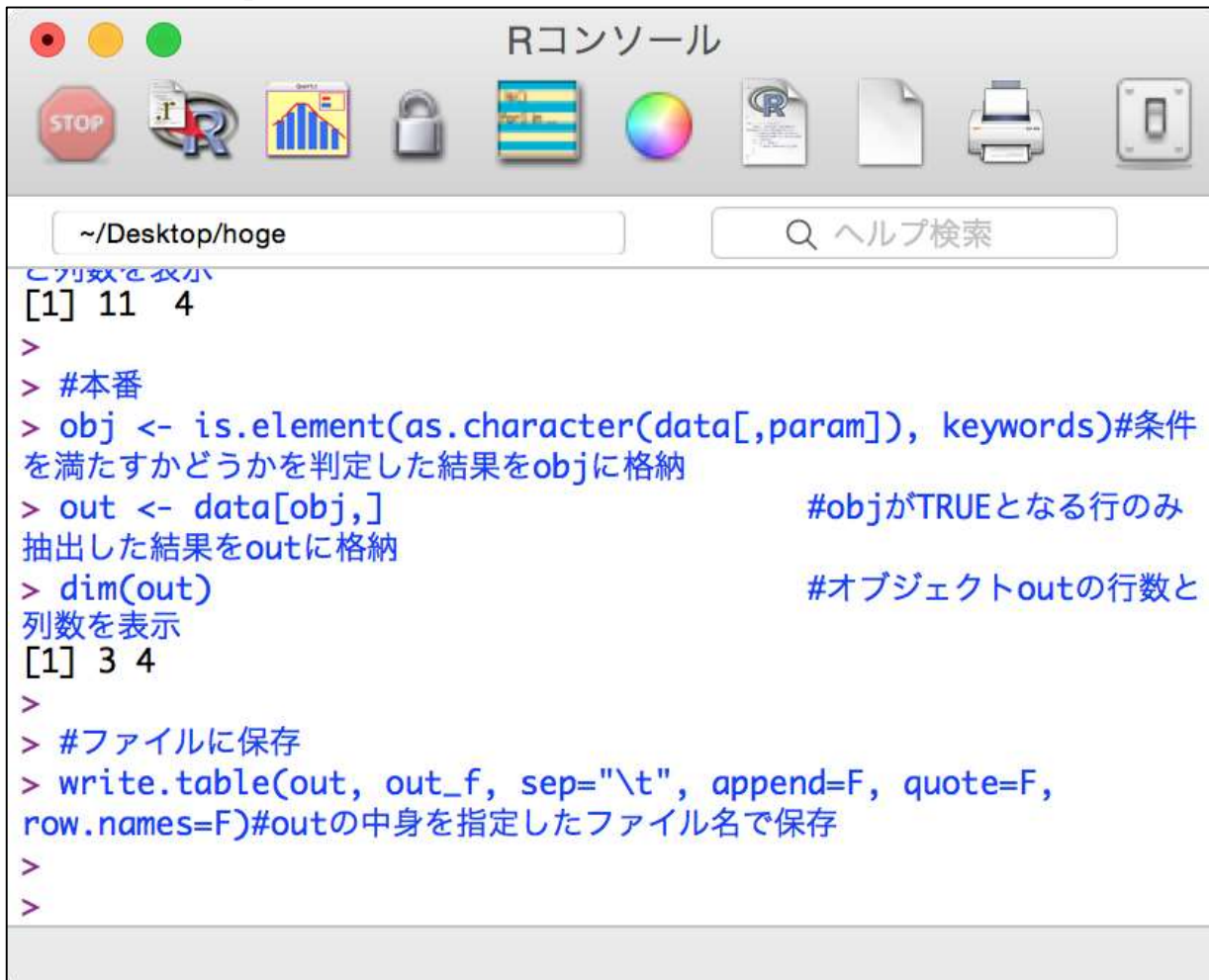
“in_f1 <- "annotation.txt..."”を調べる
Google で検索
コピー ①
スピーチ
Google で検索
スピーク

in_f1に格納(アノテーション)
in_f2に格納(リスト)
out_fに格納
検索したい列番号
f1で指定したファイルの読み込み

Rコンソール
~/Desktop/hoge
> getwd()
[1] "/Users/kadota/Desktop/"
> list.files()
[1] "annotation.txt" "genelist1.txt"
> |
カット
コピー
ペースト ②
フォント
スペルと文法
自動置換
変換
スピーチ
レイアウトの方向
現在位置の関数のヘルプを表示 ^H

基本はコピー

「リターンキー」を押すとコピーしたコードが実行される。無事実行が終わると、このような画面になる。



```
Rコンソール
~/Desktop/hoge
[1] 11 4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)#条件
を満たすかどうかを判定した結果をobjに格納
> out <- data[obj,] #objがTRUEとなる行のみ
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
[1] 3 4
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F, quote=F,
row.names=F)#outの中身を指定したファイル名で保存
>
>
```

実行結果

「list.files()」で表示される結果」と「実行後のhogeフォルダの中身」は当然同じです

```
Rコンソール
~/Desktop/hoge
ヘルプ検索

> #本番
> obj <- is.element(as.character(data[,param]),
keywords)#条件を満たすかどうかを判定した結果をobjに格納
> out <- data[obj,] #objがTRUEと
なる行のみ抽出した結果をoutに格納
> dim(out) #オブジェクト
outの行数と列数を表示
[1] 3 4
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F,
quote=F, row.names=F)#outの中身を指定したファイル名で保存
>
> list.files()
[1] "annotation.txt" "genelist1.txt" "hoge1.txt"
> |
```

実行前のhogeフォルダ

名前	変更日
annotation.txt	2012/03/28 13:35
genelist1.txt	2012/03/28 13:35

実行後のhogeフォルダ

名前	変更日
annotation.txt	2012/03/28 13:35
genelist1.txt	2012/03/28 13:35
hoge1.txt	今日 16:34

実行結果

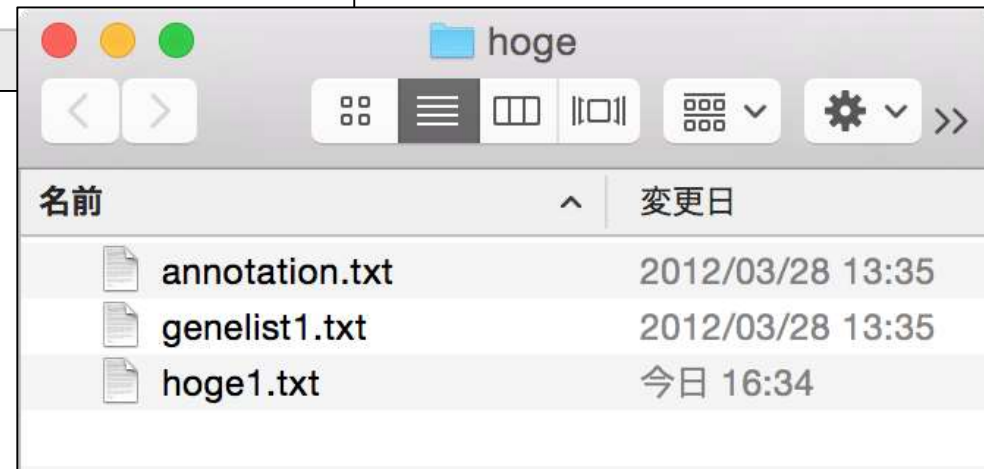
outというオブジェクトの中身をwrite.tableという関数でファイルに出力しています。それゆえ、出力ファイル(hoge1.txt)の中身は、Rコンソール画面中で「out」と打ち込むことでも見られます。

```

Rコンソール
~/Desktop/hoge
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F)#out
の中身を指定したファイル名で保存
>
> list.files()
[1] "annotation.txt" "genelist1.txt" "hoge1.txt"
> out
  gene_name accession description subcellular_location
1   gene1     hoge01  plasma_mem          nuclear
7   gene7     hoge07   tebasaki          nuclear
9   gene9     hoge09   nihonshu          nuclear
  
```

実行後のhogeフォルダ

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



色の説明

Rコード中の色の使い分けについて説明します。

(Rで)塩基配列解析

～NGS、RNA-seq、ゲノム、トランスクリプトーム、正規化、発現変動、統計、モデル、バイオインフォマティクス～
(last modified 2015/04/02, since 2010)

What's new?

- このウェブページは [インストール | についての推奨手順](#)(Win版(2015.04.01)とMac版(2015.04.02))通りにフリーソフトRと必要なパッケージをインストール済みであるという前提で記述しています。初心者の方は[基本的な利用法](#)で自習してください。本ウェブページを体系的にまとめた[書籍](#)もあります。(2015/04/02) **NEW**
- 私の所属する[アグリバイオインフォマティクス教育研究プログラム](#)では、平成27年度もバイオインフォ関連講義を行います。例年東大以外の企業の方、研究員、学生が2-3割程度受講しております。受講ガイダンスは4月6日17:15- 於東大農です。(2015/03/31) **NEW**
- R本体およびパッケージのインストール手順のところを更新しました。詳細は[インストール | について](#)をごらんください。(2015/04/02) **NEW**
- [MBCluster.Seq](#)パッケージを用いた遺伝子間クラスタリングのやり方を一通
- [参考資料 \(講義、講習会、本など\)](#)の項目を更新しました。(2015/03/09) **NEW**
- [はじめに](#) (last modified 2015/03/31) **NEW**
- [参考資料 \(講義、講習会、本など\)](#) (last modified 2015/03/09) **NEW**
- [過去のお知らせ](#) (last modified 2015/02/21) **NEW**

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

[トップページへ](#)

応用

このサンプルコードは1列目でキーワード検索する場合。別のリストファイルを読み込んで4列目で検索したい場合のやり方を示します。

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](#))の含まれる行全体を出力したい場合:

```
in_f1 <- "annotation.txt" #入力ファイル名を指定してin_f1に格納(アノテーションファイル)
in_f2 <- "genelist1.txt" #入力ファイル名を指定してin_f2に格納(リストファイル)
out_f <- "hoge1.txt" #出力ファイル名を指定してout_fに格納
param <- 1 #アノテーションファイル中の検索したい列番号を指定
```

#入力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)
```

#in_f2で指定したオブジェクト

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

#本番

```
obj <- is.element(as.character(data[,param]), keywords) #条件を満たすかどうかを判定した結果をobjに格納
out <- data[obj,] #objがTRUEとなる行のみ抽出した結果をoutに格納
dim(out) #オブジェクトoutの行数と列数を表示
```

#ファイル出力
write.table(out, out_f, sep="\t", quote="")

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

row.name



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

一連の作業手順を記述したスクリプトを1つのファイルとして保存することをお勧め

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ー

```
nuclear↓
membrane↓
↓
```

```
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 <- "hogel.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 <- "hogel.txt"
param <- 4

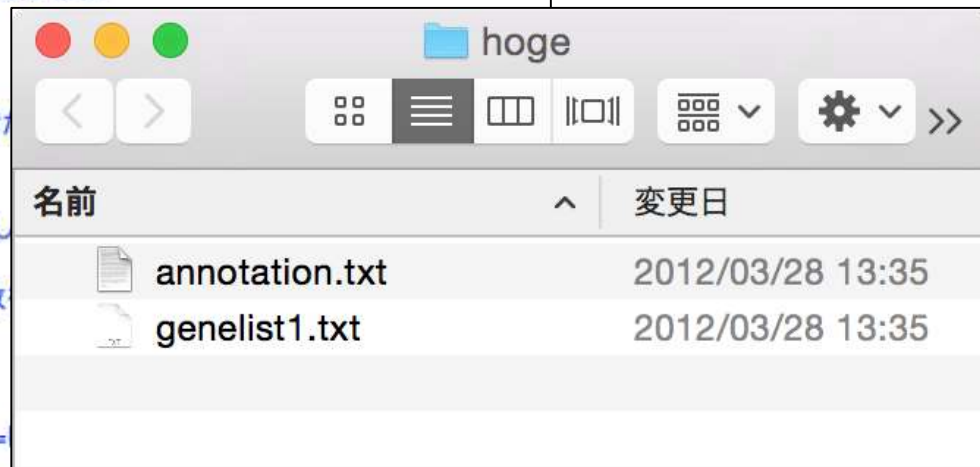
#ファイルの読み込み
data <- read.table(in_f1, header=TRUE, sep="¥t", quote="")
keywords <- readLines(in_f2)
dim(data)

#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=
```

ありがちなミス1

作業ディレクトリの変更を忘れていたため、in_f1で指定した最初のファイルの読み込み段階でエラーが出る。つまり、実際に行ったフォルダ中にはannotation.txtというファイルは存在しないということ。

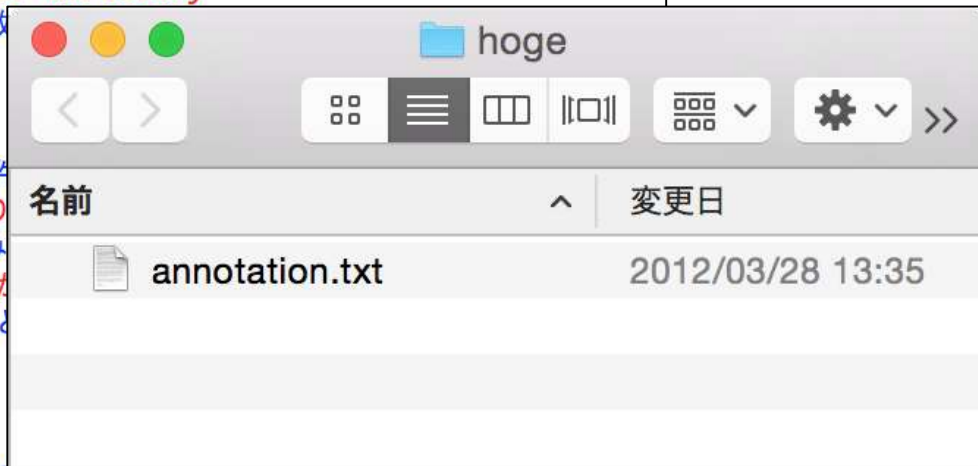
```
> in_f1 <- "annotation.txt" #入力ファイル名を指定してin_f1に格納(アノテーションファイル)
> in_f2 <- "genelist1.txt" #入力ファイル名を指定してin_f2に格納(リストファイル)
> out_f <- "hoge1.txt" #出力ファイル名を指定してout_fに格納
> param <- 1 #アノテーションファイル中の検索したい列番号を指定
>
> #入力ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", quote="") #in_f1で指定したファイルの読み込み
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 directory
> keywords <- readLines(in_f2) #in_f2で指定したファイルの読み込み
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
#オブジェクトdataの行数と列数を表示
> dim(data)
NULL
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords) #条件を満たす行の抽出
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,] #objがTRUEとなる行のみ抽出
エラー: オブジェクト 'obj' がありません
> dim(out) #オブジェクトoutの行数と列数
エラー: オブジェクト 'out' がありません
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=FALSE)
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] "/Users/kadota"
```



ありがちなミス2

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。この場合、in_f2で指定したgenelist1.txtが存在しないため、その読み込み段階でエラーが出ている。それゆえ、その情報を用いているコマンド部分でエラーが出ている。

```
> getwd()
[1] "/Users/kadota/Desktop/hoge"
> list.files()
[1] "annotation.txt"
> in_f1 <- "annotation.txt" #入力ファイル名を指定してin_f1に格納(アンノテーションファイル)
> in_f2 <- "genelist1.txt" #入力ファイル名を指定してin_f2に格納(リストファイル)
> out_f <- "hoge1.txt" #出力ファイル名を指定してout_fに格納
> param <- 1 #アンノテーションファイル中の検索したい列番号を指定
>
> #入力ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", quote="") #in_f1で指定したファイルの読み込み
> keywords <- readLines(in_f2) #in_f2で指定したファイルの読み込み
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dataの行数
[1] 11 4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords) #条件
以下にエラー match(el, set, 0L) : オブジェクト 'keywords' があり
> out <- data[obj,] #objがTRUEとなる行のみ
以下にエラー `[.data.frame`(data, obj, ) : オブジェクト 'obj' が
> dim(out) #オブジェクトoutの行数
エラー: オブジェクト 'out' がありません
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F) #outの中身を指定したファイル名で保存
以下にエラー is.data.frame(x) : オブジェクト 'out' がありません
>
```



ありがちなミス3

出力予定のファイル名と同じものをエクセルなど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。対処法は、出力ファイル名を変更するか、開いている別のプログラムを閉じる。これはWindowsの例。Macの場合、Excelで開いているがおかまいなしに上書き保存されてしまうが、他のアプリケーションで開いている場合にこのようなエラーに遭遇する可能性があるため、念のため掲載。

```

RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ Vign
[Icons]

R Console
> in_f1 <- "annotation.txt"
> in_f2 <- "list.txt"
> out_f <- "hogel.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)
[1] 11 4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)
> out <- data[obj,]
> dim(out)
[1] 7 4
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=
以下にエラー file(file, ifelse(append, "a", "w")) :
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, ifelse(append, "a", "w")) :
ファイル 'hogel.txt' を開くことができません: Permission denied
> |

```

#入

1	gene	name	accession	description	subcellular_location
2	gene1		hoge01	plasma_mer	nuclear
3	gene7		hoge07	tebasaki	nuclear
4	gene9		hoge09	nihonshu	nuclear
5					
6					

```

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 <- "hogel.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=

```

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)。これもWindowsの例。Macの場合、コピー後にリターンキーを押すのでこのようなミスは犯しにくいが念のため掲載。

```
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" #入力ファイル名
in_f2 <- "list.txt" #入力ファイル名
out_f <- "hogel.txt" #出力ファイル名
param <- 4 #in_f1で読み込んだファイル中の列目番号

#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とkeywordsに格納した文字列ベクトルを比較
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。

> 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
[1] 11 4
> #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とkeywordsに格納した文字列ベクトルを比較
> 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
[1] 7 4
> 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。
```

警告メッセージ

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(左)と入れない場合(右)の挙動の違いを把握し、後学のために警告メッセージの意味を理解しておくとい。この場合は結果には影響していないことがわかる。Rは警告メッセージ後の記述内容が比較的分かりやすいのでよく読むべし。

```
> in_f1 <- "annotation.txt"
> in_f2 <- "list.txt"
> out_f <- "hoge1.txt"
> param <- 4
>
> #入力ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t"
> keywords <- readLines(in_f2)
> dim(data)
[1] 11 4
>
> #本番
> obj <- is.element(as.character(data[,param]), k
> out <- data[obj,]
> dim(out)
[1] 7 4
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F, qu
>
```

nuclear ↓
membrane ↓
←

```
> in_f1 <- "annotation.txt"
> in_f2 <- "list.txt"
> out_f <- "hoge1.txt"
> param <- 4
>
> #入力ファイルの読み込み
> data <- read.table(in_f1, header=TRUE, sep="\t", quote=""
> keywords <- readLines(in_f2)
警告メッセージ:
In readLines(in_f2) : 'list.txt' で不完全な最終行が見つかりました
> dim(data)
[1] 11 4
>
> #本番
> obj <- is.element(as.character(data[,param]), keywords)#条件を満た
> out <- data[obj,]
> dim(out)
[1] 7 4
>
> #ファイルに保存
> write.table(out, out_f, sep="\t", append=F, quote=F, row.names=F
>
```

nuclear ↓
membrane ←