Package 'seqRFLP'

February 20, 2015

Type Package

Title Simulation and visualization of restriction enzyme cutting pattern from DNA sequences.

Version 1.0.1

Date 2010-10-13

Author Qiong Ding, Jinlong Zhang

Maintainer Qiong Ding <dingqiong1@gmail.com>

Description This package includes functions for handling DNA sequences, especially simulated RFLP and TRFLP pattern based on selected restriction enzyme and DNA sequences.

License GPL-2

LazyLoad yes

Repository CRAN

Date/Publication 2012-10-29 08:59:40

NeedsCompilation no

R topics documented:

seqRFLP-package *Simulation and visualization of Restriction Fragment Length Polymorphism(RFLP) pattern from DNA sequences*

Description

This package includes functions for handling DNA sequences, especially for simulating RFLP and TRFLP pattern based on selected restriction enzyme and DNA sequences.

Details

Package:	seqRFLP
Type:	Package
Version:	1.0.1
Date:	2010-10-13
License:	GPL-2
LazyLoad:	yes

Author(s)

Qiong Ding <dingqiong1@gmail.com> Jinlong zhang <jinlongzhang01@gmail.com>

Maintainer: Qiong Ding <dingqiong1@gmail.com>

References

Saiki RK, Scharf S, Faloona F, Mullis KB, Erlich HA, Arnheim N (1985). Enzymatic amplification of beta-globin genomic sequences and restriction site analysis for diagnosis of sickle cell anemia. Science 230 (4723):1350-4

Examples

```
### Featured examples
### 1. File(s) importing:

### 1.1 importing fasta files.
### read.fasta() example
###
cat(
”>No305”,
”NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT”,
file = ”dna1.fas”, sep = ”\n”)
sequences <- read.fasta(”dna1.fas”)
unlink(”dna1.fas”)


### 1.2 Concatenating all fasta files in working directory to fasta data
### raw2Fas() example
cat(
”>No305”,
”NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT”,
file = ”dna1.fas”, sep = ”\n”)
cat(
”>No304”,
”ATTCGAAAAACACACCCACTACTAAAAATTATCAACCACT”,
file = ”dna2.fas”, sep = ”\n”)
cat(
”>No305”,
”NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT”,
file = ”dna3.fas”, sep = ”\n”)

raw2Fas(getwd(), appendix = ”.fas”)

unlink(”dna1.fas”)
unlink(”dna2.fas”)
unlink(”dna3.fas”)

## 1.3 importing phy, nexus, and converting to fasta objects for further analysis.

##### read.nxs() example
data(fil.nxs)
writeLines(fil.nxs, ”example.nxs”)
nxs <- read.nxs(”example.nxs”)
as.fasta(nxs)
unlink(”example.nxs”)

##### read.phy() example
data(fil.phy)
writeLines(fil.phy, ”example.phy”)
phy <- read.phy(”example.phy”)
as.fasta(phy)
unlink(”example.phy”)

## 2 Check and visualising cutting patterns of restriction enzymes on sequences.
## 2.1 enzCut() example
data(enzdata)
enznames <- c("EcoRI", "Acc65I")
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
enzCut(DNAsq = fas[2], enznames = "AluI", enzdata = enzdata)
rm(list = ls())

### 2.2  fragdat() example
data(enzdata)
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
frag.dat(fil = fas, enznames = "EcoRI", enzdata = enzdata)

## 2.3 plotenz() example
data(enzdata)
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
enznames <- c("EcoRI", "Acc65I", "HinfI")

plotenz(sequences = fas, enznames = enznames,
        enzdata = enzdata, side = TRUE, type = "RFLP")

plotenz(sequences = fas, enznames = enznames,
        enzdata = enzdata, side = FALSE, type = "RFLP")

plotenz(sequences = fas, enznames = enznames,
        enzdata = enzdata, side = TRUE, type = "TRFLP", "T3")

### 3 Sequence selection based on given primers.

### clipprobe()
## 3.1 Specify the forward and backward primer.
#clip the sequence between the backword and forward primer.
forProbe = ITS1F = 'CTTGGTCATTTAGAGGAAGTAA' # forward primer should be from the 5' to 3' end.
bacProbe = ITS4 = 'GCATATCAATAAGCGGAGGA'    # backward primer
#only sequence with two probes found could be clipped.

### 3.2 reading fasta data.
directory <- system.file("extdata", package = "seqRFLP")
path <- file.path(directory, "seqs.fasta")
fas <- read.fasta(path)

## 3.3 Clipping. Find clipped sequences, this usually takes less than 1 minunite.
## please wait for a moment.
clipped <- clipprobe(fas, forProbe, bacProbe, tol = 0, clipped.only = TRUE)

## 3.4 Checking the results.
## There are 368 selected sequences that could be clipped.
length(gnames.fas(clipped))
## ... in 393 original sequences.
length(gnames.fas(fas))
## Sequences that can not be clipped.
```

```
setdiff(gnames.fas(fas), gnames.fas(clipped))
```

as.fasta *Coerce "phy" or "nex" objects to fasta format.*

Description

This function could be used to coerce "phy", "nex" to "fasta" format.

Usage

```
as.fasta(x)
```

Arguments

x	Objects from read.phy() or read.nxs().

Details

The input data must be either in class "phy", "nxs", "fasta". This means that it must be the object from read.phy() or read.nxs() etc.

Value

fasta object.

Author(s)

Jinlong Zhang <jinlongzhang01@gmail.com>

References

None.

Examples

```
data(fil.nxs)
nxs <- as.fasta(fil.nxs)
nxs

data(fil.phy)
phy <- as.fasta(fil.phy)
phy
```

clipprobe *Finding the sequences that could be clipped given two primers.*

Description

Finding the sequences that could be clipped given the forward and backward primers.

Usage

```
clipprobe(fil, forProbe, bacProbe, tol = 3, clipped.only = TRUE)
```

Arguments

`fil`	Sequences in fasta format to be analyzed. Must be in `"fasta"` format.
`forProbe`	The forward primer.
`bacProbe`	The backward primer.
`tol`	Mumber of DNA base that can not be matched by the primers.
`clipped.only`	Whether only show the sequences that can be clipped.

Details

This function can be used for predicting whether the sequences can be obtained by Polymerase chain reaction (PCR) given the forward and backward primers. Users may adjust the precision of DNA sites matching by the given primers on the sequences.

Value

if `clipped.only = TRUE`, the function will return to the sequence(s) that could be obtained by PCR.

Author(s)

Qiong Ding `<dingqiong1@gmail.com>`

References

None.

See Also

See Also `frag.dat` for restriction enzyme clippling pattern.

Examples

```
### clipprobe() example

## Step1 Specify the forward and backward primer.
#clip the sequence between the backword and forward primer.
forProbe = ITS1F = 'CTTGGTCATTTAGAGGAAGTAA' # forward primer should be from the 5' to 3' end.
bacProbe = ITS4 = 'GCATATCAATAAGCGGAGGA'    # backward primer
#only sequence with two probes found could be clipped.

### Step2 reading sequences from external data in package.
directory <- system.file("extdata", package = "seqRFLP")
path <- file.path(directory, "seqs.fasta")
fas <- read.fasta(path)

## Step3 Clipping. Find clipped sequences, this usually takes less than 1 minute.
## please wait for a moment.
clipped <- clipprobe(fas, forProbe, bacProbe, tol = 2, clipped.only = TRUE)

## Step4 Checking the results.
## 368 selected sequences that could be clipped.
length(gnames.fas(clipped))
## 393 original sequences.
length(gnames.fas(fas))
## Sequences that can not be clipped.
setdiff(gnames.fas(fas), gnames.fas(clipped))
```

ConvFas *Convert files to fasta format*

Description

This function could be used to convert raw phy,nex, fas files to fasta format. It is also a internal function called by as.fasta() which is more friendly to use.

Usage

```
ConvFas(fil = NULL, type = c("fas", "nxs", "phy"))
```

Arguments

fil	file that to be converted.
type	File types that will be converted.

Details

Users may have to use readLines() to import the content when dealing with external datasets. Currently, this function could handling the standard phy, nex, fas files, lines between "[]" must be deleted in nex files.

Value

The "fasta" format data.

Author(s)

Qiong Ding <dingqiong1@gmail.com>

References

None.

See Also

See Also as.fasta for converting data types.

Examples

```
data(fil.phy)
ConvFas(fil = fil.phy, type = "phy")
data(fil.nxs)
ConvFas(fil = fil.nxs, type = "nxs")
```

---

dataframe2fas *Convert dataframe to fasta format*

Description

This function could be used to convert dataframe to fasta format.

Usage

```
dataframe2fas(x, file = NULL)
```

Arguments

x	The input dataframe
file	A character naming the file to be saved to.

Details

The input dataframe with the first column the sequences' names, the second column DNA sequences.

Value

Returns data in fasta format.

Author(s)

Jinlong Zhang <jinlongzhang01@gmail.com>

References

None.

See Also

See also file.cat and raw2Fas

Examples

```
######## fataframe2fas() example
dd <- dataframe2fas(paste("AAACCCTTAAAAAAATTA
TTTTCTATTGGTTTCTTGGGGGGGTT", 1:10, sep = ""))
dd
```

enzCut *Restriction enzyme cutting pattern*

Description

Function for calculating Restriction enzyme cutting pattern

Usage

```
enzCut(DNAsq, enznames, enzdata = enzdata)
```

Arguments

DNAsq	The input fasta data.
enznames	A character vector of the restriction enzyme's names, from one to multiple names.
enzdata	Dataframe with first column for enzyme names, second column for enzyme cutting patterns.

Details

Users could add restriction sites manually according to the enzdata styles.

Value

`enz`	The selected restriction enzyme.
`RFLP.site`	The sites recognized by enzyme.
`RFLP.frag`	The fragments generated by enzyme cutting.
`TRFLP`	The fragments predicted by TRFLP

Author(s)

Qiong Ding <dingqiong1@gmail.com>

References

None.

See Also

See Also `frag.dat`

Examples

```
## enzCut() example
data(enzdata)
enznames <- c("EcoRI", "Acc65I")
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
enzCut(DNAsq = fas[2], enznames = "AluI", enzdata = enzdata)
```

enzdata *The restriction enzyme datasets.*

Description

The rebase restriction enzyme datasets.

Usage

```
data(enzdata)
```

Format

A data frame with 777 restriction enzymes data with first column the names of enzyme, the second column the corresponding cutting pattern.

`nam` Names of restriction enzyme.

`site` Cutting pattern for each restriction enzyme.

Source

Roberts, R.J., Vincze, T., Posfai, J., Macelis, D. (2010) REBASE–a database for DNA restriction and modification: enzymes, genes and genomes. Nucl. Acids Res. 38: D234-D236. http://rebase.neb.com

References

None.

Examples

```
data(enzdata)
head(enzdata)
```

`fil.fas` *Fasta data example.*

Description

Internal Fasta format example

Usage

```
data(fil.fas)
```

Format

Class ’fasta’ with elements containing ">"indicating sequence names and the DNA sequences in subsequent elements.

Source

From Qiong Ding

References

None.

Examples

```
data(fil.fas)
fil.fas
```

fil.nxs *Internal nexus data.*

Description

Internal data in class nxs

Usage

```
data(fil.nxs)
```

Format

Class ’nxs’

Source

Qiong Ding

References

None.

Examples

```
data(fil.nxs)
fil.nxs
```

fil.phy *Internal phylip format data*

Description

Provide an example of phylip data format.

Usage

```
data(fil.phy)
```

Format

class ’phy’. With each line stored as one element in a character vector.

Details

To be modified later.

Source

Provided by Qiong Ding.

References

None.

Examples

```
data(fil.phy)
fil.phy
```

`file.cat` *Copy or concatenate files to one.*

Description

This function could be used to concatenate the files in the given directory.

Usage

```
file.cat(dir = NULL, appendix = ".fas", file = NULL)
```

Arguments

`dir`	A string name the directory where files to be concatenated exist.
`appendix`	The appendix that files to be concatenated have.
`file`	The file name of the result to be save to.

Details

Any plain text files that match the `appendix` could be concatenate or copy to one file.

Value

Copy all the contents of the files having the matched appendix to one file.

Author(s)

Jinlong zhang <`jinlongzhang01@gmail.com`> Qiong Ding <`dingqiong1@gmail.com`>

References

None.

See Also

See Also `raw2Fas` for copying all different fasta file contents to one.

Examples

```
#### file.cat() example
cat(
">No305",
"NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT",
file = "dna1.fas", sep = "\n")
cat(
">No304",
"ATTCGAAAAACACACCCACTACTAAAAATTATCAACCACT",
file = "dna2.fas", sep = "\n")
cat(
">No305",
"NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT",
file = "dna3.fas", sep = "\n")

file.cat(dir = getwd(), appendix = ".fas", file = "total")

unlink("dna1.fas")
unlink("dna2.fas")
unlink("dna3.fas")
unlink("total.fas")
```

findprobe *Find probes matching sites*

Description

This function could be used to search probes (primer) matching sites. Users may lower the tol value to get more precise match result.

Usage

```
findprobe(dna, probe, tol = 3)
```

Arguments

dna	The input DNA sequence.
probe	The probe(primer) to match the DNA sequence.
tol	The number of sites that do not match

Details

`dna` should be a character string. `tol` is used to adjust the matching precision and should be integer. The smaller the number, the more precise the matching results. This function is called by `clipprobe()` for searching the sequences that could be used in PCR.

Value

Returns a vector containing the matched sites.

Note

To be added.

Author(s)

Qiong Ding <dingqiong1@gmail.com>

References

None.

See Also

See Also clipprobe

Examples

```
### findprobe() example
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
findprobe(dna = fas[2], probe = "TATTTAAC", tol = 1)
```

---

frag.dat *Reports for simulated RFLP cutting pattern*

Description

Generating reports of simulated RFLP cutting patterns.

Usage

```
frag.dat(fil, enznames, enzdata)
```

Arguments

fil	sequence data in class fasta.
enznames	selected enzyme name from enzdata.
enzdata	a dataframe contained enzyme information.

Details

Given the name of restriction enzyme, the function will give the simulated RFLP pattern. The input sequences must be a fasta format object, which means it must be first converted to class "fasta". Users are encouraged to use read.fasta, read.phy, read.nxs to read files from local machine, and converted the data to class fasta using as.fasta.

Value

This function returns to a dataframe which including the following columns. with rownames the sequence name.

enznames	enzyme name
recogSite	The recognition site of the specified enzyme.
cutting_Site	The cutting site number on the sequence from 5' to 3'.
fragment_Length	The length of each fragments from 5' to 3'
T5	Length of the 5' terminal fragment from the original sequence.
T3	Length of the 3' terminal fragment from the original sequence.

Note

To be added.

Author(s)

Qiong Ding <dingqiong1@gmail.com>

References

None.

Examples

```
### fragdat() example
data(enzdata)
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")
frag.dat(fil = fas, enznames = "EcoRI", enzdata = enzdata)
```

---

gnames.fas *Get the names of sequences from fasta objects.*

Description

This function could be used to obtain the names of DNA sequences from fasta objects.

Usage

```
gnames.fas(x = NULL)
```

Arguments

`x`	Object with class `"fasta"`.

Details

To be added.

Value

Returns the names of sequences stored in a vector of character strings.

Note

To be added

Author(s)

Jinlong zhang `<jinlongzhang01@gmail.com>`

References

None.

See Also

See Also `rename.fas` for renaming the names of the DNA sequences.

Examples

```
### gnames.fas() example
data(fil.fas)
gnames.fas(fil.fas)
```

plotenz *Plotting the simulated RFLP or TRFLP pattern*

Description

Plotting the simulated RFLP or TRFLP pattern using selected restriction enzymes.

Usage

```
plotenz(sequences, enznames, enzdata, side = TRUE, type = c("RFLP", "TRFLP"), Terminal = c("T5", "T3")
```

Arguments

`sequences` The input data in class `fasta` containing the DNA sequences.

`enznames` The speciefied restriction enzyme names to be applied in RFLP or TRFLP analysis.

`enzdata` The dataframe contained enzyme data.

`side` Whether to plot the markers for each sequence. Default is TURE which means to plot the marker only once.

`type` Pattern type to be specified, should be either `"RFLP"` or `"TRFLP"`.

`Terminal` Terminal noted in `"TRFLP"` analysis, should be either `"T3"` or `"T5"`.

Details

If `type = "TRFLP"` (The `"TRFLP"` were selected), the `Terminal` must also been provided by the user, it's value is either "T3" or "T5".

Value

Returns the plot of simulated RFLP or TRFLP.

Author(s)

Qiong Ding <`dingqiong1@gmail.com`>

References

None.

See Also

See Also `frag.dat` for the summary of RFLP results.

Examples

```
## plotenz() example

data(enzdata)
data(fil.phy)
fas <- ConvFas(fil = fil.phy, type = "phy")

enznames <- c("EcoRI", "Acc65I", "HinfI")

plotenz(sequences = fas, enznames = enznames,
        enzdata = enzdata, side = TRUE, type = "RFLP")

plotenz(sequences = fas, enznames = enznames,
        enzdata = enzdata, side = FALSE, type = "RFLP")

plotenz(sequences = fas, enznames = enznames,
          enzdata = enzdata, side = TRUE, type = "TRFLP", "T3")
```

raw2Fas *Read and converting raw DNA files to fasta format.*

Description

Given the specified directory, this function could read and convert raw DNA files to fasta format. It is an alternative to `read.fasta` as the later read only one fasta file. The advantage of using `raw2Fas` when dealing with fasta files rather than using `file.cat` is that this function would convert the fasta files to the fasta in a robost way.

Usage

```
raw2Fas(dir = NULL, appendix = ".fasta")
```

Arguments

`dir`	a character string naming the directory fasta files exist.
`appendix`	Appendix of raw DNA data files to combine.

Details

Only the file possesses the the specied appendix will be read and converted.

Value

Returns the object of class "fasta".

Author(s)

Jinlong zhang <`jinlongzhang01@gmail.com`> Qiong Ding <`dingqiong1@gmail.com`>

References

None.

See Also

See Also `read.fasta`, `file.cat` for importing `"fasta"` files from local machine.

Examples

```
#########
######
### raw2Fas() example
cat(
">No305”,
"NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT”,
file = "dna1.fas”, sep = "\n”)
cat(
">No304”,
"ATTCGAAAAACACACCCACTACTAAAAATTATCAACCACT”,
file = "dna2.fas”, sep = "\n”)
cat(
">No305”,
"NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT”,
file = "dna3.fas”, sep = "\n”)

raw2Fas(getwd(), appendix = ".fas”)

unlink("dna1.fas”)
unlink("dna2.fas”)
unlink("dna3.fas”)
```

read.fasta *Read fasta file*

Description

Read fasta file from a specified file path or URL.

Usage

```
read.fasta(file = NULL)
```

Arguments

file	A character string naming the file path.

Details

Read fasta file from a specified file path (usually local machine) or URL.

Value

Returns a fasta object stored as a vector of character strings.

Note

To add.

Author(s)

Jinlong zhang <jinlongzhang01@gmail.com>

References

None.

See Also

See Also read.phy, read.nxs for importing data from other data types.

Examples

```
### read.fasta() example
###
cat(
">No305",
"NTTCGAAAAACACACCCACTACTAAAANTTATCAGTCACT",
file = "dna1.fas", sep = "\n")

sequences <- read.fasta("dna1.fas")

unlink("dna1.fas")
```

read.nxs *Read nexus file*

Description

Function to read nexus file.

Usage

```
read.nxs(fil = NULL)
```

Arguments

fil	A character string naming the file path.

Details

Read nexus file from a specified file path (usually local machine) or URL.

Value

Returns a nexus "nxs" object stored as a vector of character strings.

Note

To add.

Author(s)

Jinlong zhang <jinlongzhang01@gmail.com>

References

None

See Also

See Also read.phy, read.fasta for importing data from other data types.

Examples

```
#####
##### read.nxs() example
data(fil.nxs)
writeLines(fil.nxs, "example.nxs")
nxs <- read.nxs("example.nxs")
as.fasta(nxs)
unlink("example.nxs")
```

---

read.phy *Read phylip file*

Description

Function to read phylip file.

Usage

```
read.phy(fil = NULL)
```

Arguments

fil	A character string naming the file path.

Details

Read phylip file from a specified file path (usually local machine) or URL.

Value

Returns a nexus "nxs" object stored as a vector of character strings.

Note

To be added.

Author(s)

Jinlong zhang <jinlongzhang01@gmail.com>

References

None.

See Also

See Also read.nxs, read.fasta for importing data from other data types.

Examples

```
##### read.phy() example
data(fil.phy)
writeLines(fil.phy, "example.phy")
phy <- read.phy("example.phy")
as.fasta(phy)
unlink("example.phy")
```

---

rename.fas *Change the sequence names of fasta objects.*

Description

This function could be used to change the names of the DNA squences according to given names. If no names provided by names, the function will return the original sequences.

Usage

```
rename.fas(x, names = NULL)
```

Arguments

x	fasta object to be renamed.
names	A vector of character strings indicating the names..

Details

The number of names provided by the user must be equal to the number of sequences, otherwise the function will stop.

Value

A fasta object with all the sequences with new names.

Note

To be added.

Author(s)

Jinlong zhang <jinlongzhang01@gmail.com>

References

None.

See Also

See Also gnames.fas, for obtaining the names of sequences from fasta objects.

Examples

```
### rename.fas() example
data(fil.fas)
rename.fas(fil.fas, name = paste("Sequence", as.character(1:19), sep = ""))
```

---

revComp *Reverse complement sequence*

Description

Given an segment of DNA sequence, this function will give the reverse complement sequence.

Usage

```
revComp(dna)
```

Arguments

dna	A character string containing the input DNA sequence.

Details

Imcompletely specified base in DNA sequences must use the standard abbreviations:

```
R   = G or A
 Y   = C or T
 M   = A or C
 K   = G or T
 S   = G or C
 W   = A or T
 B   = not A (C or G or T)
 D   = not C (A or G or T)
 H   = not G (A or C or T)
 V   = not T (A or C or G)
 N   = A or C or G or T
```

Value

Returns to the reverse complement sequence.

Note

None.

Author(s)

Qiong Ding <dingqiong1@gmail.com>

References

None.

Examples

```
### revComp() example
revComp("TTGAACC")
```

selEnz *Selecting restriction enzyme*

Description

Function for selecting restriction enzyme.

Usage

```
selEnz(names, enzdata = enzdata)
```

Arguments

`names`	A character string indicating the restriction enzyme name.
`enzdata`	A dataframe including enzyme data.

Details

`enzdata` must be a dataframe with the first column enzyme name, the second column the restriction clipping sites.

Value

Returns to a dataframe with only the selected enzyme information.

Note

To be added.

Author(s)

Jinlong zhang <jinlongzhang01@gmail.com> Qiong Ding <dingqiong1@gmail.com>

References

None.

Examples

```
## selEnz() example
data(enzdata)
selEnz(enzdata = enzdata, names = "EcoRI")
```

`write.fasta` *Write fasta format object to file*

Description

To save the fasta format object to speciefied file.

Usage

```
write.fasta(sequences, file = NULL)
```

Arguments

`sequences`	The fasta object to be saved.
`file`	A character string naming the file to be saved to.

Details

sequences must be an object of class fasta.

Value

Saved fasta file.

Author(s)

Jinlong Zhang <jinlongzhang01@gmail.com>

References

None.

See Also

See Also read.fasta

Examples

```
data(fil.fas)
write.fasta(fil.fas, "example.fasta")
## Remove the file.
unlink("example.fasta")
```

Index