

# Package ‘CorMut’

October 17, 2017

**Type** Package

**Title** Detect the correlated mutations based on selection pressure

**Version** 1.18.0

**Date** 2014-03-18

**Author** Zhenpeng Li, Yang Huang, Yabo Ouyang, Yiming Shao, Liying Ma

**Maintainer** Zhenpeng Li<zpli21@gmail.com>

**Depends** seqinr,igraph

**Description** CorMut provides functions for computing kaks for individual sites or specific amino acids and detecting correlated mutations among them. Three methods are provided for detecting correlated mutations ,including conditional selection pressure, mutual information and Jaccard index. The computation consists of two steps: First, the positive selection sites are detected; Second, the mutation correlations are computed among the positive selection sites. Note that the first step is optional. Meanwhile, CorMut facilitates the comparison of the correlated mutations between two conditions by the means of correlated mutation network.

**License** GPL-2

**biocViews** Sequencing

**NeedsCompilation** no

## R topics documented:

CorMut-package . . . . .	2
biCkaksCodon-class . . . . .	3
biCompare-class . . . . .	3
biCompare-methods . . . . .	4
ckaks-class . . . . .	5
ckaksAA . . . . .	6
ckaksCodon . . . . .	7
filterSites-methods . . . . .	8
Jl-class . . . . .	9
jiAA . . . . .	9
kaksAA . . . . .	10
kaksAA-class . . . . .	11

kaksCodon . . . . .	12
kaksCodon-class . . . . .	13
MI-class . . . . .	14
miAA . . . . .	15
miCodon . . . . .	16
plot . . . . .	17
seqFormat-methods . . . . .	17

<b>Index</b>	<b>19</b>
--------------	-----------

---

CorMut-package	<i>Detect the correlated mutations based on selection pressure</i>
----------------	--

---

## Description

CorMut provides functions for computing kaks for individual site or specific amino and detecting correlated mutation among them. Three methods are provided for detecting correlated mutation, including conditional selection pressure, mutual information and Jaccard index. the computation consists of two steps: First, the positive selection sites are detected; Second, the mutation correlation are computed among the positive selection sites. Note that the first step is optional. Meanwhile, CorMut facilitates the comparison of the correlated mutations between two conditions by the means of correlated mutation network.

## Details

Package:	CorMut
Type:	Package
Version:	1.5.5
Date:	2014-03-18
License:	GPL-2

## Author(s)

Zhenpeng Li, Yang Huang, Yabo Ouyang, Liying Ma Maintainer: Zhenpeng Li <zpli21@gmail.com>

## References

1. Chen, L., Perlina, A. & Lee, C. J. Positive selection detection in 40,000 human immunodeficiency virus (HIV) type 1 sequences automatically identifies drug resistance and positive fitness mutations in HIV protease and reverse transcriptase. *Journal of virology* 78,3722-3732 (2004).
2. Cover, T. M., Thomas, J. A. & Wiley, J. Elements of information theory. 6, (Wiley Online Library: 1991).

---

biCkaksCodon-class      *Class "biCkaksCodon", "biCkaksAA", "biMICodon" and "biMIAA"*

---

### Description

biCkaksCodon/biCkaksAA/biMICodon/biMIAA class for representing seqFormat result of two conditions

### Slots

seq\_formatted: Object of class "DNAStrngSet". The result of seqFormat for the first condition

seq\_formatted02: Object of class "DNAStrngSet". The result of seqFormat for the second condition

### Methods

**biCompare** signature(object = "biCkaksCodon"): Compare the correlated mutations(in the unit of codon) between two conditions using condition kaks method.

**biCompare** signature(object = "biCkaksAA"): Compare the correlated mutations(in the unit of amino) between two conditions using condition kaks method.

**biCompare** signature(object = "biMICodon"): Compare the correlated mutations(in the unit of codon) between two conditions using mutual information method.

**biCompare** signature(object = "biMIAA"): Compare the correlated mutations(in the unit of amino) between two conditions using mutual information method.

### See Also

[biCompare,plot](#)

### Examples

```
#examplefile=system.file("extdata", "PI_treatment_naive.aln", package="CorMut")
#examplefile02=system.file("extdata", "PI_treatment.aln", package="CorMut")
#seq_formatted=seqFormat(examplefile)
#seq_formatted02=seqFormat(examplefile02)
#biexample=biCkaksAA(seq_formatted, seq_formatted02)
```

---

biCompare-class      *Class "biCompare"*

---

### Description

biCompare class for representing the biCompare results

**Slots**

**method:** Object of class "character". The method to compute the correlated mutations, including ckaksCodon,ckaksAA,miCodon,miAA

**state\_1:** Object of class "matrix". The conditional kaks(conditional selection pressure) or mutual information among codons or amino mutations.

**statistic\_1:** Object of class "matrix". LOD confidence score or p value among codons or amino mutations for the first condition.

**state\_2:** Object of class "matrix". The conditional kaks(conditional selection pressure) or mutual information among codons or amino mutations.

**statistic\_2:** Object of class "matrix". LOD confidence score or p value among codons or amino mutations for the second condition.

**positiveSite01:** Object of class "ANY". A vector of positive selection sites for the first condition.

**positiveSite02:** Object of class "ANY". A vector of positive selection sites for the second condition.

**Methods**

**plot** signature(object = "biCompare"): plot the result of biCompare class

**See Also**

[biCompare,plot-methods](#)

**Examples**

```
#examplefile=system.file("extdata","PI_treatment_naive.aln",package="CorMut")
#examplefile02=system.file("extdata","PI_treatment.aln",package="CorMut")
#seq_formated=seqFormat(examplefile)
#seq_formated02=seqFormat(examplefile02)
#biexample=biCkaksAA(seq_formated,seq_formated02)
#result=biCompare(biexample)
```

---

biCompare-methods

*biCompare implement biCompare methods for comparison of the correlated mutations between two conditions by the means of correlated mutation network*

---

**Description**

Comparison of the correlated mutations between two conditions by the means of correlated mutation network, two correlated mutation network will be displayed in a plot, which represent the global mutation correlation among sites or mutations in two conditions. In the plot, blue nodes indicate the positive selection nodes in the first condition, while red nodes indicate the positive selection nodes in the second condition.

**Value**

A biCompare instance is returned.

**Methods**

**x = "biCkaksCodon"** Compare the correlated mutations(in the unit of codon) between two conditions using condition kaks method.

**x = "biCkaksAA"** Compare the correlated mutations(in the unit of amino) between two conditions using condition kaks method.

**x = "biMICodon"** Compare the correlated mutations(in the unit of codon) between two conditions using mutual information method.

**x = "biMIAA"** Compare the correlated mutations(in the unit of amino) between two conditions using mutual information method.

**x = "biJIAA"** Compare the correlated mutations(in the unit of amino) between two conditions using mutual information method.

**Author(s)**

Zhenpeng Li

**See Also**

[biCompare-class](#), [plot.biCompare](#)

**Examples**

```
examplefile=system.file("extdata", "PI_treatment_naive.aln", package="CorMut")
examplefile02=system.file("extdata", "PI_treatment.aln", package="CorMut")
seq_formated=seqFormat(examplefile)
seq_formated02=seqFormat(examplefile02)
biexample=biCkaksAA(seq_formated, seq_formated02)
result=biCompare(biexample)
```

---

ckaks-class

*Class "ckaks"*

---

**Description**

ckaks class for representing the ckaksCodon/ckaksAA results

**Slots**

**ckaks:** Object of class "matrix". Conditional kaks among codons or amino mutations

**lod:** Object of class "matrix". LOD confidence score indicates the significance of correlated mutations

**Methods**

**filterSites** signature(object = "ckaks"): Filter positive selection amino mutations for objects of ckaks class.

**plot** signature(object = "ckaks"): Plot the influence network among mutations for objects of ckaks class.

**See Also**

[filterSites](#), [ckaksCodon](#), [ckaksAA](#), [plot.ckaks](#)

**Examples**

```
#examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
#example=seqFormat(examplefile)
#result=ckaksAA(example)
```

---

ckaksAA

*Compute the conditional kaks(conditional selection pressure) among amino mutations*

---

**Description**

Compute the conditional kaks(conditional selection pressure) among codons, the amino mutation in a site will be treated as whole in computation.

**Usage**

```
ckaksAA(seq_formatted, kaks = T, lod_cut = 2, setPosition = c())
```

**Arguments**

seq_formatted	Formatted alignment sequence. i.e. the result after the treatment of DataFormat-CorMut.
kaks	A logical variable to indicate whether kaks is turn on or off, if kaks is TRUE, conditional kaks will be computed only among positive selection sites, or if kaks is FALSE, conditional kaks will be computed only among all sites of sequence.
lod_cut	The LOD confidence score cutoff, the default value is 2. If lod is larger than 2, it means the positive selection of individual site or the conditional selection pressure among sites are significant.
setPosition	The positions of sequence to compute. setPosition should be a vector of interger type.

**Value**

An ckaks instance will be returned. ckaks includes two slots of matrix type:ckaks and lod, which indicate the conditional kaks and LOD confidence score respectively.

**Author(s)**

Zhenpeng Li

**References**

Chen, L., Perlina, A. & Lee, C. J. Positive selection detection in 40,000 human immunodeficiency virus (HIV) type 1 sequences automatically identifies drug resistance and positive fitness mutations in HIV protease and reverse transcriptase. *Journal of virology* 78,3722-3732 (2004).

**See Also**

[filterSites](#), [plot.ckaks](#), [ckaksCodon](#)

**Examples**

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=ckaksAA(example)
```

---

ckaksCodon	<i>Compute the conditional kaks(conditional selection pressure) among codons</i>
------------	--

---

**Description**

Compute the conditional kaks(conditional selection pressure) among codons, the amino mutation in a site will be treated as whole in computation.

**Usage**

```
ckaksCodon(seq_formatted, kaks = TRUE, lod_cut = 2, setPosition = c())
```

**Arguments**

seq_formatted	Formatted alignment sequence. i.e. the result after the treatment of DataFormat-CorMut.
kaks	A logical variable to indicate whether kaks is turn on or off, if kaks is TRUE, conditional kaks will be computed only among positive selection sites, or if kaks is FALSE, conditional kaks will be computed only among all sites of sequences.
lod_cut	The LOD confidence score cutoff, the default value is 2. If lod is larger than 2, it means the positive selection of individual site or the conditional selection pressure among sites are significant.
setPosition	The positions of sequence to compute. setPosition should be a vector of interger type indicating the positions to compute.

**Value**

A object of ckaks class will be return. ckaks includes two slots of matrix:ckaks and lod, which indicate the ckaks and lod confidence score respectively.

**Author(s)**

Zhenpeng Li

**References**

Chen, L., Perlina, A. & Lee, C. J. Positive selection detection in 40,000 human immunodeficiency virus (HIV) type 1 sequences automatically identifies drug resistance and positive fitness mutations in HIV protease and reverse transcriptase. *Journal of virology* 78,3722-3732 (2004).

**See Also**

[filterSites,plot.ckaks,ckaksAA](#)

**Examples**

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=ckaksCodon(example)
```

---

filterSites-methods     *filterSites methods*

---

**Description**

filterSites implement filterSites methods for kaksCodon,kaksAA,ckaks and MI instances respectively.The functions filter the results of corresponding objects.

**Methods**

**x = "kaksCodon"** filterSites(x, lod\_cut = 2, freq\_cut=0.01): Filter the results of kaksCodon object,two parameters are provided to control the output, lod\_cut can assign an cutoff for lod confidence score to filter, and freq\_cut assign the cutoff frequency of mutation, thus low-frequency mutations are ignored in the later analyses.

**x = "kaksAA"** filterSites(x, lod\_cut = 2, freq\_cut=0.01): Filter the results of kaksAA object,two parameters are provided to control the output, lod\_cut can assign an cutoff for lod confidence score to filter, and freq\_cut assign the cutoff frequency of mutation, thus low-frequency mutations are ignored in the later analyses.

**x = "ckaks"** filterSites(x, lod\_cut=2): Filter the results of ckaks object,lod\_cut can assign an cutoff for lod confidence score to filter.

**x = "MI"** filterSites(x, p\_cut=0.05): Filter the results of MI object,p\_cut can assign an p value cutoff to filter the correlated mutations.

**x = "JI"** filterSites(x, p\_cut=0.05): Filter the results of MI object,p\_cut can assign an p value cutoff to filter the correlated mutations.

**Author(s)**

Zhenpeng Li

**See Also**

[kaksCodon, kaksAA, ckaksCodon, ckaksAA, miCodon, miAA, plot-methods](#)

**Examples**

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=kaksCodon(example)
resultfilter=filterSites(result)
```



---

 JI-class

 Class "JI"
 

---

### Description

JI class for representing the jiAA results

### Slots

JI: Object of class "matrix". Mutual information among codons or amino mutations

p.value: Object of class "matrix". P value for the significance of correlated mutations

OR: Object of class "matrix". Odds Ratios for the correlated mutations

### Methods

**filterSites** signature(object = "MI"): Filter positive selection amino mutations for objects of MI class.

**plot** signature(object = "MI"): Plot the influence interaction among mutations for objects of MI class.

### See Also

[miAA](#), [filterSites](#)

### Examples

```
#examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
#example=seqFormat(examplefile)
#result=jiAA(example)
```

---

 jiAA

 Compute the Jaccard index among individual amino mutations
 

---

### Description

Compute the Jaccard index among individual amino mutations, the amino mutations in a specific position will be considered respectively.

### Usage

```
jiAA(seq_formated, kaks = TRUE, lod_cut = 2, setPosition = c(),fdr=FALSE)
```

**Arguments**

seq_formatted	Formatted multiple alignment sequence. i.e. the result after the treatment of DataFormatCorMut.
kaks	A logical variable to indicate whether kaks is turn on or off, if kaks is TRUE, conditional kaks will be computed only among positive selection sites, or if kaks is FALSE, conditional kaks will be computed only among all sites of sequence.
lod_cut	The LOD confidence score cutoff, the default value is 2. If lod is larger than 2, it means the positive selection of individual site or the conditional selection pressure among sites are significant.
setPosition	The positions of sequence to compute. setPosition should be a vector of interger type.
fdr	Decide whether use FDR procedure to control the p value of the computation. The default value is False.

**Value**

A object of JI class will be returned. JI class includes three slots: JI, p.value and OR, which indicate the Jaccard index, p value and odds ratios respectively.

**Author(s)**

Zhenpeng Li

**See Also**

[filterSites,miAA](#)

**Examples**

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=jiAA(example)
```

---

kaksAA

*Compute kaks for individual amino mutation*

---

**Description**

Compute kaks for individual amino mutation.

**Usage**

```
kaksAA(seq_formatted)
```

**Arguments**

seq_formatted	Formatted alignment sequence. i.e. the result after the treatment of DataFormatCorMut.
---------------	--

## Details

Ka/Ks ratio was used as an indicator of selective pressure acting on a protein-coding gene. A Ka/Ks ratio of 1 indicates neutral selection, i.e., the observed ratio of non-synonymous mutations versus synonymous mutations exactly matches the ratio expected under a random mutation model. Thus, amino acid changes are neither being selected for nor against. A Ka/Ks value of <1 indicates negative selection pressure. That is to say most amino acid changes are deleterious and are selected against, producing an imbalance in the observed mutations that favors synonymous mutations. In the condition of  $Ka/Ks > 1$ , it indicates that amino acid changes are favored, i.e., they increase the organism's fitness. This unusual condition may reflect a change in the function of a gene or a change in environmental conditions that forces the organism to adapt. For example, highly variable viruses mutations which confer resistance to new antiviral drugs might be expected to undergo positive selection in a patient population treated with these drugs, such as HIV and HCV.

## Value

A kaksAA instance will be returned.

## Note

The reference sequence should be included in in the alignment as the first sequence. Before using kaksCodon, the alignment should be treated with DataFormatCorMut.

## Author(s)

Zhenpeng Li

## References

Chen, L., Perlina, A. & Lee, C. J. Positive selection detection in 40,000 human immunodeficiency virus (HIV) type 1 sequences automatically identifies drug resistance and positive fitness mutations in HIV protease and reverse transcriptase. *Journal of virology* 78, 3722-3732 (2004).

## See Also

[filterSites](#), [kaksCodon](#)

## Examples

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=kaksAA(example)
```

---

kaksAA-class

*Class "kaksAA"*

---

## Description

kaksAA class for representing the kaksAA results

**Slots**

seq\_num: Object of class "list". The number of sequences

kaks: Object of class "list". kaks value

lod: Object of class "list". LOD confidence score, refer to vignette for details

q: Object of class "list". q value, refer to vignette for details

ka: Object of class "list".ka value of kaks ratio, i.e. the samples count of the nonsynonymous mutation.

ks: Object of class "list". ks value of kaks ratio, i.e. the samples count of the synonymous mutation.

**Methods**

**filterSites** signature(object = "kaksAA"): Filter positive selection amino mutations for objects of kaksAA class.

**See Also**

[filterSites,kaksAA](#)

**Examples**

```
#examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
#example=seqFormat(examplefile)
#result=kaksAA(example)
```

---

kaksCodon

*Compute kaks for individual codon*

---

**Description**

Compute kaks for individual codon, the different mutations in a site will be considered as whole.

**Usage**

```
kaksCodon(seq_formated)
```

**Arguments**

seq\_formated      Formated alignment sequence. i.e. the result after the treatment of DataFormat-CorMut.

**Details**

Ka/Ks ratio was used as an indicator of selective pressure acting on a protein-coding gene. A Ka/Ks ratio of 1 indicates neutral selection, i.e., the observed ratio of non-synonymous mutations versus synonymous mutations exactly matches the ratio expected under a random mutation model. Thus, amino acid changes are neither being selected for nor against. A Ka/Ks value of <1 indicates negative selection pressure. That is to say most amino acid changes are deleterious and are selected against, producing an imbalance in the observed mutations that favors synonymous mutations. In the condition of  $Ka/Ks > 1$ , it indicates that amino acid changes are favored, i.e., they increase the

organism's fitness. This unusual condition may reflect a change in the function of a gene or a change in environmental conditions that forces the organism to adapt. For example, highly variable viruses mutations which confer resistance to new antiviral drugs might be expected to undergo positive selection in a patient population treated with these drugs, such as HIV and HCV.

### Value

A kaksCodon instance will be returned.

### Note

The reference sequence should be included in the alignment as the first sequence. Before using kaksCodon, the alignment should be treated with DataFormatCorMut.

### Author(s)

Zhenpeng Li

### References

Chen, L., Perlina, A. & Lee, C. J. Positive selection detection in 40,000 human immunodeficiency virus (HIV) type 1 sequences automatically identifies drug resistance and positive fitness mutations in HIV protease and reverse transcriptase. *Journal of virology* 78, 3722-3732 (2004).

### See Also

[filterSites-methods,kaksAA](#)

### Examples

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
example=seqFormat(examplefile)
result=kaksCodon(example)
```

---

kaksCodon-class	<i>Class "kaksCodon"</i>
-----------------	--------------------------

---

### Description

kaksCodon class for representing the kaksCodon results

### Slots

seq\_num: Object of class "list". The number of sequences

kaks: Object of class "numeric". kaks value

lod: Object of class "numeric". LOD confidence score, refer to vignette for details

q: Object of class "numeric". q value, refer to vignette for details

ka: Object of class "numeric". ka value of kaks ratio, i.e. the samples count of the nonsynonymous mutation.

ks: Object of class "numeric". ks value of kaks ratio, i.e. the samples count of the synonymous mutation.

**Methods**

**filterSites** signature(object = "kaksCodon"): Filter positive selection sites for objects of kaksCodon class.

**See Also**

[kaksCodon](#), [filterSites](#)

**Examples**

```
#examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
#example=seqFormat(examplefile)
#result=kaksCodon(example)
```

---

MI-class

*Class "MI"*

---

**Description**

MI class for representing the miCodon/miAA results

**Slots**

**mi**: Object of class "matrix". Mutual information among codons or amino mutations

**p.value**: Object of class "matrix". P value for the significance of correlated mutations

**Methods**

**filterSites** signature(object = "MI"): Filter positive selection amino mutations for objects of MI class.

**plot** signature(object = "MI"): Plot the influence interaction among mutations for objects of MI class.

**See Also**

[miCodon](#), [filterSites](#)

**Examples**

```
#examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")
#example=seqFormat(examplefile)
#result=miAA(example)
```

miAA

*Compute the mutual information among individual amino mutations***Description**

Compute the mutual information among individual amino mutations, the amino mutations in a specific position will be considered respectively.

**Usage**

```
miAA(seq_formated, kaks = TRUE, lod_cut = 2, setPosition = c(),fdr=FALSE)
```

**Arguments**

seq_formated	Formatted multiple alignment sequence. i.e. the result after the treatment of DataFormatCorMut.
kaks	A logical variable to indicate whether kaks is turn on or off, if kaks is TRUE, conditional kaks will be computed only among positive selection sites, or if kaks is FALSE, conditional kaks will be computed only among all sites of sequence.
lod_cut	The LOD confidence score cutoff, the default value is 2. If lod is larger than 2, it means the positive selection of individual site or the conditional selection pressure among sites are significant.
setPosition	The positions of sequence to compute. setPosition should be a vector of interger type.
fdr	Decide whether use FDR procedure to control the p value of the computation. The default value is False.

**Value**

A object of MI class will be returned. miAA includes two slots of matrix:mi and p.value, which indicate the mutual information and p value respectively.

**Author(s)**

Zhenpeng Li

**References**

Cover, T. M., Thomas, J. A. & Wiley, J. Elements of information theory. 6, (Wiley Online Library: 1991).

**See Also**

[filterSites](#),[miCodon](#)

**Examples**

```
examplefile=system.file("extdata","PI_treatment.aln",package="CorMut")
example=seqFormat(examplefile)
result=miAA(example)
```

miCodon

*Compute the mutual information among codons***Description**

Compute the mutual information among codons, the amino mutation in a site will be treated as whole in computation.

**Usage**

```
miCodon(seq_formated, kaks = TRUE, lod_cut = 2, setPosition = c(),fdr=FALSE)
```

**Arguments**

seq_formated	Formatted alignment sequence. i.e. the result after the treatment of DataFormat-CorMut.
kaks	A logical variable to indicate whether kaks is turn on or off, if kaks is TRUE, conditional kaks will be computed only among positive seelction sites, or if kaks is FALSE, conditional kaks will be computed only among all sites of sequence.
lod_cut	The LOD confidence score cutoff, the default value is 2. If lod is larger than 2, it means the positive selection of individual site or the conditional selection pressure among sites are significant.
setPosition	The positions of sequence to compute. setPosition should be a vector of interger type.
fdr	Decide whether use FDR procedure to control the p value of the computation.The default value is False.

**Value**

A object of MI class will be return. miCodon includes two slots of matrix:mi and p.value, which indicate the mutual information and p value respectively.

**Author(s)**

Zhenpeng Li

**References**

Cover, T. M., Thomas, J. A. & Wiley, J. Elements of information theory. 6, (Wiley Online Library: 1991).

**See Also**

[filterSites](#),[miAA](#)

**Examples**

```
examplefile=system.file("extdata","PI_treatment.aln",package="CorMut")
example=seqFormat(examplefile)
result=miCodon(example)
```



---

plot

*Plot methods for CorMut package*

---

### Description

Plot implement plot methods for ckaksCodon,ckaksAA,miCodon, miAA and biCompare objects respectively. Plot visualize the mutation correlation among sites or amino mutations.

### Methods

**x = "ckaks"** Plot the results of ckaks object

**x = "MI"** Plot the results of MI object

**x = "JI"** Plot the results of JI object

**x = "biCompare"** Plot the results of biCompare object

### Author(s)

Zhenpeng Li

### See Also

[ckaksCodon](#),[ckaksAA](#),[miCodon](#),[miAA](#),[jiAA](#)

### Examples

```
examplefile=system.file("extdata","PI_treatment.aln",package="CorMut")
example=seqFormat(examplefile)
result=ckaksAA(example)
plot(result)
```

---

seqFormat-methods

*Process the multiple sequence alignment files*

---

### Description

Process the multiple sequence alignment files, the results can be used for the other functional functions. Note that the reference sequence should be included as the first sequence.

### Usage

```
seqFormat(x, format = c("clustal","fasta","mase","phylip","msf"))
```

### Arguments

x multiple sequence alignment files

format a character string specifying the format of the file: "clustal","fasta","mase","phylip","msf", refer to read.alignment function in seqinr package for details.

**Value**

A vector of sequences with names is returned.

**Note**

To ensure an exact result, the reference sequence should be included in the alignment as the first sequence.

**Author(s)**

Zhenpeng Li

**Examples**

```
examplefile=system.file("extdata", "PI_treatment.aln", package="CorMut")  
example=seqFormat(examplefile)
```

# Index

## \*Topic **classes**

- biCkaksCodon-class, 3
- biCompare-class, 3
- ckaks-class, 5
- JI-class, 9
- kaksAA-class, 11
- kaksCodon-class, 13
- MI-class, 14

## \*Topic **methods**

- filterSites-methods, 8
- plot, 17

- biCkaksAA (biCkaksCodon-class), 3
- biCkaksAA-class (biCkaksCodon-class), 3
- biCkaksCodon (biCkaksCodon-class), 3
- biCkaksCodon-class, 3
- biCompare, 3, 4
- biCompare (biCompare-methods), 4
- biCompare, biCkaksAA-method (biCompare-methods), 4
- biCompare, biCkaksCodon-method (biCompare-methods), 4
- biCompare, biJIAA-method (biCompare-methods), 4
- biCompare, biMIAA-method (biCompare-methods), 4
- biCompare, biMICodon-method (biCompare-methods), 4
- biCompare-class, 3
- biCompare-methods, 4
- biCompare.biCkaksAA (biCompare-methods), 4
- biCompare.biCkaksCodon (biCompare-methods), 4
- biCompare.biJIAA (biCompare-methods), 4
- biCompare.biMIAA (biCompare-methods), 4
- biCompare.biMICodon (biCompare-methods), 4
- biJIAA (biCkaksCodon-class), 3
- biJIAA-class (biCkaksCodon-class), 3
- biMIAA (biCkaksCodon-class), 3
- biMIAA-class (biCkaksCodon-class), 3
- biMICodon (biCkaksCodon-class), 3
- biMICodon-class (biCkaksCodon-class), 3

- ckaks (ckaks-class), 5

- ckaks-class, 5

- ckaks.CorMut (ckaks-class), 5

- ckaks.CorMut, ckaks-method (ckaks-class), 5

- ckaksAA, 6, 6, 8, 17

- ckaksCodon, 6, 7, 7, 8, 17

- CorMut (CorMut-package), 2

- CorMut-package, 2

- filterSites, 6–12, 14–16

- filterSites (filterSites-methods), 8

- filterSites, ckaks-method (filterSites-methods), 8

- filterSites, JI-method (filterSites-methods), 8

- filterSites, kaksAA-method (filterSites-methods), 8

- filterSites, kaksCodon-method (filterSites-methods), 8

- filterSites, MI-method (filterSites-methods), 8

- filterSites-methods, 8

- filterSites.ckaks (filterSites-methods), 8

- filterSites.JI (filterSites-methods), 8

- filterSites.kaksAA (filterSites-methods), 8

- filterSites.kaksCodon (filterSites-methods), 8

- filterSites.MI (filterSites-methods), 8

- JI-class, 9

- ji.CorMut, JI-method (JI-class), 9

- jiAA, 9, 17

- ka.CorMut (kaksAA-class), 11

- ka.CorMut, kaksAA-method (kaksAA-class), 11

- ka.CorMut, kaksCodon-method (kaksCodon-class), 13

- kaks.CorMut (kaksAA-class), 11

- kaks.CorMut, kaksAA-method (kaksAA-class), 11

- kaks.CorMut,kaksCodon-method  
(kaksCodon-class), 13
- kaksAA, 8, 10, 12, 13
- kaksAA-class, 11
- kaksCodon, 8, 11, 12, 14
- kaksCodon-class, 13
- ks.CorMut (kaksAA-class), 11
- ks.CorMut,kaksAA-method (kaksAA-class),  
11
- ks.CorMut,kaksCodon-method  
(kaksCodon-class), 13
  
- lod.CorMut (ckaks-class), 5
- lod.CorMut,ckaks-method (ckaks-class), 5
- lod.CorMut,kaksAA-method  
(kaksAA-class), 11
- lod.CorMut,kaksCodon-method  
(kaksCodon-class), 13
  
- method.CorMut (biCompare-class), 3
- method.CorMut,biCompare-method  
(biCompare-class), 3
- MI-class, 14
- mi.CorMut (MI-class), 14
- mi.CorMut,MI-method (MI-class), 14
- miAA, 8–10, 15, 16, 17
- miCodon, 8, 14, 15, 16, 17
  
- p.value.CorMut (MI-class), 14
- p.value.CorMut,JI-method (JI-class), 9
- p.value.CorMut,MI-method (MI-class), 14
- plot, 3, 17
- plot,biCompare-method (plot), 17
- plot,ckaks-method (plot), 17
- plot,JI-method (plot), 17
- plot,MI-method (plot), 17
- plot.biCompare, 5
- plot.biCompare (plot), 17
- plot.ckaks, 6–8
- plot.ckaks (plot), 17
- plot.JI (plot), 17
- plot.MI (plot), 17
- positiveSite01.CorMut  
(biCompare-class), 3
- positiveSite01.CorMut,biCompare-method  
(biCompare-class), 3
- positiveSite02.CorMut  
(biCompare-class), 3
- positiveSite02.CorMut,biCompare-method  
(biCompare-class), 3
  
- q.CorMut (kaksAA-class), 11
- q.CorMut,kaksAA-method (kaksAA-class),  
11
  
- q.CorMut,kaksCodon-method  
(kaksCodon-class), 13
  
- seq\_num.CorMut (kaksAA-class), 11
- seq\_num.CorMut,kaksAA-method  
(kaksAA-class), 11
- seq\_num.CorMut,kaksCodon-method  
(kaksCodon-class), 13
- seqFormat (seqFormat-methods), 17
- seqFormat-methods, 17
- show,biCkaksAA-method  
(biCkaksCodon-class), 3
- show,biCkaksCodon-method  
(biCkaksCodon-class), 3
- show,biMIAA-method  
(biCkaksCodon-class), 3
- show,biMICodon-method  
(biCkaksCodon-class), 3
- show,ckaks-method (ckaks-class), 5
- show,JI-method (JI-class), 9
- show,kaksAA-method (kaksAA-class), 11
- show,kaksCodon-method  
(kaksCodon-class), 13
- show,MI-method (MI-class), 14
- state\_1.CorMut (biCompare-class), 3
- state\_1.CorMut,biCompare-method  
(biCompare-class), 3
- state\_2.CorMut (biCompare-class), 3
- state\_2.CorMut,biCompare-method  
(biCompare-class), 3
- statistic\_1.CorMut (biCompare-class), 3
- statistic\_1.CorMut,biCompare-method  
(biCompare-class), 3
- statistic\_2.CorMut (biCompare-class), 3
- statistic\_2.CorMut,biCompare-method  
(biCompare-class), 3