Package: rMSA (via r-universe)

September 19, 2024

Title Interface for Popular Multiple Sequence Alignment Tools

Description Seamlessly interfaces the Multiple Sequence Alignment software packages ClustalW, MAFFT, MUSCLE and Kalign (downloaded separately) and provides support to calcualte distances between sequences. This work was partially supported by grant no. R21HG005912 from the National Human Genome Research Institute.

Version 0.99.1

Date 2024-05-22

Author Michael Hahsler, Anurag Nagar

Maintainer Michael Hahsler <mhahsler@lyle.smu.edu>

biocViews Genetics, Sequencing, Infrastructure, Alignment

Depends Biostrings (>= 2.26.2)

Imports methods, pwalign, seqLogo, proxy, ape

SystemRequirements ClustalW, Kalign, MAFFT, MUSCLE, boxshade

License GPL-3

Repository https://mhahsler.r-universe.dev

RemoteUrl https://github.com/mhahsler/rMSA

RemoteRef HEAD

RemoteSha 4772797ab600332496167a20791f0a26c3b711e3

Contents

`boxshade` *Boxshade: Shading Multiple Aligned Sequences*

Description

Executes boxshade on a multiple sequence alignment.

Usage

```
boxshade(x, file, dev="pdf", param="-thr=0.5 -cons -def",
    pdfCrop=TRUE)
boxshade_help()
```

Arguments

`x`	a multiple alignment as an object of class DNAMultipleAlignment, RNAMultipleAlignment or AAMultipleAlignment.
`file`	output file
`dev`	used output device. Available are: ps, eps, hpgl, rtf, crt, ansi, vt, ascii, fig, pict, html and pdf.
`param`	character string with the command line parameters for clustal (see output of `boxshade_help()`).
`pdfCrop`	crop the pdf file if it is smaller than a page. Use `FALSE` if you want the results on a page or the alignment covers multiple pages.

Details

For installation details see: https://github.com/mhahsler/rMSA/blob/master/INSTALL

Value

Only a file is created.

Author(s)

Michael Hahsler

References

Boxshade has been written by Kay Hofmann and Michael D. Baron

Examples

```
## Not run:
rna <- readRNAStringSet(system.file("examples/RNA_example.fasta",
 package="rMSA"))
rna <- narrow(rna, start=1, end=50)

al <- clustal(rna)

boxshade(al, file="alignment.pdf", dev="pdf")

## End(Not run)
```

---

clustal *Run Multiple Sequence Alignment (ClustalW) on a Set of Sequences*

Description

Executes Clustal on a set of sequences to obtain a multiple sequence alignment.

Usage

```
clustal(x, param)
clustal_help()
```

Arguments

x	an object of class XStringSet (e.g., DNAStringSet) with the sequences to be aligned.
param	character string with the command line parameters for clustal (see output of `clustal_help()`).

Details

For installation details see: https://github.com/mhahsler/rMSA/blob/master/INSTALL

Value

An object of class DNAMultipleAlignment (see **BioStrings**).

Author(s)

Michael Hahsler

References

Larkin M., et al. Clustal W and Clustal X version 2.0, Bioinformatics 2007 23(21):2947-29

Examples

```
## Not run:
### DNA
dna <- readDNAStringSet(system.file("examples/DNA_example.fasta",
 package="rMSA"))
dna

al <- clustal(dna)
al

### inspect alignment
detail(al)

### plot a sequence logo for the first 20 positions
plot(al, 1, 20)

### RNA
rna <- readRNAStringSet(system.file("examples/RNA_example.fasta",
 package="rMSA"))
rna

al <- clustal(rna)
al

### Proteins
aa <- readAAStringSet(system.file("examples/Protein_example.fasta",
 package="rMSA"))
aa

al <- clustal(aa)
al

## End(Not run)
```

`dist` *Calculate Distances between Sets of Sequences*

Description

Implements different methods to calculate distance between sets of sequences based on k-mer distribution, edit distance/alignment or evolutionary distance.

Usage

```
# k-mer-based methods
distFFP(x, k=3, method="JSD", normalize=TRUE)
distCV(x, k=3)
distNSV(x, k=3, method="Manhattan", normalize=FALSE)
distKMer(x, k=3)
```

```
distSimRank(x, k=7)

# edit distance/alignment
distEdit(x)
distAlignment(x, substitutionMatrix=NULL, ...)

# evolutionary distance
distApe(x, model="K80" ,...)
```

Arguments

`x`	an object of class XStringSet containing the sequences. For `distApe`, `x` needs to be a multiple sequence alignment.
`k`	size of used k-mers.
`method`	metric used to calculate the dissimilarity between two k-mer frequency distributions.
`substitutionMatrix`	matrix with substitution scores (defaults to a matrix with match=1, mismatch=0)
`normalize`	normalize the k-mer frequencies by the total number of k-mers in the sequence.
`model`	evolutionary model used.
`...`	further arguments passed on.

Details

- *Feature frequency profile* (`distFFP`): A FFP is the normalized (by the number of k-mers in the sequence) count of each possible k-mer in a sequence. The distance is defined as the Jensen-Shannon divergence (JSD) between FFPs (Sims and Kim, 2011).
- *Composition Vector* (`distCV`): A CV is a vector with the frequencies of each k-mer in the sequency minus the expected frequency of random background of neutral mutations obtained from a Markov Model. The cosine distance is used between CVs. (Qi et al, 2007).
- *Numerical Summarization Vector* (`distNSV`): An NSV is frequency distribution of all possible k-mers in a sequence. The Manhattan distance is used between NSVs (Nagar and Hahsler, 2013).
- *Distance between sets of k-mers* (`distkMer`): Each sequence is represented as a set of k-mers. The Jaccard (binary) distance is used between sets (number of unique shared k-mers over the total number of unique k-mers in both sequences).
- *Distance based on SimRank* (`distSimRank`): 1-simRank (see `simRank`).
- *Edit (Levenshtein) Distance* (`distEdit`): Edit distance between sequences.
- *Distance based on alignment score* (`distAlignment`): see `stringDist` in **Biostrings**.
- *Evolutionary distances* (`distApe`): see `dist.dna` in **ape**.

Value

A dist object.

Author(s)

Michael Hahsler

References

Sims, GE; Kim, SH (2011 May 17). "Whole-genome phylogeny of Escherichia coli/Shigella group by feature frequency profiles (FFPs).". Proceedings of the National Academy of Sciences of the United States of America 108 (20): 8329-34. PMID 21536867.

Gao, L; Qi, J (2007 Mar 15). "Whole genome molecular phylogeny of large dsDNA viruses using composition vector method.". BMC evolutionary biology 7: 41. PMID 17359548.

Qi J, Wang B, Hao B: Whole Proteome Prokaryote Phylogeny without Sequence Alignment: A K-String Composition Approach. Journal of Molecular Evolution 2004, 58:1-11.

Anurag Nagar; Michael Hahsler (2013). "Fast discovery and visualization of conserved regions in DNA sequences using quasi-alignment." BMC Bioinformatics, 14(Suppl. 11), 2013

Examples

```
s <- mutations(random_sequences(100), 100)
s

### calculate NSV distance
dNSV <- distNSV(s)

### relationship with edit distance
dEdit <- distEdit(s)

df <- data.frame(dNSV=as.vector(dNSV), dEdit=as.vector(dEdit))
plot(sapply(df, jitter), cex=.1)
### add lower bound (2*k, for Manhattan distance)
abline(0,1/(2*3), col="red", lwd=2)
### add regression line
abline(lm(dEdit~dNSV, data=df), col="blue", lwd=2)

### check correlation
cor(dNSV,dEdit)
```

---

kalign — *Multiple Sequence Alignment (Kalign)*

Description

Runs Kalign progressive multiple sequence alignment on a set of sequences.

Usage

```
kalign(x, param=NULL)
kalign_help()
```

Arguments

x	an object of class DNAStringSet with the sequences to be aligned.
param	character string with the command line parameters for kalign (see output of `kalign_help()`).

Details

For installation details see: https://github.com/mhahsler/rMSA/blob/master/INSTALL

Value

An object of class DNAMultipleAlignment (see **BioStrings**).

Author(s)

Michael Hahsler

References

Lassmann T., Sonnhammer E. Kalign - an accurate and fast multiple sequence alignment algorithm, BMC Bioinformatics 2005, 6:298

Examples

```
## Not run:
dna <- readDNAStringSet(system.file("examples/DNA_example.fasta",
 package="rMSA"))
dna

### align the sequences
al <- kalign(dna)
al

## End(Not run)
```

---

mafft — *Run Multiple Sequence Alignment (MAFFT) on a Set of Sequences*

Description

Executes mafft on a set of sequences to obtain a multiple sequence alignment.

Usage

```
mafft(x, param="--auto")
mafft_help()
```

Arguments

x	an object of class XStringSet (e.g., DNAStringSet) with the sequences to be aligned.
param	character string with the command line parameters (see output of `mafft_help()`).

Details

For installation details see: https://github.com/mhahsler/rMSA/blob/master/INSTALL

Value

An object of class DNAMultipleAlignment (see **BioStrings**).

Author(s)

Michael Hahsler

References

Katoh, Standley 2013 (Molecular Biology and Evolution 30:772-780) MAFFT multiple sequence alignment software version 7: improvements in performance and usability.

Examples

```
## Not run:
### DNA
dna <- readDNAStringSet(system.file("examples/DNA_example.fasta",
 package="rMSA"))
dna

al <- mafft(dna)
al

### inspect alignment
detail(al)

### plot a sequence logo for the first 20 positions
plot(al, 1, 20)

### RNA
rna <- readRNAStringSet(system.file("examples/RNA_example.fasta",
 package="rMSA"))
rna

al <- mafft(rna)
al

### Proteins
aa <- readAAStringSet(system.file("examples/Protein_example.fasta",
 package="rMSA"))
aa
```

```
al <- mafft(aa)
al

## End(Not run)
```

MUSCLE *Run Multiple Sequence Alignment (MUSCLE) on a Set of Sequences*

Description

Executes MUSLCE on a set of sequences to obtain a multiple sequence alignment.

Usage

```
muscle(x, param="")
muscle_help()
```

Arguments

x	an object of class XStringSet (e.g., DNAStringSet) with the sequences to be aligned.
param	character string with the command line parameters (see output of `musce_help()`).

Details

For installation details see: https://github.com/mhahsler/rMSA/blob/master/INSTALL

Value

An object of class DNAMultipleAlignment (see **BioStrings**).

Author(s)

Michael Hahsler

References

Edgar, R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput, Nucleic Acids Res. 32(5):1792-1797

Edgar, R.C. (2004) MUSCLE: a multiple sequence alignment method with reduced time and space complexity, BMC Bioinformatics, (5) 113

Examples

```
## Not run:
### DNA
dna <- readDNAStringSet(system.file("examples/DNA_example.fasta",
 package="rMSA"))
dna

al <- muscle(dna)
al

### inspect alignment
detail(al)

### plot a sequence logo for the first 20 positions
plot(al, 1, 20)

### RNA
rna <- readRNAStringSet(system.file("examples/RNA_example.fasta",
 package="rMSA"))
rna

al <- MUSCLE(rna)
al

### Proteins
aa <- readAAStringSet(system.file("examples/Protein_example.fasta",
 package="rMSA"))
aa

al <- MUSCLE(aa)
al

## End(Not run)
```

`mutations` *Creates Random Mutations of a Sequence*

Description

Creates a set of sequences which are random mutations (with base changes, insertions and deletions) for a given DNA, RNA or AA sequence.

Usage

```
mutations(x, number=1, change=0.01, insertion=0.01, deletion=0.01, prob=NULL)
```

Arguments

`x`	A XString or an XStringSet of length 1.

`number` number of sequences to create.

`change`, `insertion`, `deletion`
probability of this operation.

`prob` a named vector with letter probabilities. 4 for DNA and RNA and 20 for AA (see `DNA_BASES`, `RNA_BASES` and the first 20 letters in `AA_ALPHABET`). The default is to estimate the probabilities from the sequence in x.

Value

A XStringSet.

Author(s)

Michael Hahsler

Examples

```
### create random sequences
s <- random_sequences(100, number=1)
s

### create 10 sequences with 1 percent base changes, insertions and deletions
m <- mutations(s, 10, change=0.01, insertion=0.01, deletion=0.01)
m

### calculate edit distance between the original sequence and the mutated
### sequences
stringDist(c(s,m))

### multiple sequence alignment
## Not run:
al <- clustal(c(s,m))
detail(al)

## End(Not run)
```

`plot` *Plot Genetic Sequences and Alignments*

Description

Plots genetic sequences (RNA/DNA) using sequence logos.

Details

`plot` creates a sequence logo. Parameters are `start` (position to start the logo), end (position to end the logo), `ic.scale` (if TRUE then each column are scaled proportional to its information content).

See Also

seqLogo in **seqLogo**.

---

random_sequences *Create a Set of Random Sequences*

Description

Creates a set of random DNA, RNA or AA sequences.

Usage

```
random_sequences(len, number=1, prob=NULL, type=c("DNA", "RNA", "AA"))
```

Arguments

len	sequence length
number	number of sequences in the set
prob	a named vector with letter probabilities or a transition probability matrix (as produced by oligonucleotideTransitions). 4 letters for DNA and RNA and 20 for AA (see DNA_BASES, RNA_BASES and the first 20 letters in AA_ALPHABET).
type	sequence type

Value

A XStringSet.

Author(s)

Michael Hahsler

Examples

```
### create random sequences (using given letter frequencies)
seqs <- random_sequences(100, number=10, prob=c(a=.5, c=.3, g=.1, t=.1))
seqs

### check letter frequencies
summary(oligonucleotideFrequency(seqs, width=1, as.prob=TRUE))

### creating random sequences using a random dinocleodite transition matrix
prob <-  matrix(runif(16), nrow=4, ncol=4, dimnames=list(DNA_BASES, DNA_BASES))
prob <- prob/rowSums(prob)

seqs <- random_sequences(100, number=10, prob=prob)
seqs

### check dinocleodite transition probabilities
```

```
prob
oligonucleotideTransitions(seqs, as.prob=TRUE)
```

---

simRank *Compute the SimRank Similarity between Sets of Sequences*

Description

Computes the SimRank similarity (number of shared unique k-mers over the smallest number of unique k-mers.)

Usage

```
simRank(x, k = 7)
```

Arguments

x	an object of class DNAStringSet containing the sequences.
k	size of used k-mers.

Details

`distSimRank()` returns 1-`simRank()`.

Value

`simRank()` returns a similarity object of class "simil" (see **proxy**). `distSimRank()` returns a dist object.

Author(s)

Michael Hahsler

References

Santis et al, Simrank: Rapid and sensitive general-purpose k-mer search tool, BMC Ecology 2011, 11:11

Examples

```
### load sequences
sequences <- readDNAStringSet(system.file("examples/DNA_example.fasta",
 package="rMSA"))
sequences

### compute similarity
simil <- simRank(sequences)

### use hierarchical clustering
```

```
hc <- hclust(distSimRank(sequences))
plot(hc)
```

`string2character` *Convenience Functions to Convert Strings to Character Vectors*

Description

These convenience function can be used to convert character strings into vectors of single characters and back.

Usage

```
c2s(x)
s2c(x)
```

Arguments

`x`	for `c2s` a single character string and for `s2c` a vector of single characters.

Value

Either a single character string or a vector of single characters.

Author(s)

Michael Hahsler

Examples

```
s <- sample(c("A", "C", "G", "T"), 10, replace = TRUE)
s

s2 <- c2s(s)
s2

s2c(s2)
```

Index