Package 'seq2R'

January 6, 2023

Type Package

Title Simple Method to Detect Compositional Changes in Genomic Sequences

Version 1.1.0

Date 2022-12-16

Maintainer Nora M. Villanueva <nmvillanueva@uvigo.es>

Depends R (>= 2.15.1)

Description This software is useful for loading '.fasta' or '.gbk' files, and for retrieving sequences from 'GenBank' dataset <https://www.ncbi.nlm.nih.gov/genbank/>. This package allows to detect differences or asymmetries based on nucleotide composition by using local linear kernel smoothers. Also, it is possible to draw inference about critical points (i. e. maximum or minimum points) related with the derivative curves. Additionally, bootstrap methods have been used for estimating confidence intervals and speed computational techniques (binning techniques) have been implemented in 'seq2R'.

Imports seqinr

License GPL

LazyData true

RoxygenNote 7.1.2

NeedsCompilation yes

Author Nora M. Villanueva [aut, cre] (<https://orcid.org/0000-0001-8085-2745>),
Marta Sestelo [aut] (<https://orcid.org/0000-0003-4284-6509>),
Alan Miller [ctb] (FORTRAN code lsq.f90: weighted least-squares module)

Repository CRAN

Date/Publication 2023-01-06 11:10:02 UTC

R topics documented:

`seq2R-package` *Simple method to detect compositional changes in genomic sequences*

Description

`seq2R` is just a shortcut for "sequence to `R`". This software is useful for loading .fasta or .gbk files, and for recovering sequences from GenBank database. This package allows to detect differences or asymmetries based on nucleotide composition by using local linear kernel smoothers. Also, it is possible to draw inference about critical points (i. e. maximum or minimum points) related with the derivative curves. Additionally, bootstrap methods have been used for estimating confidence intervals and speed computational techniques (binning techniques) have been implemented in "seq2R".

Details

Package:	seq2R
Type:	Package
Version:	1.0
Date:	2012-01-08
License:	GPL
LazyLoad:	yes

In the package are included several functions that enable users to analyze asymmetries based on nucleotide composition of the DNA sequences. It is useful for loading different types of files (`read.all` function), and for retrieving sequences from GenBank dataset (`read.genbank` function). After reading the sequence, it is necessary to convert the character vector into a binary code, by applying `change.binary` function. At this point, it is possible to apply the `change.points` function, for fitting nonparametric regression models and obtaining the estimates and their first derivatives. The object obtained with this function is the argument required for `plot.change.points` function which provides a graphical output. Finally, to determine the position of the critical points (maxima or minima), only applying `critical` function can be obtained.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Maintainer: Nora M. Villanueva <nmvillanueva@uvigo.es>

References

Efron, B. (1979). Bootstrap methods: another look at the jackknife. Annals of Statistics, 7:1-26.

Efron, E. and Tibshirani, R. J. (1993). An introduction to the Bootstrap. Chapman and Hall, London.

Fan, J. and Marron, J.S. (1994). Fast implementation of nonparametric curve estimators. Journal of Computational and Graphical Statistics, 3:35-56.

Hastie, T. and Tibshirani, R. (1990). Generalized additive models. Chapman and Hall, London.

Stone, C. J. (1977). Consistent nonparametric regression. Annals of Statistics, 5: 595-620.

Wand, M. P. and Jones, M. C. (1995). Kernel Smoothing. Chapman and Hall, London.

`change.binary` *Convert biological sequences into binary code*

Description

Biological sequences are categorical variables. With this function the four nucleotides are coded with two bits, 0 and 1 (binary numeral system) for being used by almost all modern computers.

Usage

```
change.binary(x)
```

Arguments

`x`	The object obtained with `read.all` or `read.genbank` functions is the argument required for `change.binary`. The nature of the sequence is DNA. Sequences are returned as a vector of single characters.

Value

The returned list has two component (`$AT`, `$CG`). Both of them containing a matrix with values about their critical points (maximum and minimum), and their lower and upper 95% confidence intervals.

`AT`	Variable A and T with binary system.
`CG`	Variable C and G with binary system.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)

mthumanDNA <- read.genbank("NC_012920")
DNA <- change.binary(mthumanDNA)
DNA
```

`change.points` *Simple method to detect compositional changes in genomic sequences*

Description

`change.points` is used to detect changes at genomic sequences composition. The method is based on fitting nonparametric models by using local linear kernel smoothers.

Usage

```
change.points(x,kbin= 300, p= 3, h=-1, W= 1, nboot=100, kernel="gaussian",
nh= 20, seed = NULL, ...)
```

Arguments

`x`	Sequences in binary system (by using `change.binary` function previously) are to be analyzed from.
`kbin`	The number of binning nodes over which the function is to be estimated.
`p`	Degree of the polynomial. By default `p=3`.
`h`	The kernel bandwidth or smoothing parameter. Large values of bandwidth make smoother estimates, smaller values of bandwidth make less smooth estimates. The default `h=-1` is a bandwidth compute by cross validation.
`W`	Weights.
`nboot`	Number of bootstrap repeats.
`kernel`	Character which denotes the kernel function (a symmetric density). By default `kernel = "gaussian"`, this is, the Gaussian density function. Also, other types of kernel functions can be used: Epanechnikov and triangle, `kernel="Epanech"` and `kernel="triang"`, respectively.
`nh`	Number that it will be used to calculate the grid of bandwidths in a range between 0 and 1. In this grid, it will be selected the optimum bandwidth by cross-validation.If the optimum bandwidth value is close to 0, we will obtain rough estimates; when it is close to 1, we will obtain smooth estimates.
`seed`	Seed to be used in the bootstrap procedure.
`...`	Other options.

Details

For each genomic sequence the AT and CG skews profiles were calculated as $Avs.T = (A - T)/(A + T)$ and $Cvs.G = (C - G)/(C + G)$.

Value

The function computes and returns a list of short information for a fitted `change.points` object.

`Number of A-T base pairs`
: The returned value is the total nucleotide (adenine and thymine) contained at the sequence analyzed.

`Number of C-G base pairs`
: In this case, the returned value is the sum of cytosine and guanine contained at the sequence.

`Number of binning nodes`
: The number of binning nodes over which the function is to be estimated.

`Number of bootstrap repeats`
: Number of bootstrap repeats.

`Bandwidth`
: Value of the kernel bandwidth or smoothing parameter used in the fitting for A vs. T and C vs. G.

`Exists any critical point`
: Emphasize if there is or not any critical.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)


mtDNAhuman <- read.genbank("NC_012920")
DNA<- change.binary(mtDNAhuman)
seq1<-change.points(DNA)
seq1
```

`critical` *Critical points (maxima and minima)*

Description

Function that maximizes or minimizes the first derivative of the model obtained with `change.points` function. Also, it is included their 95% confidence intervals.

Usage

```
critical(model, base.pairs = NULL)
```

Arguments

`model`	`change.points` object.
`base.pairs`	Character string for A vs. T or C vs G.

Value

The returned list has two component ($AT, $CG). Both of them containing a matrix with values about their critical points (maxima and minima), lower and upper 95% confidence intervals.

AT	Critical points for AT.
CG	Critical points for CG.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)

mthumanDNA <- read.genbank("NC_012920")
DNA <- change.binary(mthumanDNA)
seq1 <- change.points(DNA)


critical(seq1,base.pairs="CG")


critical(seq1,base.pairs="AT")
```

---

mtDNAhum *Human Mitochondrial DNA*

Description

The complete sequence of the human mitochondrial genome contains 16569 base pair. The sequence presents extreme economy in that the genes have none or only a few noncoding bases between them, and in many cases the termination codons are not coded in the DNA but are created post-transcriptionally by polyadenylation of the mRNAs. The genes for the 12S and 16S rRNAs, 22tRNAs, cytochrome c oxidase subunits I, II, and III, ATPase subunit 6, cytochrome b and eight other predicted protein coding genes have been located.

Usage

```
data(mtDNAhum)
```

References

Anderson, S. and Bankier, A. T. and Barrell, B. G. and de Bruijn, M. H. L. and Coulson, A. R. and Drouin, J. and Eperon, I. C. and Nierlich, D. P. and Roe, B. A. and Sanger, F. and Schreier, P. H. and Smith, A. J. H. and Staden, R. and Young, I. G.(1981) Sequence and organization of the human mitochondrial genome. Nature, 5806(290):457:465

Examples

```
data(mtDNAhum)
```

`plot.change.points` *Visualization of change.points objects*

Description

Useful for drawing the estimation and first derivative of the skew profile.

Usage

```
## S3 method for class 'change.points'
plot(x = model, y = NULL, base.pairs = NULL, der = NULL,
xlab = "x", ylab = "y", col = "black", CIcol = "black", main = NULL, type = "l",
CItype = "l", critical = FALSE, CIcritical = FALSE,ylim=NULL,...)
```

Arguments

`x`	`change.points` object.
`y`	`NULL`
`base.pairs`	Character string about the skew profile for A vs. T or C vs. G.
`der`	Number which determines inference process to be drawing into the plot. By default `der` is `NULL`. If it is `0`, the plot represents the initial estimate. If `der` is `1`, the first derivative is plotted.
`xlab`	Title for x axis.
`ylab`	Title for y axis.
`col`	A specification for the default plotting color.
`CIcol`	A specification for the default confidence intervals plotting color.
`main`	An overall title for the plot.
`type`	Type of plot should be drawn. Possible types are, `p` for points, `l` for lines, `o` for overplotted, etc. See details in `?par`.
`CItype`	Type of plot should be drawn for confidence intervals. Possible types are, `p` for points, `l` for lines, `o` for overplotted, etc. See details in `?par`.
`critical`	A logical value. If `TRUE` (not by default), the critical points are drawn into the plot.
`CIcritical`	A logical value. If `TRUE` (not by default), the 95% confidence intervals for the critical points are drawn into the plot.
`ylim`	The y limits of the plot.
`...`	Other options.

Value

Simply produce a plot.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)


mtDNAhuman <- read.genbank("NC_012920")
DNA<- change.binary(mtDNAhuman)
seq1<-change.points(DNA)



plot(seq1,der=0,base.pairs="CG",CIcritical=TRUE,ylim=c(0.08,0.67))
plot(seq1,der=1,base.pairs="CG",CIcritical=TRUE,ylim=c(-0.0005,0.00045))
abline(h=0)



plot(seq1,critical=TRUE, CIcritical=TRUE)
```

---

print.change.points *Short change.points summary*

Description

change.points summary.

Usage

```
## S3 method for class 'change.points'
print(x=model,...)
```

Arguments

x	change.points object.
...	Other options.

Value

The function computes and returns a list of short information for a fitted `change.points` object.

`Number of A-T base pairs`
: The returned value is the total nucleotide (adenine and thymine) contained in the sequence analyzed.

`Number of C-G base pairs`
: In this case, the returned value is the sum of cytosine and guanine contained at the sequence.

`Number of binning nodes`
: The number of binning nodes over which the function is to be estimated.

`Number of bootstrap repeats`
: Number of bootstrap repeats.

`Bandwidth`
: Value of the Kernel bandwidth or smoothing parameter used in the fiiting for A vs. T and C vs. G.

`Exists any critical point`
: Emphasize if there is or not any critical.

Note

See details in `change.points`.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)
mtDNAhuman <- read.genbank("NC_012920")
DNA<- change.binary(mtDNAhuman)
seq1<-change.points(DNA)
seq1
```

---

read.all *Read FASTA and GBK formatted files*

Description

Read nucleic acid sequences from a file in FASTA or GBK format.

Usage

```
read.all(file = system.file(""), seqtype = "DNA")
```

Arguments

`file`	The name of the file which the sequences in FASTA or GBK format are to be read from.
`seqtype`	The nature of the sequence. Nowadays only `DNA`, in further updates it will be possible to use for different type of sequences.

Details

Fasta is a widely used format in molecular biology. Sequence in FASTA format starts with a single-line description, distinguished by a greater-than '>' symbol, followed by sequence data on the next lines.

'GenBank' format files have the extension GBK, by convention. Files contain fields with different types of information well-labeled. The header of the file has information describing the sequence, such as its type, shape, length and source. The features of the genome sequence follow the header, and include protein translations. The DNA sequence is the last element of the file, which ends with (and must include) a soluble slash. Complete genomes in this format are available at the https://ftp.ncbi.nlm.nih.gov/genbank/.

Value

`Sequence`	The returned list has a component `Sequence` containing the DNA sequence taken from the field "ORIGIN" in GenBank. The sequence is a vector of single characters.
`Locus or accession`	the returned list has a component `Locus/Accession` containing the names of the locus or accession number taken from the field "LOCUS" or "ACCESSION" in 'GenBank'. Also, return sequence length.

Author(s)

Nora M. Villanueva and Marta Sestelo.

Examples

```
library(seq2R)
data(mtDNAhum)
## Not run:
data<-read.all("file.fasta")
data<-read.all("file.gbk")

## End(Not run)
```

`read.genbank` *Read DNA sequences from GenBank via internet*

Description

This function connects to the GenBank database, and reads nucleotide sequences using locus code given as arguments.

Usage

```
read.genbank(locus)
```

Arguments

`locus`	Vector of mode character giving the locus code or accession number.

Details

This function uses the site https://pubmed.ncbi.nlm.nih.gov/ (E-utilities) from where the sequences are downloaded. E-utilities are a set of eight server-side programs that provide a stable interface into the Entrez query and database system at the National Center for Biotechnology Informatio (NCBI). The E-utilities use a fixed URL syntax that translates a standard set of input parameters into the values necessary for various NCBI software components to search for and retrieve the requested data. The E-utilities are therefore the structured interface to the Entrez system, which currently includes 38 databases covering a variety of biomedical data, including nucleotide and protein sequences, gene records, three-dimensional molecular structures, and the biomedical literature.

Value

`Sequence`	The returned list has a component `Sequence` containing the DNA sequence taken from the field "ORIGIN" in GenBank. The sequence is a vector of single characters.
`Locus or accession`	The returned list has a component `Locus/Accession` containing the names of the locus or accession number taken from the field "LOCUS" or "ACCESSION" in GenBank.
`Species`	The returned list has an attribute `Species` containing the names of the species taken from the field "ORGANISM" in GenBank.

Note

If the computer is not connected to the internet, this function will not work.

Author(s)

Nora M. Villanueva and Marta Sestelo.

References

Bethesda M. D. (2010) Entrez Programming Utilities Help. NCBI Help Manual. NCBI, USA

Examples

```
library(seq2R)
mthumanDNA <- read.genbank("NC_012920")
mthumanDNA
```

Index