Package 'ftrCOOL'

August 31, 2020

Type Package

Title Feature Extraction from Biological Sequences

Version 0.1.0

Author Sare Amerifar

Maintainer Sare Amerifar <sare.ameri.01@gmail.com>

Description Extracts features from biological sequences. It contains most features which are presented in related work and also includes features which have never been introduced before. It extracts numerous features from Nucleotide and Peptide sequences. Each feature converts the input sequences to discrete numbers in order to use them as predictors in machine learning models. There are many features and information which are hidden inside a sequence. Utilizing the package, users can convert biological sequences to discrete models based on chosen properties. References: 'iLearn' 'Z. Chen et al.' (2019) <DOI:10.1093/bib/bbz041>. 'iFeature' 'Z. Chen et al.' (2018) <DOI:10.1093/bioinformatics/bty140>. <https://CRAN.R-project.org/package=rDNAse>. 'PseKRAAC' 'Y. Zuo et al.' 'PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition' (2017) <DOI:10.1093/bioinformatics/btw564>. 'iDNA6mA-PseKNC' 'P. Feng et al.' 'iDNA6mA-PseKNC: Identifying DNA N6-methyladenosine sites by incorporating nucleotide physicochemical properties into PseKNC' (2019) <DOI:10.1016/j.ygeno.2018.01.005>. 'I. Dubchak et al.' 'Prediction of protein folding class using global description of amino acid sequence' (1995) <DOI:10.1073/pnas.92.19.8700>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports stats, utils

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2020-08-31 09:10:02 UTC

R topics documented:

`AA2Binary` *Amino Acid To Binary*

Description

This function transforms an amino acid to a binary format. The type of the binary format is determined by the binaryType parameter. For details about each format, please refer to the description of the binaryType parameter.

Usage

```
AA2Binary(
  seqs,
  binaryType = "strBin",
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
binaryType	It can take any of the following values: ('strBin','logicBin','numBin'). 'strBin'(String binary): each amino acid is represented by a string containing 20 characters(0-1). For example, A = ALANIN = "1000000...0" 'logicBin'(logical value): Each amino acid is represented by a vector containing 20 logical entries. For example, A = ALANIN = c(T,F,F,F,F,F,F,...F) 'numBin' (numeric bin): Each amino acid is represented by a numeric (i.e., integer) vector containing 20 numerals. For example, A = ALANIN = c(1,0,0,0,0,0,0,...,0)
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Value

The output is different depending on the outFormat parameter ('mat' or 'txt'). If outFormat is set to 'mat', it returns a feature matrix for sequences with the same lengths. The number of rows is equal to the number of sequences and if binaryType is 'strBin', the number of columns is the length of the sequences. Otherwise, it is equal to (length of the sequences)*20. 'mat' format is used for machine learning purposes. If outFormat is 'txt', all binary values will be written to a 'txt' file. Each line in the file shows the binary format of a sequence.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
```

```
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat<-AA2Binary(seqs = ptmSeqsVect, binaryType="strBin",outFormat="mat")
```

AAindex *Amino Acid Index*

Description

This function converts the amino acids of a sequence to a list of physicochemical properties in the aaIndex file. For each amino acid, the function uses a numeric vector which shows the aaIndex of the amino acid.

Usage

```
AAindex(
  seqs,
  selectedAAidx = 1:554,
  standardized = TRUE,
  threshold = 1,
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
selectedAAidx	AAindex function works based on physicochemical properties. Users select the properties by their ids or indexes in aaIndex2 file.
standardized	is a logical parameter. If it is set to TRUE, amino acid indices will be in the standard format. The default value is TRUE.
threshold	is a number between (0 , 1]. In selectedAAidx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Details

In this function each amino acid is converted to a numeric vector. Elements of the vector represent a physicochemical property for the amino acid. In the aaIndex database, there are 554 amino acid indices. Users can choose the desired aaindex by specifying aaindexes through their ids or indexes in the aaIndex file, via selectedAAidx parameter.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length)*(number of selected amino acid indexes) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
dir = tempdir()
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat<-AAindex(seqs = ptmSeqsVect, selectedAAidx=1:5,outFormat="mat")

ad<-paste0(dir,"/aaidx.txt")
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
AAindex(seqs = filePrs, selectedAAidx=1:5,standardized=TRUE,threshold=1,outFormat="txt"
,outputFileDist=ad)
```

AAKpartComposition *Amino Acid to K Part Composition*

Description

In this function, each sequence is divided into k equal partitions. The length of each part is equal to ceiling(l(lenght of the sequence)/k). The last part can have a different length containing the residual amino acids. The amino acid composition is calculated for each part.

Usage

```
AAKpartComposition(seqs, k = 3, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.

k	is an integer value. Each sequence should be divided to k partition(s).
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

a feature matrix with k*20 number of columns. The number of rows is equal to the number of sequences.

Note

Warning: The length of all sequences should be greater than k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-AAKpartComposition(seqs=filePrs,k=5,normalized=FALSE)
```

AAutoCor *Amino Acid Autocorrelation-Autocovariance*

Description

It creates the feature matrix for each function in autocorelation (i.e., Moran, Greay, NormalizeM-Borto) or autocovariance (i.e., AC, CC,ACC). The user can select any combination of the functions too. In this case, the final matrix will contain features of each selected function.

Usage

```
AAutoCor(
  seqs,
  selectedAAidx = list(c("CIDH920105", "BHAR880101", "CHAM820101", "CHAM820102",
    "CHOC760101", "BIGC670101", "CHAM810101", "DAYM780201")),
  maxlag = 3,
  threshold = 1,
  type = c("Moran", "Geary", "NormalizeMBorto", "AC", "CC", "ACC"),
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
selectedAAidx	Function takes as input the physicochemical properties. Users select the properties by their ids or indices in the aaIndex2 file. This parameter could be a vector or a list of amino acid indices.
maxlag	This parameter shows the maximum gap between two amino acids. The gaps change from 1 to maxlag (the maximum lag).
threshold	is a number between (0 , 1]. In selectedAAidx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
type	could be 'Moran', 'Greay', 'NormalizeMBorto', 'AC', 'CC', or 'ACC'. Also, it could be any combination of them.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

For CC and AAC autocovriance functions, which consider the covariance of the two physicochemical properties, we have provided users with the ability to categorize their selected properties in a list. The binary combination of each group will be taken into account. Note: If all the features are in a group or selectedAAidx parameter is a vector, the binary combination will be calculated for all the physicochemical properties.

Value

This function returns a feature matrix. The number of columns in the matrix changes depending on the chosen autocorrelation or autocovariance types and nlag parameter. The output is a matrix. The number of rows shows the number of sequences.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-AAutoCor(seqs=filePrs,maxlag=20,threshold=0.9,
type=c("Moran","Geary","NormalizeMBorto","AC"))

mat2<-AAutoCor(seqs=filePrs,maxlag=20,threshold=0.9,selectedAAidx=
list(c('CIDH920105','BHAR880101','CHAM820101','CHAM820102'),c('CHOC760101','BIGC670101')
,c('CHAM810101','DAYM780201')),type=c("AC","CC","ACC"))
```

alphabetCheck *AlphabetCheck*

Description

This function checks the alphabets in a sequence. If one of the following conditions hold, the sequence will be deleted: 1. A peptide sequence containing non-standard amino acids, 2. A DNA sequence with an alphabet other than A, C, G, or T, 3. An RNA sequence having an alphabet other than A, C, G, or U.

Usage

```
alphabetCheck(sequences, alphabet = "aa", label = c())
```

Arguments

sequences	is a string vector. Each element is a peptide, protein, DNA, or RNA sequences.
alphabet	This parameter shows the alphabet of sequences. If it is set to 'aa', it indicates the alphabet of amino acids. When it is 'dna', it shows the nucleotide alphabet and in case it equals 'rna', it represents ribonucleotide alphabet.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

'alphabetCheck' returns a list with two elements. The first element is a vector which contains valid sequences. The second element is a vector which contains the labels of the sequences (if any exists).

Note

This function receives a sequence vector and the label of sequences (if any). It deletes sequences (and their labels) containing non-standard alphabets.

Examples

```
seq<-alphabetCheck(sequences=c("AGDFLIAACNMLKIVYT","ADXVGAJK"),alphabet="aa")
```

APAAC *Amphiphilic Pseudo-Amino Acid Composition(APAAC) (series)*

Description

This function calculates the amphiphilic pseudo amino acid composition (Series) for each sequence.

Usage

```
APAAC(
  seqs,
  aaIDX = c("ARGP820101", "HOPT810101"),
  lambda = 3,
  w = 0.5,
  l = 1,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
aaIDX	is a vector of Ids or indexes of the user-selected physicochemical properties in the aaIndex2 database. The default values of the vector are the hydrophobicity ids and hydrophilicity ids in the amino acid index file.
lambda	is a tuning parameter. Its value indicates the maximum number of spaces between di-nucleotide pairs. The number changes from 1 to lambda.
w	(weight) is a tuning parameter. It changes in from 0 to 1. The default value is 0.5.
l	This parameter keeps the value of l in lmer composition. The lmers form the first 20^l elements of the APAAC descriptor.
threshold	is a number between (0 , 1]. In aaIDX, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

This function computes the pseudo amino acid composition for each physicochemical property. We have provided users with the ability to choose among different properties (i.e., not confined to hydrophobicity or hydrophilicity).

Value

A feature matrix such that the number of columns is 20^l+(number of chosen aaIndex*lambda) and the number of rows equals the number of sequences.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-APAAC(seqs=filePrs,l=2,lambda=3,threshold=1)
```

APkNUCdi *Amphiphilic Pseudo-k Nucleotide Composition-di(series)*

Description

This function calculates the amphiphilic pseudo k nucleotide composition(Di) (Series) for each sequence.

Usage

```
APkNUCdi(
  seqs,
  selectedNucIdx,
  lambda = 3,
  w = 0.05,
  l = 2,
  ORF = FALSE,
  reverseORF = TRUE,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	is a vector of Ids or indices of the desired physicochemical properties of dinucleotides. Users can choose the desired indices by their ids or their names in the di-nucleotide index file.
lambda	is a tuning parameter. This integer value shows the maximum limit of spaces between dinucleotide pairs. The Number of spaces changes from 1 to lambda.
w	(weight) is a tuning parameter. It changes in the range of 0 to 1. The default value is 0.5.

l	This parameter keeps the value of l in lmer composition. The lmers form the first 4^l elements of the APkNCdi descriptor.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
threshold	is a number between (0 to 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

This function computes the pseudo nucleotide composition for each physicochemical property of dinucleotides. We have provided users with the ability to choose among the 125 properties in the di-nucleotide index database.

Value

It is a feature matrix. The number of columns is 4^l+(number of the chosen indices*lambda) and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-APkNUCdi(seqs=fileLNC,selectedNucIdx=1:125,ORF=TRUE,threshold=0.8)
```

APkNUCTri *Amphiphilic Pseudo-k Nucleotide Composition-Tri(series)*

Description

This function calculates the amphiphilic pseudo k nucleotide composition(Tri) (Series) for each sequence.

Usage

```
APkNUCTri(
  seqs,
  selectedNucIdx,
  lambda = 3,
  w = 0.05,
  l = 3,
  ORF = FALSE,
```

```
  reverseORF = TRUE,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	is a vector of Ids or indices of the desired physicochemical properties of trinucleotides. Users can choose the desired indices by their ids or their names in the tri-nucleotide index file.
lambda	is a tuning parameter. This integer value shows the maximum limit of spaces between trinucleotide pairs. The Number of spaces changes from 1 to lambda.
w	(weight) is a tuning parameter. It changes in the range of 0 to 1. The default value is 0.5.
l	This parameter keeps the value of l in lmer composition. The lmers form the first 4^l of the APkNCTri descriptor.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
threshold	is a number between (0 , 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

This function computes the pseudo nucleotide composition for each physicochemical property of trinucleotides. We have provided users with the ability to choose among the 12 properties in the tri-nucleotide index database.

Value

It is a feature matrix. The number of columns is 4^l+(number of the chosen indices*lambda) and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-APkNUCTri(seqs=fileLNC,selectedNucIdx=1:12,l=2,ORF=TRUE,threshold=0.8)
```

AutoCorDiNuc *Di Nucleotide Autocorrelation-Autocovariance*

Description

It creates the feature matrix for each function in autocorelation (i.e., Moran, Greay, NormalizeMBorto) or autocovariance (i.e., AC, CC,ACC). The user can select any combination of the functions too. In this case, the final matrix will contain features of each selected function.

Usage

```
AutoCorDiNuc(
  seqs,
  selectedNucIdx,
  maxlag = 3,
  threshold = 1,
  type = c("Moran", "Geary", "NormalizeMBorto", "AC", "CC", "ACC"),
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	function takes as input the physicochemical properties. Users select the properties by their ids or indices in the nuIdx2 file. This parameter could be a vector or a list of dinucleotide indices.
maxlag	This parameter shows the maximum gap between two amino acids. The gaps change from 1 to maxlag (the maximum lag).
threshold	is a number between (0 to 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted.The default value is 1.
type	could be 'Moran', 'Greay', 'NormalizeMBorto', 'AC', 'CC', or 'ACC'. Also, it could be any combination of them.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

For CC and AAC autocovriance functions, which consider the covariance of the two physicochemical properties, we have provided users with the ability to categorize their selected properties in a list. The binary combination of each group will be taken into account. Note: If all the features are in a group or selectedAAidx parameter is a vector, the binary combination will be calculated for all the physicochemical properties.

Value

This function returns a feature matrix. The number of columns in the matrix changes depending on the chosen autocorrelation or autocovariance types and nlag parameter. The output is a matrix. The number of rows shows the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat1<-AutoCorDiNuc(seqs=fileLNC,selectedNucIdx=c(1:7),maxlag=20,type=c("Moran","Geary"))

mat2<-AutoCorDiNuc(seqs=fileLNC,selectedNucIdx=list(c(1,3),6:13,c(2:7)),maxlag=15,type=c("AC","CC"))
```

AutoCorTriNuc *Tri Nucleotide Autocorrelation-Autocovariance*

Description

It creates the feature matrix for each function in autocorelation (i.e., Moran, Greay, NormalizeMBorto) or autocovariance (i.e., AC, CC,ACC). The user can select any combination of the functions too. In this case, the final matrix will contain features of each selected function.

Usage

```
AutoCorTriNuc(
  seqs,
  selectedNucIdx,
  maxlag = 3,
  threshold = 1,
  type = c("Moran", "Geary", "NormalizeMBorto", "AC", "CC", "ACC"),
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	function takes as input the physicochemical properties. Users select the properties by their ids or indices in the nuIdx3 file. This parameter could be a vector or a list of trinucleotide indices.
maxlag	This parameter shows the maximum gap between two amino acids. The gaps change from 1 to maxlag (the maximum lag).
threshold	is a number between (0 to 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted.The default value is 1.

type	could be 'Moran', 'Greay', 'NormalizeMBorto', 'AC', 'CC', or 'ACC'. Also, it could be any combination of them.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

For CC and AAC autocovriance functions, which consider the covariance of the two physicochemical properties, we have provided users with the ability to categorize their selected properties in a list. The binary combination of each group will be taken into account. Note: If all the features are in a group or selectedAAidx parameter is a vector, the binary combination will be calculated for all the physicochemical properties.

Value

This function returns a feature matrix. The number of columns in the matrix changes depending on the chosen autocorrelation or autocovariance types and nlag parameter. The output is a matrix. The number of rows shows the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat1<-AutoCorTriNuc(seqs=fileLNC,selectedNucIdx=c(1:7),maxlag=20,type=c("Moran","Geary"))

mat2<-AutoCorTriNuc(seqs=fileLNC,selectedNucIdx=list(c(1,3),6:10,c(2:7)),
maxlag=15,type=c("AC","CC"))
```

BLOSUM62 *Blosum62*

Description

This function creates a 20-dimentional numeric vector for each amino acid of a sequence. Each entry of the vector contains the similarity score of the amino acid with other amino acids including itself. The score is extracted from the Blosum62 matrix.

Usage

```
BLOSUM62(seqs, label = c(), outFormat = "mat", outputFileDist = "")
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.

label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length)*20 and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
dir = tempdir()
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
filePr<-system.file("extdata/protein.fasta",package="ftrCOOL")
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

ad<-paste0(dir,"/blosum62.txt")
vect<-BLOSUM62(seqs = filePr,outFormat="mat")
BLOSUM62(seqs = filePrs,outFormat="txt",outputFileDist=ad)
```

CkSAApair *Composition of k-Spaced Amino Acids pairs*

Description

This function calculates the composition of k-spaced amino acid pairs. In other words, it computes the frequency of all amino acid pairs with k spaces.

Usage

```
CkSAApair(seqs, rng = 3, upto = FALSE, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
rng	This parameter can be a number or a vector. Each element of the vector shows the number of spaces between amino acid pairs. For each k in the rng vector, a new vector (whose size is 400) is created which contains the frequency of pairs with k gaps.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from [0 to rng].
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 400*(length of rng vector).

Note

'upto' is enabled only when rng is a number and not a vector.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-CkSAApair(seqs=filePrs,rng=2,upto=TRUE,normalized=TRUE)

mat2<-CkSAApair(seqs=filePrs,rng=c(1,3,5))
```

CkSGAApair *Composition of k-Spaced Grouped Amino Acids pairs*

Description

In this function, amino acids are first grouped into a category which is defined by the user. Later, the composition of the k-spaced grouped amino acids is computed. Please note that this function differs from CkSAApair which works on individual amino acids.

Usage

```
CkSGAApair(
  seqs,
  rng = 3,
  upto = FALSE,
  normalized = TRUE,
  Grp = "locFus",
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
rng	This parameter can be a number or a vector. Each element of the vector shows the number of spaces between amino acid pairs. For each k in the rng vector, a new vector (whose size is (number of categorizes)^2) is created which contains the frequency of pairs with k gaps.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from [0 to rng].
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Column names in the feature matrix follow G(?ss?). For example, G(1ss2) means Group1**Group2, where '*' is a wild character.

Value

This function returns a feature matrix. Row length is equal to the number of sequences and the number of columns is ((number of categorizes)^2)*(length of rng vector).

Note

'upto' is enabled only when rng is a number and not a vector.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-CkSGAApair(seqs=filePrs,rng=2,upto=TRUE,Grp="aromatic")

mat2<-CkSGAApair(seqs=filePrs,rng=c(1,3,5),upto=FALSE,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

CkSNUCpair *Composition of k-Spaced Nucleotides Pairs*

Description

This function calculates the composition of k-spaced nucleotide pairs. In other words, it computes the frequency of all nucleotide pairs witn k spaces.

Usage

```
CkSNUCpair(
  seqs,
  rng = 3,
  upto = FALSE,
  ORF = FALSE,
  reverseORF = TRUE,
  normalized = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
rng	This parameter can be a number or a vector. Each element of the vector shows the number of spaces between nucleotide pairs. For each k in the rng vector, a new vector (whose size is 16) is created which contains the frequency of pairs with k gaps.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from [0 to rng].
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).

normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 16*(length of rng vector).

Note

'upto' is enabled only when rng is a number and not a vector.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat1<-CkSNUCpair(seqs=fileLNC,rng=2,upto=TRUE,ORF=TRUE,reverseORF=FALSE)
mat2<-CkSNUCpair(seqs=fileLNC,rng=c(1,3,5))
```

---

codonAdaptionIndex *Codon Adaption Index*

Description

This function calculates the codon adaption index for each sequence.

Usage

```
codonAdaptionIndex(seqs, ORF = FALSE, reverseORF = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature vector. The length of the vector is equal to the number of sequences. Each entry in the vector contains the value of the codon adaption index.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-codonAdaptionIndex(seqs=fileLNC,ORF=TRUE,reverseORF=FALSE)
```

CodonFraction *Codon Fraction*

Description

This function calculates the codon fraction for each sequence.

Usage

```
CodonFraction(seqs, ORF = FALSE, reverseORF = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix such that the number of columns is 4^3 and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-CodonFraction(seqs=fileLNC,ORF=TRUE,reverseORF=FALSE)
```

CodonUsage *Codon Usage*

Description

This function calculates the codon usage for each sequence.

Usage

```
CodonUsage(seqs, ORF = FALSE, reverseORF = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix such that the number of columns is 4^3 and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-CodonUsage(fileLNC,ORF=TRUE,reverseORF=FALSE)
```

conjointTriad *Conjoint Triad*

Description

This function calculates the grouped tripeptide composition with the conjoint triad grouping type.

Usage

```
conjointTriad(seqs, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows equals to the number of sequences and the number of columns is 7^3.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-conjointTriad(seqs=filePrs)
```

---

conjointTriadKS *k-Spaced Conjoint Triad*

Description

This function calculates the grouped tripeptide composition with conjoint triad grouping type. For each k, it creates a 7^3 feature vector. K is the space between the first and the second amino acids and the second and the third amino acids of the tripeptide.

Usage

```
conjointTriadKS(seqs, rng = 3, upto = FALSE, normalized = FALSE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
rng	This parameter can be a number or a vector. Each element of the vector shows the number of spaces between the first and the second amino acids and the second and the third amino acids of the tripeptide. For each k in the rng vector, a new vector (whose size is 7^3) is created which contains the frequency of trinucleotide with k gaps.

upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from 0 to rng.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

A tripeptide with k spaces looks like AA1(ss..s)AA2(ss..s)AA3. AA stands for amino acids and s means space.

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (7^3)*(length rng vector).

Note

'upto' is enabled only when rng is a number and not a vector.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-conjointTriadKS(filePrs,rng=2,upto=TRUE,normalized=TRUE)

mat2<-conjointTriadKS(filePrs,rng=c(1,3,5))
```

CTD *Composition_Transition_Distribution*

Description

This function calculates the composition, transition, and distribution for each sequence.

Usage

```
CTD(seqs, normalized = FALSE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

Output is a combination of three different matrices: Composition, Transition, and Distribution. You can obtain any of the three matrices by executing the corresponding function, i.e., CTDC, CTDT, and CTDD.

References

Dubchak, Inna, et al. "Prediction of protein folding class using global description of amino acid sequence." Proceedings of the National Academy of Sciences 92.19 (1995): 8700-8704.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
CTDtotal<-CTD(seqs=filePrs,normalized=FALSE)
```

CTDC *CTDC(CTD Composition)*

Description

This function computes the composition part of CTD. Thirteen properties are defined in this function. Each property categorizes the amino acids of the sequences into three groups. The grouped amino acid composition is calculated for each property. For more information, please check the references.

Usage

```
CTDC(seqs, normalized = FALSE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 3*13, where three is the number of groups and thirteen is the number of properties.

References

Dubchak, Inna, et al. "Prediction of protein folding class using global description of amino acid sequence." Proceedings of the National Academy of Sciences 92.19 (1995): 8700-8704.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
CTD_C<-CTDC(seqs=filePrs,normalized=FALSE,label=c())
```

CTDD *CTDD(CTD Distribution)*

Description

This function computes the distribution part of CTD. It calculates fifteen values for each property. For more information, please check the references.

Usage

```
CTDD(seqs, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 15*13.

References

Dubchak, Inna, et al. "Prediction of protein folding class using global description of amino acid sequence." Proceedings of the National Academy of Sciences 92.19 (1995): 8700-8704.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
CTD_D<-CTDD(seqs=filePrs)
```

`CTDT` *CTDT(CTD Transition)*

Description

This function computes the transition part of CTD. Thirteen properties are defined in this function. Each property categorizes the amino acids of a sequence into three groups. For each property, the grouped amino acid transition (i.e., transitions 1-2, 1-3, and 2-3) is calculated. For more information, please check the references.

Usage

```
CTDT(seqs, normalized = FALSE, label = c())
```

Arguments

`seqs`	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
`normalized`	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
`label`	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 3*13, where three is the number of transition types (i.e., 1-2, 1-3, and 2-3) and thirteen is the number of properties.

References

Dubchak, Inna, et al. "Prediction of protein folding class using global description of amino acid sequence." Proceedings of the National Academy of Sciences 92.19 (1995): 8700-8704.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
CTD_T<-CTDT(seqs=filePrs,normalized=FALSE)
```

DDE *Dipeptide Deviation from Expected Mean value*

Description

This function computes the dipeptide deviation from the expected mean value.

Usage

```
DDE(seqs, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix with 20^2=400 number of columns. The number of rows is equal to the number of sequences.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-DDE(seqs=filePrs)
```

Dinuc2Binary *Dinucleotide To Binary*

Description

This function transforms a dinucleotide to a binary number with four bits which is enough to represent all the possible types of dinucleotides. The type of the binary format is determined by the binaryType parameter. For details about each format, please refer to the description of the binaryType parameter.

Usage

```
Dinuc2Binary(
  seqs,
  binaryType = "strBin",
  outFormat = "mat",
  outputFileDist = "",
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
binaryType	It can take any of the following values: ('strBin','logicBin','numBin'). 'strBin' (String binary): each dinucleotide is represented by a string containing 4 characters(0-1). For example, AA = "0000" AC="0001" ... TT="1111" 'logicBin' (logical value): Each amino acid is represented by a vector containing 4 logical entries. For example, AA = c(F,F,F,F) AC=c(F,F,F,T) ... TT=c(T,T,T,T) 'numBin' (numeric bin): Each amino acid is represented by a numeric (i.e., integer) vector containing 4 numeric entries. For example, AA = c(0,0,0,0) AC = c(0,0,0,1) ... TT = c(1,1,1,1)
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The output is different depending on the outFormat parameter ('mat' or 'txt'). If outFormat is set to 'mat', it returns a feature matrix for sequences with the same lengths. The number of rows is equal to the number of sequences and if binaryType is 'strBin', the number of columns is the length of the sequences. Otherwise, it is equal to (length of the sequences)*4. 'mat' format is used for machine learning purposes. If outFormat is 'txt', all binary values will be written to a 'txt' file. Each line in the file shows the binary format of a sequence.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-Dinuc2Binary(seqs = LNC50Nuc, binaryType="strBin",outFormat="mat")
```

DiNucIndex *Di Nucleotide Index (DiNucIndex)*

Description

This function replaces dinucleotides in a sequence with their physicochemical properties in the dinucleotide index file.

Usage

```
DiNucIndex(
  seqs,
  selectedNucIdx,
  standardized = TRUE,
  threshold = 1,
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	DiNucIndex function works based on physicochemical properties. Users, select the properties by their ids or indexes in dinucleotide file.
standardized	is a logical parameter. If it is set to TRUE, dinucleotide indices will be in the standard format. The default value is TRUE.
threshold	is a number between (0 , 1]. In selectedAAidx, indices with a correlation higher than the threshold will be deleted. The default value is 1.

label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Details

There are 125 physicochemical indexes in the dinucleotide database.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length)*(number of selected di-nucleotide indexes) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
fileLNC<-system.file("extdata/Athaliana1.fa",package="ftrCOOL")
vect<-DiNucIndex(seqs = fileLNC, selectedNucIdx=1:10,outFormat="mat")
```

EAAComposition *Enhanced Amino Acid Composition*

Description

This function slides a window over the input sequence(s). Also, it computes the composition of amino acids that appears within the limits of the window. Please note that when a feature vector of sequences is given as the input, their length should be equal. Otherwise, either an error (in case the outFortmat is 'mat') or a text file (when the outFortmat is 'txt') is returned. Please note that the text file is not suitable for machine learning purposes.

Usage

```
EAAComposition(
  seqs,
  winSize = 50,
  overLap = TRUE,
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
winSize	is a number which shows the size of the window.
overLap	This parameter shows how the window moves over the sequence. If overlap is set to FALSE, the window slides over the sequence in such a way that every time the window moves, it covers a unique portion of the sequence. Otherwise, portions of the sequence which appear within the window limits have "winSize-1" amino acids in common.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Details

Column names in the output matrix are Wi(aa), where aa shows an amino acid type ("A", "C", "D",..., "Y") and i indicates the number of times that the window has moved over the sequence(s).

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (20 * number of partitions displayed by the window) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

When overlap is FALSE, the last partition represented by the window may have a different length with other parts.

References

Chen, Zhen, et al. "iFeature: a python package and web server for features extraction and selection from protein and peptide sequences." Bioinformatics 34.14 (2018): 2499-2502.

Examples

```
dir = tempdir()
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat<-EAAComposition(seqs = ptmSeqsVect,winSize=50, overLap=FALSE,outFormat='mat')

ad<-paste0(dir,"/EaaCompos.txt")
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
EAAComposition(seqs = filePrs,winSize=50, overLap=FALSE,outFormat="txt"
,outputFileDist=ad)
```

EffectiveNumberCodon *Effective Number of Codon*

Description

This function calculates the effective number of codon for each sequence.

Usage

```
EffectiveNumberCodon(seqs, ORF = FALSE, reverseORF = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature vector. The length of the vector is equal to the number of sequences. Each entry in the vector contains the effective number of codon.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
vect<-EffectiveNumberCodon(seqs=fileLNC,ORF=TRUE,reverseORF=FALSE)
```

EGAAComposition *Enhanced Grouped Amino Acid Composition*

Description

In this function, amino acids are first grouped into user-defined categories. Then, enhanced grouped amino acid composition is computed. For details about the enhanced feature, please refer to function EAAComposition. Please note that this function differs from function EAAComposition which works on individual amino acids.

Usage

```
EGAAComposition(
  seqs,
  winSize = 50,
  overLap = TRUE,
  Grp = "locFus",
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
winSize	shows the size of sliding window. It is a numeric value.
overLap	This parameter shows how the window moves on the sequence. If the overlap is set to TRUE, the next window would have distance 1 with the previous window. Otherwise, the next window will start from the next amino acid after the previous window. There is no overlap between the next and previous windows.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is ((number of categorizes) * (number of windows)) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Examples

```
dir = tempdir()
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat1<-EGAAComposition(seqs = ptmSeqsVect,winSize=20,overLap=FALSE,Grp="locFus")

mat2<-EGAAComposition(seqs = ptmSeqsVect,winSize=30,overLap=FALSE,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")),outFormat="mat")

ad<-paste0(dir,"/EGrpaaCompos.txt")
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
EGAAComposition(seqs = filePrs,winSize=20,Grp="cTriad",outFormat="txt"
,outputFileDist=ad)
```

EIIP *Electron-ion interaction pseudopotentials (EIIP)*

Description

This function replaces each nucleotide in the input sequence with its electron-ion interaction value. The resulting sequence is represented by a feature vector whose length is equal to the length of the sequence. Please check the references for more information.

Usage

```
EIIP(seqs, outFormat = "mat", outputFileDist = "", label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.

outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is equal to the length of the sequences and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

References

Chen, Zhen, et al. "iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data." Briefings in bioinformatics 21.3 (2020): 1047-1057.

Examples

```
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-EIIP(seqs = LNC50Nuc,outFormat="mat")
```

ENUComposition *Enhanced nucleotide Composition*

Description

This function slides a window over the input sequence(s). Also, it computes the composition of nucleotides that appears within the limits of the window. Please note that when a feature vector of sequences is given as the input, their length should be equal. Otherwise, either an error (in case the outFortmat is 'mat') or a text file (when the outFortmat is 'txt') is returned. Please note that the text file is not suitable for machine learning purposes.

Usage

```
ENUComposition(
  seqs,
  winSize = 20,
  overLap = TRUE,
  outFormat = "mat",
  outputFileDist = "",
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
winSize	is a number which shows the size of the window.
overLap	This parameter shows how the window moves on the sequence. If the overlap is set to TRUE, the next window would have distance 1 with the previous window. Otherwise, the next window will start from the next amino acid after the previous window. There is no overlap between the next and previous windows.
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (4 * number of partitions displayed by the window) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

When overlap is FALSE, the last partition represented by the window may have a different length with other parts.

Examples

```
dir = tempdir()
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-ENUComposition(seqs = LNC50Nuc, winSize=20,outFormat="mat")

ad<-paste0(dir,"/ENUCcompos.txt")
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
ENUComposition(seqs = fileLNC,outFormat="txt",winSize=20
,outputFileDist=ad,overLap=FALSE)
```

ExpectedValKmerNUC *Expected Value for k-mers(Nuc)*

Description

This function is introduced by this package for the first time. It computes the expected value for each k-mer in a sequence. ExpectedValue(k-mer) = freq(k-mer) / (freq(nucleotide1) * freq(nucleotide2) * ... * freq(nucleotidek))

Usage

```
ExpectedValKmerNUC(
  seqs,
  k = 4,
  ORF = FALSE,
  reverseORF = TRUE,
  normalized = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
k	is an integer value. The default is four.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (4^k).

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-ExpectedValKmerNUC(seqs=fileLNC,k=4,ORF=TRUE,reverseORF=FALSE)
```

ExpectedValueAA *Expected Value for each Amino Acid*

Description

This function is introduced by this package for the first time. It computes the expected value for each k-mer in a sequence. ExpectedValue(k-mer) = freq(k-mer) / (c_1 * c_2 * ... * c_k), where c_i is the number of codons that encrypt the i'th amino acid in the k-mer.

Usage

```
ExpectedValueAA(seqs, k = 2, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is an integer value. The default is two.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is 20^k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-ExpectedValueAA(seqs=filePrs,k=2,normalized=FALSE)
```

ExpectedValueGAA *Expected Value for Grouped Amino Acid*

Description

This function is introduced by this package for the first time. In this function, amino acids are first grouped into user-defined categories. Later, the expected value of grouped amino acids is computed. Please note that this function differs from Function ExpectedValueAA which works on individual amino acids.

Usage

```
ExpectedValueGAA(seqs, k = 3, Grp = "locFus", normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is an integer value. The default is three.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

for more information about ExpectedValueGAA, please refer to function ExpectedValueKmer.

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (number of categories)^k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-ExpectedValueGAA(seqs=filePrs,k=2,Grp="locFus")

mat2<-ExpectedValueGAA(seqs=filePrs,k=1,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

ExpectedValueGKmerAA *Expected Value for Grouped k-mers*

Description

This function is introduced by this package for the first time. In this function, amino acids are first grouped into user-defined categories. Later, the expected value of grouped k-mer is computed. Please note that this function differs from Function ExpectedValueKmerAA which works on individual amino acids.

Usage

```
ExpectedValueGKmerAA(
  seqs,
  k = 2,
  Grp = "locFus",
  normalized = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is an integer. The default value is two.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (number of categorizes)^k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-ExpectedValueGKmerAA(seqs=filePrs,k=2,Grp="locFus")

mat2<-ExpectedValueGKmerAA(seqs=filePrs,k=1,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

ExpectedValueKmerAA *Expected Value for k-mers*

Description

This function computes the expected value of each k-mer by dividing the frequency of the kmer to multiplying frequency of each amino acid of the k-mer in the sequence.

Usage

```
ExpectedValueKmerAA(seqs, k = 2, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is an integer value and it shows the size of kmer in the kmer composition. The default value is 2.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

ExpectedValue(k-mer) = freq(k-mer) / (freq(aminoacid1) * freq(aminoacid2) * ... * freq(aminoacidk))

Value

This function returns a feature matrix. The number of rows equals the number of sequences and the number of columns if upto set false, is 20^k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-ExpectedValueKmerAA(filePrs,k=2,normalized=FALSE)
```

fa.read *Fasta File Reader*

Description

This function reads a FASTA file. Each sequence starts with '>' in the file. This is a general function which can be applied to all types of sequences (i.e., protein/peptide, dna, and rna).

Usage

```
fa.read(file, legacy.mode = TRUE, seqonly = FALSE, alphabet = "aa")
```

Arguments

file	The address of the FASTA file.
legacy.mode	comments all lines which start with ";".
seqonly	if it is set to true, the function will return sequences with no description.
alphabet	is a vector which contains amino acid, RNA, or DNA alphabets.

Value

a string vector such that each element is a sequence.

References

https://cran.r-project.org/web/packages/rDNAse/index.html

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
sequenceVectLNC<-fa.read(file=fileLNC,alphabet="dna")

filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
sequenceVectPRO<-fa.read(file=filePrs,alphabet="aa")
```

fickettScore *Ficket Score*

Description

This function calculates the ficket score of each sequence.

Usage

```
fickettScore(seqs, ORF = FALSE, reverseORF = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature vector. The length of the vector is equal to the number of sequences. Each entry in the vector contains the value of the fickett score.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
vect<-fickettScore(seqs=fileLNC,ORF=TRUE,reverseORF=FALSE)
```

GAAKpartComposition *Grouped Amino Acid K Part Composition*

Description

In this function, amino acids are first grouped into user-defined categories. Later, the composition of the grouped amino acid k part is computed. Please note that this function differs from AAKpart-Composition which works on individual amino acids.

Usage

```
GAAKpartComposition(
  seqs,
  k = 5,
  normalized = TRUE,
  Grp = "locFus",
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is an integer. Each sequence should be divided to k partition(s).
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

a feature matrix with k*(number of categorizes) number of columns. The number of rows is equal to the number of sequences.

Note

Warning: The length of all sequences should be greater than k.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-GAAKpartComposition(seqs=filePrs,k=5,Grp="aromatic")

mat2<-GAAKpartComposition(seqs=filePrs,k=3,normalized=FALSE,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

GCcontent *G_C content*

Description

This function calculates G-C content of each sequence.

Usage

```
GCcontent(seqs, ORF = FALSE, reverseORF = TRUE, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

The function returns a feature vector. The length of the vector is equal to the number of sequences. Each entry in the vector contains G-C content of a sequence.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
vect<-GCcontent(seqs=fileLNC,ORF=TRUE,reverseORF=FALSE)
```

GrpDDE *Group Dipeptide Deviation from Expected Mean*

Description

This function is introduced by this package for the first time. In this function, amino acids are first grouped into user-defined categories. Later, DDE is applied to grouped amino acids. Please note that this function differs from DDE which works on individual amino acids.

Usage

```
GrpDDE(seqs, Grp = "locFus", label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.

Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix with (number of categorizes)^2 number of columns. The number of rows is equal to the number of sequences.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-GrpDDE(seqs=filePrs,Grp="aromatic")

mat2<-GrpDDE(seqs=filePrs,Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

---

kAAComposition *K Amino Acid Composition*

Description

This function calculates the frequency of all k-mers in the sequence(s).

Usage

```
kAAComposition(seqs, rng = 3, upto = FALSE, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
rng	This parameter can be a number or a vector. Each entry of the vector holds the value of k in the k-mer composition.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from 1 to rng.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns depends on rng vector. For each value k in the vector, (20)^k columns are created in the matrix.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-kAAComposition(seqs=filePrs,rng=3,upto=TRUE)
mat2<-kAAComposition(seqs=filePrs,rng=c(1,3),upto=TRUE)
```

---

kGAAComposition *K Grouped Amino Acid Composition*

Description

In this function, amino acids are first grouped into user-defined categories. Later, the composition of the k grouped amino acids is computed. Please note that this function differs from kAAComposition which works on individual amino acids.

Usage

```
kGAAComposition(
  seqs,
  rng = 3,
  upto = FALSE,
  normalized = TRUE,
  Grp = "locFus",
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
rng	This parameter can be a number or a vector. Each entry of the vector holds the value of k in the k-mer composition. For each k in the rng vector, a new vector (whose size is 20^k) is created which contains the frequency of k-mers.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from 1 to rng.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.

Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

for more details, please refer to kAAComposition

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is ((number of categorizes)^k)*(length of rng vector).

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat1<-CkSGAApair(seqs=filePrs,rng=2,upto=TRUE,Grp="aromatic")

mat2<-CkSGAApair(seqs=filePrs,rng=c(1,3,5),Grp=
list(Grp1=c("G","A","V","L","M","I","F","Y","W"),Grp2=c("K","R","H","D","E")
,Grp3=c("S","T","C","P","N","Q")))
```

---

kNUComposition *K Nucleotide Composition*

Description

This function calculates the frequency of all k-mers in the sequence.

Usage

```
kNUComposition(
  seqs,
  rng = 3,
  reverse = FALSE,
  upto = FALSE,
  normalized = TRUE,
  ORF = FALSE,
  reverseORF = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
rng	This parameter can be a number or a vector. Each entry of the vector holds the value of k in the k-mer composition. For each k in the rng vector, a new vector (whose size is 20^k) is created which contains the frequency of kmers.
reverse	It is a logical parameter which assumes the reverse complement of the sequence.
upto	It is a logical parameter. The default value is FALSE. If rng is a number and upto is set to TRUE, rng is converted to a vector with values from 1 to rng.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns depends on the rng vector. For each value k in the vector, (4)^k columns are created in the matrix.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-kNUComposition(seqs=fileLNC,rng=c(1,3))
```

LocalPoSpKaaF *Local Position Specific k Amino Acids Frequency*

Description

For each sequence, this function creates a feature vector denoted as (f1,f2, f3, ..., fN), where fi = freq(i'th k-mer of the sequence) / i. It should be applied to sequences with the same length.

Usage

```
LocalPoSpKaaF(seqs, k = 2, label = c(), outFormat = "mat", outputFileDist = "")
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	is a numeric value which holds the value of k in the k-mers.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length-k+1) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Examples

```
dir = tempdir()
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat<-LocalPoSpKaaF(seqs = ptmSeqsVect, k=2,outFormat="mat")

ad<-paste0(dir,"/LocalPoSpKaaF.txt")
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
LocalPoSpKaaF(seqs = filePrs, k=1,outFormat="txt"
,outputFileDist=ad)
```

LocalPoSpKnucF *Local Position Specific k Nucleotide Frequency*

Description

For each sequence, this function creates a feature vector denoted as (f1,f2, f3, ..., fN), where fi = freq(i'th k-mer of the sequence) / i. It should be applied to sequences with the same length.

Usage

```
LocalPoSpKnucF(
  seqs,
  k = 2,
```

```
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
k	is a numeric value which holds the value of k in the k-mers.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length-k+1) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Examples

```
dir = tempdir()
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-LocalPoSpKnucF(seqs = LNC50Nuc, k=2,outFormat="mat")

ad<-paste0(dir,"/LocalPoSpKnucF.txt")
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
LocalPoSpKnucF(seqs = fileLNC,k=1,outFormat="txt"
,outputFileDist=ad)
```

---

maxORF *Maximum Open Reading Frame*

Description

This function gets a sequence as the input. If reverse is true, the function extracts the max Open Reading Frame in the sequence and its reverse complement (hint: Six frames). Otherwise, only the sequence is searched (hint: Three frames).

Usage

```
maxORF(seqs, reverse = TRUE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
reverse	It is a logical parameter which assumes the reverse complement of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A vector containing a subsequence for each given sequences. The subsequence is the maximum ORF of the sequence.

Note

If a sequence does not contain ORF, the function deletes the sequence.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
ORF<-maxORF(seqs=fileLNC,reverse=FALSE)
```

---

maxORFlength *Maximum Open Reading Frame length*

Description

This function returns the length of the maximum Open Reading Frame for each sequence. If reverse is FALSE, ORF region will be searched in a sequence. Otherwise, it will be searched both in the sequence and its reverse complement.

Usage

```
maxORFlength(seqs, reverse = TRUE, normalized = FALSE, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
reverse	It is a logical parameter which assumes the reverse complement of the sequence.

normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A vector containing the lengths of maximum ORFs for each sequence.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
vect<-maxORFlength(seqs=fileLNC,reverse=TRUE,normalized=TRUE)
```

---

nameKmer *nameKmer*

Description

This function creates all possible k-combinations of the given alphabets.

Usage

```
nameKmer(k = 3, type = "aa", num = 0)
```

Arguments

k	is a numeric value.
type	can be one of "aa", "rna", "dna", or "num".
num	When type is set to "num", it shows the numeric alphabet(1,..,,num).

Value

a string vector of length (20^k for 'aa' type), (4^k for 'dna' type), (4^k for 'rna' type), and (num^k for 'num' type).

Examples

```
all_kmersAA<-nameKmer(k=2,type="aa")

all_kmersDNA<-nameKmer(k=3,type="dna")

all_kmersNUM<-nameKmer(k=3,type="num",num=2)
```

needleman *Needleman-Wunsch*

Description

This function works based on Needleman-Wunsch algorithm which computes similarity score of two sequences.

Usage

```
needleman(seq1, seq2, gap = -1, mismatch = -1, match = 1)
```

Arguments

seq1	(sequence1) is a string.
seq2	(sequence2) is a string.
gap	The penalty for gaps in sequence alignment. Usually, it is a negative value.
mismatch	The penalty for the mismatch in the sequence alignment. Usually, it is a negative value.
match	A score for the match in sequence alignment. Usually, it is a positive value.

Value

The function returns a number which indicates the similarity between sequence1 and sequence2.

References

https://gist.github.com/juliuskittler/ed53696ac1e590b413aac2dddf0457f6

Examples

```
simScore<-needleman(seq1="Hello",seq2="Hello",gap=-1,mismatch=-2,match=1)
```

nonStandardSeq *nonStandard amino acid sequence*

Description

This function returns sequences which contain at least one non-standard alphabet.

Usage

```
nonStandardSeq(file, legacy.mode = TRUE, seqonly = FALSE, alphabet = "aa")
```

Arguments

file	The address of fasta file which contains all the sequences.
legacy.mode	It comments all lines starting with ";"
seqonly	If it is set to true, the function returns sequences with no description.
alphabet	It is a vector which contains the amino acid, RNA, or DNA alphabets.

Value

This function returns a string vector. Each element of the vector is a sequence which contains at least one non-standard alphabet.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
nonStandardPrSeq<-nonStandardSeq(file = filePrs,alphabet="aa")

fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
nonStandardNUCSeq<-nonStandardSeq(file = filePrs, alphabet="dna")
```

---

novel_PseKNC *Novel Pseudo k Nucleotide Composition (series)*

Description

This function replaces nucleotides with a four-length vector. The first three elements represent the nucleotides and the forth holds the frequency of the nucleotide from the beginning of the sequence until the position of the nucleotide in the sequence. 'A' will be replaced with c(1, 1, 0, freq), 'C' with c(0, 1, 1, freq),'G' with c(1, 0, 1, freq), and 'T' with c(0, 0, 0, freq).

Usage

```
novel_PseKNC(seqs, outFormat = "mat", outputFileDist = "", label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix. The number of rows is equal to the number of sequences.

References

Feng, Pengmian, et al. "iDNA6mA-PseKNC: Identifying DNA N6-methyladenosine sites by incorporating nucleotide physicochemical properties into PseKNC." Genomics 111.1 (2019): 96-102.

Examples

```
dir = tempdir()
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-novel_PseKNC(seqs = LNC50Nuc,outFormat="mat")

ad<-paste0(dir,"/ENUCcompos.txt")
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
novel_PseKNC(seqs = fileLNC,outFormat="txt",outputFileDist=ad)
```

NUC2Binary *Nucleotide To Binary*

Description

This function transforms a nucleotide to a binary format. The type of the binary format is determined by the binaryType parameter. For details about each format, please refer to the description of the binaryType parameter.

Usage

```
NUC2Binary(
  seqs,
  binaryType = "strBin",
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
binaryType	It can take any of the following values: ('strBin','logicBin','numBin'). 'strBin'(String binary): each nucleotide is represented by a string containing 4 characters(0-1). A = "0001" , C = "0010" , G = "0100" , T = "1000" 'logicBin'(logical value):

	Each nucleotide is represented by a vector containing 4 logical entries. A = c(F,F,F,T) , ... , T = c(T,F,F,F) 'numBin' (numeric bin): Each nucleotide is represented by a numeric (i.e., integer) vector containing 4 numerals. A = c(0,0,0,1) , ... , T = c(1,0,0,0)
`label`	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
`outFormat`	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
`outputFileDist`	shows the path and name of the 'txt' output file.

Value

The output is different depending on the outFormat parameter ('mat' or 'txt'). If outFormat is set to 'mat', it returns a feature matrix for sequences with the same lengths. The number of rows is equal to the number of sequences and if binaryType is 'strBin', the number of columns is the length of the sequences. Otherwise, it is equal to (length of the sequences)*4. 'mat' format is used for machine learning purposes. If outFormat is 'txt', all binary values will be written to a 'txt' file. Each line in the file shows the binary format of a sequence.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
dir = tempdir()
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-NUC2Binary(seqs = LNC50Nuc,outFormat="mat")

ad<-paste0(dir,"/NUC2Binary.txt")
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
NUC2Binary(seqs = fileLNC,binaryType="strBin",outFormat="txt",outputFileDist=ad)
```

NUCKpartComposition *Nucleotide to K Part Composition*

Description

In this function, each sequence is divided into k equal partitions. The length of each part is equal to ceiling(l(lenght of the sequence)/k). The last part can have a different length containing the residual nucleotides. The nucleotide composition is calculated for each part.

Usage

```
NUCKpartComposition(
  seqs,
  k = 5,
  ORF = FALSE,
  reverseORF = TRUE,
  normalized = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
k	is an integer value. Each sequence should be divided to k partition(s).
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

a feature matrix with k*4 number of columns. The number of rows is equal to the number of sequences.

Note

Warning: The length of all sequences should be greater than k.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-NUCKpartComposition(seqs=fileLNC,k=5,ORF=TRUE,reverseORF=FALSE,normalized=FALSE)
```

PSEAAC *Pseudo-Amino Acid Composition(PSEAAC) (Parallel)*

Description

This function calculates the pseudo amino acid composition (parallel) for each sequence.

Usage

```
PSEAAC(
  seqs,
  aaIDX = c("ARGP820101", "HOPT810101", "Mass"),
  lambda = 3,
  w = 0.05,
  l = 1,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
aaIDX	is a vector of Ids or indexes of the user-selected physicochemical properties in the aaIndex2 database. The default values of the vector are the hydrophobicity ids and hydrophilicity ids and Mass of residual in the amino acid index file.
lambda	is a tuning parameter. Its value indicates the maximum number of spaces between di-nucleotide pairs. The number changes from 1 to lambda.
w	(weight) is a tuning parameter. It changes in from 0 to 1. The default value is 0.5.
l	This parameter keeps the value of l in lmer composition. The lmers form the first 20^l elements of the APAAC descriptor.
threshold	is a number between (0 , 1]. It deletes aaIndexes which have a correlation bigger than the threshold. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

A feature matrix such that the number of columns is 20^l+(lambda) and the number of rows is equal to the number of sequences.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-PSEAAC(seqs=filePrs,l=2)
```

PseEIIP *Pseudo Electron-ion Interaction Pseudopotentials of Trinucleotide(PseEIIP)*

Description

This function calculates the pseudo electron-ion interaction for each sequence. It creates a feature vector for each sequence. The vector contains a value for each for each tri-nucleotide. The value is computed by multiplying the aggregate value of electron-ion interaction of each nucleotide

Usage

```
PseEIIP(seqs, label = c())
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

This function returns a feature matrix which the number of rows is equal to the number of sequences and the number of columns is 4^3=64.

References

Chen, Zhen, et al. "iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data." Briefings in bioinformatics 21.3 (2020): 1047-1057.

Examples

```
LNCSeqsADR<-system.file("extdata/",package="ftrCOOL")
LNC50Nuc<-as.vector(read.csv(paste0(LNCSeqsADR,"/LNC50Nuc.csv"))[,2])
mat<-PseEIIP(seqs = LNC50Nuc)
```

PSEkNCdi *Pseudo k Nucleotide Composition-Di(Parallel)*

Description

This function calculates the pseudo-k nucleotide composition(Di) (Parallel) for each sequence.

Usage

```
PSEkNCdi(
  seqs,
  selectedNucIdx,
  lambda = 3,
  w = 0.05,
  l = 2,
  ORF = FALSE,
  reverseORF = TRUE,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	is a vector of Ids or indices of the desired physicochemical properties of dinucleotides. Users can choose the desired indices by their ids or their names in the dinucleotide peoperties file.
lambda	is a tuning parameter. This integer value shows the maximum limit of spaces between dinucleotide pairs. The Number of spaces changes from 1 to lambda.
w	(weight) is a tuning parameter. It changes in the range of 0 to 1. The default value is 0.5.
l	This parameter keeps the value of l in lmer composition. The lmers form the first 4^l elements of the APkNCdi descriptor.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
threshold	is a number between (0 , 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

This function computes the pseudo nucleotide composition for each physicochemical property of di-nucleotides. We have provided users with the ability to choose among the 125 properties in the di-nucleotide index database.

Value

a feature matrix such that the number of columns is 4^l+lambda and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-PSEkNCdi(seqs=fileLNC,selectedNucIdx=1:5,l=2,ORF=TRUE,threshold=0.8)
```

PSEkNCTri *Pseudo k Nucleotide Composition-Tri(Parallel)*

Description

This function calculates pseudo-k nucleotide composition(Tri) (Parallel) for each sequence.

Usage

```
PSEkNCTri(
  seqs,
  selectedNucIdx,
  lambda = 3,
  w = 0.05,
  l = 3,
  ORF = FALSE,
  reverseORF = TRUE,
  threshold = 1,
  label = c()
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	is a vector of Ids or indices of the desired physicochemical properties of trinucleotides. Users can choose the desired indices by their ids or their names in the tri-nucleotide peoperties file.
lambda	is a tuning parameter. This integer value shows the maximum limit of spaces between di-nucleotide pairs. The Number of spaces changes from 1 to lambda.

w	(weight) is a tuning parameter. It can take a value in the range 0 to 1. The default value is 0.5.
l	This parameter keeps the value of l in lmer composition. The lmers form the first 4^l elements of the APkNCTri descriptor.
ORF	(Open Reading Frame) is a logical parameter. If it is set to true, ORF region of each sequence is considered instead of the original sequence (i.e., 3-frame).
reverseORF	is a logical parameter. It is enabled only if ORF is true. If reverseORF is true, ORF region will be searched in the sequence and also in the reverse complement of the sequence (i.e., 6-frame).
threshold	is a number between (0 , 1]. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted. The default value is 1.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

This function computes the pseudo nucleotide composition for each physicochemical property of trinucleotides. We have provided users with the ability to choose among the 12 properties in the tri-nucleotide index database.

Value

a feature matrix such that the number of columns is 4^l+lambda and the number of rows is equal to the number of sequences.

Examples

```
fileLNC<-system.file("extdata/Athaliana_LNCRNA.fa",package="ftrCOOL")
mat<-PSEkNCTri(seqs=fileLNC,selectedNucIdx=1:12, l=2,ORF=TRUE,threshold=0.8)
```

PseKRAAC_T1 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_1)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type1(PseKRAAC_T1) contains Grp 2 to 20.

Usage

```
PseKRAAC_T1(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 2,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c("CMFILVWY", "AGTSNQDEHRKP"), 3=c("CMFILVWY", "AGTSP", "NQDEHRK"), 4=c("CMFWY", "ILV", "AGTS", "NQDEHRKP"), 5=c("WFYH", "MILV", "CATSP", "G", "NQDERK"), 6=c("WFYH", "MILV", "CATS", "P", "G", "NQDERK"), 7=c("WFYH", "MILV", "CATS", "P", "G", "NQDE", "RK"), 8=c("WFYH", "MILV", "CA", "NTS", "P", "G", "DE", "QRK"), 9=c("WFYH", "MI", "LV", "CA", "NTS", "P", "G", "DE", "QRK"), 10=c("WFY", "ML", "IV", "CA", "TS", "NH", "P", "G", "DE", "QRK"), 11=c("WFY", "ML", "IV", "CA", "TS", "NH", "P", "G", "D", "QE", "RK"), 12=c("WFY", "ML", "IV", "C", "A", "TS", "NH", "P", "G", "D", "QE", "RK"), 13=c("WFY", "ML", "IV", "C", "A", "T", "S", "NH", "P", "G", "D", "QE", "RK"), 14=c("WFY", "ML", "IV", "C", "A", "T", "S", "NH", "P", "G", "D", "QE", "R", "K"), 15=c("WFY", "ML", "IV", "C", "A", "T", "S", "N", "H", "P", "G", "D", "QE", "R", "K"), 16=c("W", "FY", "ML", "IV", "C", "A", "T", "S", "N", "H", "P", "G", "D", "QE", "R", "K"), 17=c("W", "FY", "ML", "IV", "C", "A", "T", "S", "N", "H", "P", "G", "D", "Q", "E", "R", "K"), 18=c("W", "FY", "M", "L", "IV", "C", "A", "T", "S", "N", "H", "P", "G", "D", "Q", "E", "R", "K"), 19=c("W", "F", "Y", "M", "L", "IV", "C", "A", "T", "S", "N", "H", "P", "G", "D", "Q", "E", "R", "K"), 20=c("W", "F", "Y", "M", "L", "I", "V", "C", "A", "T", "S", "N", "H", "P", "G", "D", "Q", "E", "R", "K")

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T1(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T1(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T10 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_10)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type10(PseKRAAC_T10) contains Grp 2-20.

Usage

```
PseKRAAC_T10(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.

`Grp` is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.

`GapOrLambdaValue` is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.

`k` This parameter keeps the value of k in k-mer.

`label` is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c('CMFILVWY', 'AGTSNQDEHRKP'), 3=c('CMFILVWY', 'AGTSP', 'NQDEHRK'), 4=c('CMFWY', 'ILV', 'AGTS', 'NQDEHRKP'), 5=c('FWYH', 'MILV', 'CATSP', 'G', 'NQDERK'), 6=c('FWYH', 'MILV', 'CATS', 'P', 'G', 'NQDERK'), 7=c('FWYH', 'MILV', 'CATS', 'P', 'G', 'NQDE', 'RK'), 8=c('FWYH', 'MILV', 'CA', 'NTS', 'P', 'G', 'DE', 'QRK'), 9=c('FWYH', 'ML', 'IV', 'CA', 'NTS', 'P', 'G', 'DE', 'QRK'), 10=c('FWY', 'ML', 'IV', 'CA', 'TS', 'NH', 'P', 'G', 'DE', 'QRK'), 11=c('FWY', 'ML', 'IV', 'CA', 'TS', 'NH', 'P', 'G', 'D', 'QE', 'RK'), 12=c('FWY', 'ML', 'IV', 'C', 'A', 'TS', 'NH', 'P', 'G', 'D', 'QE', 'RK'), 13=c('FWY', 'ML', 'IV', 'C', 'A', 'T', 'S', 'NH', 'P', 'G', 'D', 'QE', 'RK'), 14=c('FWY', 'ML', 'IV', 'C', 'A', 'T', 'S', 'NH', 'P', 'G', 'D', 'QE', 'R', 'K'), 15=c('FWY', 'ML', 'IV', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'QE', 'R', 'K'), 16=c('W', 'FY', 'ML', 'IV', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'QE', 'R', 'K'), 17=c('W', 'FY', 'ML', 'IV', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'Q', 'E', 'R', 'K'), 18=c('W', 'FY', 'M', 'L', 'IV', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'Q', 'E', 'R', 'K'), 19=c('W', 'F', 'Y', 'M', 'L', 'IV', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'Q', 'E', 'R', 'K'), 20=c('W', 'F', 'Y', 'M', 'L', 'I', 'V', 'C', 'A', 'T', 'S', 'N', 'H', 'P', 'G', 'D', 'Q', 'E', 'R', 'K')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T10(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T10(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T11 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_11)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type11(PseKRAAC_T11) contains Grp 2-20.

Usage

```
PseKRAAC_T11(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c('CFYWMLIV', 'GPATSNHQEDRK'), 3=c('CFYWMLIV', 'GPATS', 'NHQEDRK'), 4=c('CFYW', 'MLIV', 'GPATS', 'NHQEDRK'), 5=c('CFYW', 'MLIV', 'G', 'PATS', 'NHQEDRK'), 6=c('CFYW', 'MLIV', 'G', 'P', 'ATS', 'NHQEDRK'), 7=c('CFYW', 'MLIV', 'G', 'P', 'ATS', 'NHQED', 'RK'), 8=c('CFYW', 'MLIV', 'G', 'P', 'ATS', 'NH', 'QED', 'RK'), 9=c('CFYW', 'ML', 'IV', 'G', 'P', 'ATS', 'NH', 'QED', 'RK'), 10=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'ATS',

'NH', 'QED', 'RK'), 11=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'A', 'TS', 'NH', 'QED', 'RK'), 12=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'A', 'TS', 'NH', 'QE', 'D', 'RK'), 13=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'A', 'T', 'S', 'NH', 'QE', 'D', 'RK'), 14=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'QE', 'D', 'RK'), 15=c('C', 'FYW', 'ML', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'QE', 'D', 'R', 'K'), 16=c('C', 'FY', 'W', 'ML', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'QE', 'D', 'R', 'K'), 17=c('C', 'FY', 'W', 'ML', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'Q', 'E', 'D', 'R', 'K'), 18=c('C', 'FY', 'W', 'M', 'L', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'Q', 'E', 'D', 'R', 'K'), 19=c('C', 'F', 'Y', 'W', 'M', 'L', 'IV', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'Q', 'E', 'D', 'R', 'K'), 20=c('C', 'F', 'Y', 'W', 'M', 'L', 'I', 'V', 'G', 'P', 'A', 'T', 'S', 'N', 'H', 'Q', 'E', 'D', 'R', 'K')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T11(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T11(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T12 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_12)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type12(PseKRAAC_T12) contains Grp 2-18,20.

Usage

```
PseKRAAC_T12(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c('IVMLFWYC', 'ARNDQEGHKPST'), 3=c('IVLMFWC', 'YA', 'RNDQEGHKPST'), 4=c('IVLMFW', 'C', 'YA', 'RNDQEGHKPST'), 5=c('IVLMFW', 'C', 'YA', 'G', 'RNDQEHKPST'), 6=c('IVLMF', 'WY', 'C', 'AH', 'G', 'RNDQEKPST'), 7=c('IVLMF', 'WY', 'C', 'AH', 'GP', 'R', 'NDQEKST'), 8=c('IVLMF', 'WY', 'C', 'A', 'G', 'R', 'Q', 'NDEHKPST'), 9=c('IVLMF', 'WY', 'C', 'A', 'G', 'P', 'H', 'K', 'RNDQEST'), 10=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'RN', 'DQEKPST'), 11=c('IVLMF', 'W', 'Y', 'C', 'A', 'H', 'G', 'R', 'N', 'Q', 'DEKPST'), 12=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'T', 'RDEKPS'), 13=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'DEKST'), 14=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'DEST'), 15=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'D', 'EST'), 16=c('IVLM', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'S', 'T', 'DE'), 17=c('IVL', 'M', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'S', 'T', 'DE'), 18=c('IVL', 'M', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'S', 'T', 'D', 'E'), 20=c('I', 'V', 'L', 'M', 'F', 'W', 'Y', 'C', 'A', 'H', 'G', 'N', 'Q', 'P', 'R', 'K', 'S', 'T', 'D', 'E')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T12(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)
```

```
mat2<-PseKRAAC_T12(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T13 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_13)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type13(PseKRAAC_T13) contains Grp 4,12,17,20.

Usage

```
PseKRAAC_T13(
  seqs,
  type = "gap",
  Grp = 4,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 4=c('ADKERNTSQ', 'YFLIVMCWH', 'G', 'P'), 12=c('A', 'D', 'KER', 'N', 'TSQ', 'YF', 'LIVM', 'C', 'W', 'H', 'G', 'P'), 17=c('A', 'D', 'KE', 'R', 'N', 'T', 'S', 'Q', 'Y', 'F', 'LIV', 'M', 'C', 'W', 'H', 'G', 'P'), 20=c('A', 'D', 'K', 'E', 'R', 'N', 'T', 'S', 'Q', 'Y', 'F', 'L', 'I', 'V', 'M', 'C', 'W', 'H', 'G', 'P')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T13(seqs=filePrs,type="gap",Grp=17,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T13(seqs=filePrs,type="lambda",Grp=17,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T14 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_14)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type14(PseKRAAC_T14) contains Grp 2-20.

Usage

```
PseKRAAC_T14(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.

`Grp` is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.

`GapOrLambdaValue` is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.

`k` This parameter keeps the value of k in k-mer.

`label` is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c('ARNDCQEGHKPST', 'ILMFWYV'), 3=c('ARNDQEGHKPST', 'C', 'ILMFWYV'), 4=c('ARNDQEGHKPST', 'C', 'ILMFYV', 'W'), 5=c('AGPST', 'RNDQEHK', 'C', 'ILMFYV', 'W'), 6=c('AGPST', 'RNDQEK', 'C', 'H', 'ILMFYV', 'W'), 7=c('ANDGST', 'RQEK', 'C', 'H', 'ILMFYV', 'P', 'W'), 8=c('ANDGST', 'RQEK', 'C', 'H', 'ILMV', 'FY', 'P', 'W'), 9=c('AGST', 'RQEK', 'ND', 'C', 'H', 'ILMV', 'FY', 'P', 'W'), 10=c('AGST', 'RK', 'ND', 'C', 'QE', 'H', 'ILMV', 'FY', 'P', 'W'), 11=c('AST', 'RK', 'ND', 'C', 'QE', 'G', 'H', 'ILMV', 'FY', 'P', 'W'), 12=c('AST', 'RK', 'ND', 'C', 'QE', 'G', 'H', 'IV', 'LM', 'FY', 'P', 'W'), 13=c('AST', 'RK', 'N', 'D', 'C', 'QE', 'G', 'H', 'IV', 'LM', 'FY', 'P', 'W'), 14=c('AST', 'RK', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'LM', 'FY', 'P', 'W'), 15=c('A', 'RK', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'LM', 'FY', 'P', 'ST', 'W'), 16=c('A', 'RK', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'LM', 'F', 'P', 'ST', 'W', 'Y'), 17=c('A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'LM', 'K', 'F', 'P', 'ST', 'W', 'Y'), 18=c('A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'LM', 'K', 'F', 'P', 'S', 'T', 'W', 'Y'), 19=c('A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'IV', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W', 'Y'), 20=c('A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'I', 'V', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W', 'Y')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T14(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T14(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T15 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_15)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type15(PseKRAAC_T15) contains Grp 2-16,20.

Usage

```
PseKRAAC_T15(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups:

Grp2=c('MFILVAW', 'CYQHPGTSNRKDE'), Grp3=c('MFILVAW', 'CYQHPGTSNRK', 'DE'), Grp4=c('MFILV', 'ACW', 'YQHPGTSNRK', 'DE'), Grp5=c('MFILV', 'ACW', 'YQHPGTSN', 'RK', 'DE'), Grp6=c('MFILV', 'A', 'C', 'WYQHPGTSN', 'RK', 'DE'), Grp7=c('MFILV', 'A',

'C', 'WYQHP', 'GTSN', 'RK', 'DE'), Grp8=c('MFILV', 'A', 'C', 'WYQHP', 'G', 'TSN', 'RK', 'DE'), Grp9=c('MF', 'ILV', 'A', 'C', 'WYQHP', 'G', 'TSN', 'RK', 'DE'), Grp10=c('MF', 'ILV', 'A', 'C', 'WYQHP', 'G', 'TSN', 'RK', 'D', 'E'), Grp11=c('MF', 'IL', 'V', 'A', 'C', 'WYQHP', 'G', 'TSN', 'RK', 'D', 'E'), Grp12=c('MF', 'IL', 'V', 'A', 'C', 'WYQHP', 'G', 'TS', 'N', 'RK', 'D', 'E'), Grp13=c('MF', 'IL', 'V', 'A', 'C', 'WYQHP', 'G', 'T', 'S', 'N', 'RK', 'D', 'E'), Grp14=c('MF', 'I', 'L', 'V', 'A', 'C', 'WYQHP', 'G', 'T', 'S', 'N', 'RK', 'D', 'E'), Grp15=c('MF', 'IL', 'V', 'A', 'C', 'WYQ', 'H', 'P', 'G', 'T', 'S', 'N', 'RK', 'D', 'E'), Grp16=c('MF', 'I', 'L', 'V', 'A', 'C', 'WYQ', 'H', 'P', 'G', 'T', 'S', 'N', 'RK', 'D', 'E'), Grp20=c('M', 'F', 'I', 'L', 'V', 'A', 'C', 'W', 'Y', 'Q', 'H', 'P', 'G', 'T', 'S', 'N', 'R', 'K', 'D', 'E')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T15(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T15(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T16 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_16)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type16(PseKRAAC_T16) contains Grp 2-16,20.

Usage

```
PseKRAAC_T16(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups:

2=c('IMVLFWY','GPCASTNHQEDRK'), 3=c('IMVLFWY','GPCAST','NHQEDRK'), 4=c('IMVLFWY','G','PCAST','NHQEDRK'), 5=c('IMVL','FWY','G','PCAST','NHQEDRK'), 6=c('IMVL','FWY','G','P','CAST','NHQEDRK'), 7=c('IMVL','FWY','G','P','CAST','NHQED','RK'), 8=c('IMV','L','FWY','G','P','CAST','NHQED','RK'), 9=c('IMV','L','FWY','G','P','C','AST','NHQED','RK'), 10=c('IMV','L','FWY','G','P','C','A','STNH','RKQE','D'), 11=c('IMV','L','FWY','G','P','C','A','STNH','RKQ','E','D'), 12=c('IMV','L','FWY','G','P','C','A','ST','N','HRKQ','E','D'), 13=c('IMV','L','F','WY','G','P','C','A','ST','N','HRKQ','E','D'), 14=c('IMV','L','F','WY','G','P','C','A','S','T','N','HRKQ','E','D'), 15=c('IMV','L','F','WY','G','P','C','A','S','T','N','H','RKQ','E','D'), 16=c('IMV','L','F','W','Y','G','P','C','A','S','T','N','H','RKQ','E','D'), 20=c('I','M','V','L','F','W','Y','G','P','C','A','S','T','N','H','R','K','Q','E','D')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T16(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)
```

```
mat2<-PseKRAAC_T16(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T2 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_2)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type2(PseKRAAC_T2) contains Grp 2-6,8,15,20.

Usage

```
PseKRAAC_T2(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups:

2=c('LVIMCAGSTPFYW', 'EDNQKRH'), 3=c('LVIMCAGSTP', 'FYW', 'EDNQKRH'), 4=c('LVIMC', 'AGSTP', 'FYW', 'EDNQKRH'), 5=c('LVIMC', 'AGSTP', 'FYW', 'EDNQ', 'KRH'), 6=c('LVIM', 'AGST', 'PHC', 'FYW', 'EDNQ', 'KR'), 8=c('LVIMC', 'AG', 'ST', 'P', 'FYW', 'EDNQ', 'KR', 'H'), 15=c('LVIM', 'C', 'A', 'G', 'S', 'T', 'P', 'FY', 'W', 'E', 'D', 'N', 'Q', 'KR', 'H'), 20=c('L', 'V', 'I', 'M', 'C', 'A', 'G', 'S', 'T', 'P', 'F', 'Y', 'W', 'E', 'D', 'N', 'Q', 'K', 'R', 'H')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T2(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T2(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T3A *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_3B)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type3 contain two type: type3A and type3B. 'PseKRAAC_T3A' contains Grp 2-20.

Usage

```
PseKRAAC_T3A(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: Grp2=c('AGSPDEQNHTKRMILFYVC', 'W'), Grp3=c('AGSPDEQNHTKRMILFYV', 'W', 'C'), Grp4=c('AGSPDEQNHTKRMIV', 'W', 'YFL', 'C'), Grp5=c('AGSPDEQNHTKR', 'W', 'YF', 'MIVL', 'C'), Grp6=c('AGSP', 'DEQNHTKR', 'W', 'YF', 'MIL', 'VC'), Grp7=c('AGP', 'DEQNH', 'TKRMIV', 'W', 'YF', 'L', 'CS'), Grp8=c('AG', 'DEQN', 'TKRMIV', 'HY', 'W', 'L', 'FP', 'CS'), Grp9=c('AG', 'P', 'DEQN', 'TKRMI', 'HY', 'W', 'F', 'L', 'VCS'), Grp10=c('AG', 'P', 'DEQN', 'TKRM', 'HY', 'W', 'F', 'I', 'L', 'VCS'), Grp11=c('AG', 'P', 'DEQN', 'TK', 'RI', 'H', 'Y', 'W', 'F', 'ML', 'VCS'), Grp12=c('FAS', 'P', 'G', 'DEQ', 'NL', 'TK', 'R', 'H', 'W', 'Y', 'IM', 'VC'), Grp13=c('FAS', 'P', 'G', 'DEQ', 'NL', 'T', 'K', 'R', 'H', 'W', 'Y', 'IM', 'VC'), Grp14=c('FA', 'P', 'G', 'T', 'DE', 'QM', 'NL', 'K', 'R', 'H', 'W', 'Y', 'IV', 'CS'), Grp15=c('FAS', 'P', 'G', 'T', 'DE', 'Q', 'NL', 'K', 'R', 'H', 'W', 'Y', 'M', 'I', 'VC'), Grp16=c('FA', 'P', 'G', 'ST', 'DE', 'Q', 'N', 'K', 'R', 'H', 'W', 'Y', 'M', 'L', 'I', 'VC'), Grp17=c('FA', 'P', 'G', 'S', 'T', 'DE', 'Q', 'N', 'K', 'R', 'H', 'W', 'Y', 'M', 'L', 'I', 'VC'), Grp18=c('FA', 'P', 'G', 'S', 'T', 'DE', 'Q', 'N', 'K', 'R', 'H', 'W', 'Y', 'M', 'L', 'I', 'V', 'C'), Grp19=c('FA', 'P', 'G', 'S', 'T', 'D', 'E', 'Q', 'N', 'K', 'R', 'H', 'W', 'Y', 'M', 'L', 'I', 'V', 'C'), Grp20=c('F', 'A', 'P', 'G', 'S', 'T', 'D', 'E', 'Q', 'N', 'K', 'R', 'H', 'W', 'Y', 'M', 'L', 'I', 'V', 'C')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
```

```
mat1<-PseKRAAC_T3A(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T3A(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T3B *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_3B)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type3 contain two type: type3A and type3B. 'PseKRAAC_T3B' contains Grp 2-20.

Usage

```
PseKRAAC_T3B(
  seqs,
  type = "gap",
  Grp = 2,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 2=c('HRKQNEDSTGPACVIM', 'LFYW'), 3=c('HRKQNEDSTGPACVIM', 'LFY', 'W'), 4=c('HRKQNEDSTGPA', 'CIV', 'MLFY', 'W'), 5=c('HRKQNEDSTGPA', 'CV', 'IML', 'FY', 'W'), 6=c('HRKQNEDSTPA', 'G', 'CV', 'IML', 'FY', 'W'), 7=c('HRKQNEDSTA', 'G', 'P', 'CV', 'IML', 'FY', 'W'), 8=c('HRKQSTA', 'NED', 'G', 'P', 'CV', 'IML', 'FY', 'W'), 9=c('HRKQ', 'NED', 'ASTG', 'P', 'C', 'IV', 'MLF', 'Y', 'W'), 10=c('RKHSA', 'Q', 'NED', 'G', 'P', 'C', 'TIV', 'MLF', 'Y', 'W'), 11=c('RKQ', 'NG', 'ED', 'AST', 'P', 'C', 'IV', 'HML', 'F', 'Y', 'W'), 12=c('RKQ', 'ED', 'NAST', 'G', 'P', 'C', 'IV', 'H', 'ML', 'F', 'Y', 'W'), 13=c('RK', 'QE', 'D', 'NG', 'HA', 'ST', 'P', 'C', 'IV', 'ML', 'F', 'Y', 'W'), 14=c('R', 'K', 'QE', 'D', 'NG', 'HA', 'ST', 'P', 'C', 'IV', 'ML', 'F', 'Y', 'W'), 15=c('R', 'K', 'QE', 'D', 'NG', 'HA', 'ST', 'P', 'C', 'IV', 'M', 'L', 'F', 'Y', 'W'), 16=c('R', 'K', 'Q', 'E', 'D', 'NG', 'HA', 'ST', 'P', 'C', 'IV', 'M', 'L', 'F', 'Y', 'W'), 17=c('R', 'K', 'Q', 'E', 'D', 'NG', 'HA', 'S', 'T', 'P', 'C', 'IV', 'M', 'L', 'F', 'Y', 'W'), 18=c('R', 'K', 'Q', 'E', 'D', 'NG', 'HA', 'S', 'T', 'P', 'C', 'I', 'V', 'M', 'L', 'F', 'Y', 'W'), 19=c('R', 'K', 'Q', 'E', 'D', 'NG', 'H', 'A', 'S', 'T', 'P', 'C', 'I', 'V', 'M', 'L', 'F', 'Y', 'W'), 20=c('R', 'K', 'Q', 'E', 'D', 'N', 'G', 'H', 'A', 'S', 'T', 'P', 'C', 'I', 'V', 'M', 'L', 'F', 'Y', 'W')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T3B(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T3B(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T4 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_4)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type4(PseKRAAC_T4) contains Grp 5,8,9,11,13,20.

Usage

```
PseKRAAC_T4(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 5=c('G', 'IVFYW', 'ALMEQRK', 'P', 'NDHSTC'), 8=c('G', 'IV', 'FYW', 'ALM', 'EQRK', 'P', 'ND', 'HSTC'), 9=c('G', 'IV', 'FYW', 'ALM', 'EQRK', 'P', 'ND', 'HS', 'TC'), 11=c('G', 'IV', 'FYW', 'A', 'LM', 'EQRK', 'P', 'ND', 'HS', 'T', 'C'), 13=c('G', 'IV', 'FYW', 'A', 'L', 'M', 'E', 'QRK', 'P', 'ND', 'HS', 'T', 'C'), 20=c('G', 'I', 'V', 'F', 'Y', 'W', 'A', 'L', 'M', 'E', 'Q', 'R', 'K', 'P', 'N', 'D', 'H', 'S', 'T', 'C')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T4(seqs=filePrs,type="gap",Grp=8,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T4(seqs=filePrs,type="lambda",Grp=8,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T5 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_5)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type5(PseKRAAC_T5) contains Grp 3,4,8,10,15,20.

Usage

```
PseKRAAC_T5(
  seqs,
  type = "gap",
  Grp = 4,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 3=c('FWYCILMVAGSTPHNQ', 'DE', 'KR'), 4=c('FWY', 'CILMV', 'AGSTP', 'EQNDHKR'), 8=c('FWY', 'CILMV', 'GA', 'ST', 'P', 'EQND', 'H', 'KR'), 10=c('G', 'FYW', 'A', 'ILMV', 'RK', 'P', 'EQND', 'H', 'ST', 'C'), 15=c('G', 'FY', 'W', 'A', 'ILMV', 'E', 'Q', 'RK', 'P', 'N', 'D', 'H', 'S', 'T', 'C'), 20=c('G', 'I', 'V', 'F', 'Y', 'W', 'A', 'L', 'M', 'E', 'Q', 'R', 'K', 'P', 'N', 'D', 'H', 'S', 'T', 'C')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T5(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T5(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T6A *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_6A)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type6 contain two type: type6A and type6B. 'PseKRAAC_T6A' contains Grp 4,5,20.

Usage

```
PseKRAAC_T6A(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 4=c('AGPST', 'CILMV', 'DEHKNQR', 'FYW'), 5=c('AHT', 'CFILMVWY', 'DE', 'GP', 'KNQRS'), 20=c('A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T6A(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T6A(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T6B *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_6B)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type6 contain two type: type6A and type6B. 'PseKRAAC_T6B' contains Grp 5.

Usage

```
PseKRAAC_T6B(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: 5=c('AEHKQRST', 'CFILMVWY', 'DN', 'G', 'P')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T6B(seqs=filePrs,type="gap",Grp=5,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T6B(seqs=filePrs,type="lambda",Grp=5,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T7 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_7)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type7(PseKRAAC_T7) contains Grp 2-20.

Usage

```
PseKRAAC_T7(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.

Grp is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.

GapOrLambdaValue
is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.

k This parameter keeps the value of k in k-mer.

label is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: Grp2=c('C', 'MFILVWYAGTSNQDEHRKP'), Grp3=c('C', 'MFILVWYAKR', 'GTSNQDEHP'), Grp4=c('C', 'KR', 'MFILVWYA', 'GTSNQDEHP'), Grp5=c('C', 'KR', 'MFILVWYA', 'DE', 'GTSNQHP'), Grp6=c('C', 'KR', 'WYA', 'MFILV', 'DE', 'GTSNQHP'), Grp7=c('C', 'KR', 'WYA', 'MFILV', 'DE', 'QH', 'GTSNP'), Grp8=c('C', 'KR', 'WYA', 'MFILV', 'D', 'E', 'QH', 'GTSNP'), Grp9=c('C', 'KR', 'WYA', 'MFILV', 'D', 'E', 'QH', 'TP', 'GSN'), Grp10=c('C', 'KR', 'WY', 'A', 'MFILV', 'D', 'E', 'QH', 'TP', 'GSN'), Grp11=c('C', 'K', 'R', 'WY', 'A', 'MFILV', 'D', 'E', 'QH', 'TP', 'GSN'), Grp12=c('C', 'K', 'R', 'WY', 'A', 'MFILV', 'D', 'E', 'QH', 'TP', 'GS', 'N'), Grp13=c('C', 'K', 'R', 'W', 'Y', 'A', 'MFILV', 'D', 'E', 'QH', 'TP', 'GS', 'N'), Grp14=c('C', 'K', 'R', 'W', 'Y', 'A', 'FILV', 'M', 'D', 'E', 'QH', 'TP', 'GS', 'N'), Grp15=c('C', 'K', 'R', 'W', 'Y', 'A', 'FILV', 'M', 'D', 'E', 'Q', 'H', 'TP', 'GS', 'N'), Grp16=c('C', 'K', 'R', 'W', 'Y', 'A', 'FILV', 'M', 'D', 'E', 'Q', 'H', 'TP', 'G', 'S', 'N'), Grp17=c('C', 'K', 'R', 'W', 'Y', 'A', 'FI', 'LV', 'M', 'D', 'E', 'Q', 'H', 'TP', 'G', 'S', 'N'), Grp18=c('C', 'K', 'R', 'W', 'Y', 'A', 'FI', 'LV', 'M', 'D', 'E', 'Q', 'H', 'T', 'P', 'G', 'S', 'N'), Grp19=c('C', 'K', 'R', 'W', 'Y', 'A', 'F', 'I', 'LV', 'M', 'D', 'E', 'Q', 'H', 'T', 'P', 'G', 'S', 'N'), Grp20=c('C', 'K', 'R', 'W', 'Y', 'A', 'F', 'I', 'L', 'V', 'M', 'D', 'E', 'Q', 'H', 'T', 'P', 'G', 'S', 'N')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T7(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T7(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T8 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_8)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type8(PseKRAAC_T8) contains Grp 2-20.

Usage

```
PseKRAAC_T8(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: Grp2=c('ADEGKNPQRST', 'CFHILMVWY'), Grp3=c('ADEGNPST', 'CHKQRW', 'FILMVY'), Grp4=c('AGNPST', 'CHWY', 'DEKQR', 'FILMV'), Grp5=c('AGPST', 'CFWY', 'DEN', 'HKQR', 'ILMV'), Grp6=c('APST', 'CW', 'DEGN', 'FHY', 'ILMV', 'KQR'), Grp7=c('AGST', 'CW', 'DEN', 'FY', 'HP', 'ILMV', 'KQR'), Grp8=c('AST', 'CG', 'DEN', 'FY', 'HP', 'ILV', 'KQR', 'MW'), Grp9=c('AST', 'CW', 'DE', 'FY', 'GN', 'HQ', 'ILV', 'KR', 'MP'), Grp10=c('AST', 'CW', 'DE',

'FY', 'GN', 'HQ', 'IV', 'KR', 'LM', 'P'), Grp11=c('AST', 'C', 'DE', 'FY', 'GN', 'HQ', 'IV', 'KR', 'LM', 'P', 'W'), Grp12=c('AST', 'C', 'DE', 'FY', 'G', 'HQ', 'IV', 'KR', 'LM', 'N', 'P', 'W'), Grp13=c('AST', 'C', 'DE', 'FY', 'G', 'H', 'IV', 'KR', 'LM', 'N', 'P', 'Q', 'W'), Grp14=c('AST', 'C', 'DE', 'FL', 'G', 'H', 'IV', 'KR', 'M', 'N', 'P', 'Q', 'W', 'Y'), Grp15=c('AST', 'C', 'DE', 'F', 'G', 'H', 'IV', 'KR', 'L', 'M', 'N', 'P', 'Q', 'W', 'Y'), Grp16=c('AT', 'C', 'DE', 'F', 'G', 'H', 'IV', 'KR', 'L', 'M', 'N', 'P', 'Q', 'S', 'W', 'Y'), Grp17=c('AT', 'C', 'DE', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'W', 'Y'), Grp18=c('A', 'C', 'DE', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y'), Grp19=c('A', 'C', 'D', 'E', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y'), Grp20=c('A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'V', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T8(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T8(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

PseKRAAC_T9 *Pseudo K_tuple Reduced Amino Acid Composition(PseKRAAComposition Type_9)*

Description

There are 16 types of PseKRAAC function. In the functions, a (user-selected) grouping of the amino acids might be used to reduce the amino acid alphabet. Also, the functions have a type parameter. The parameter determines the protein sequence analyses which can be either gap or lambda-correlation. PseKRAAC_type9(PseKRAAC_T9) contains Grp 2-20.

Usage

```
PseKRAAC_T9(
  seqs,
  type = "gap",
  Grp = 5,
  GapOrLambdaValue = 2,
```

```
  k = 4,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
type	This parameter has two valid value "lambda" and "gap". "lambda" calls lambda_model function and "gap" calls gap_model function.
Grp	is a numeric value. It shows the id of an amino acid group. Please find the available groups in the detail section.
GapOrLambdaValue	is an integer. If type is gap, this value shows number of gaps between two k-mers. If type is lambda, the value of GapOrLambdaValue shows the number of gaps between each two amino acids of k-mers.
k	This parameter keeps the value of k in k-mer.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Groups: Grp2=c('ADEGKNPQRST', 'CFHILMVWY'), Grp3=c('ADEGNPST', 'CHKQRW', 'FILMVY'), Grp4=c('AGNPST', 'CHWY', 'DEKQR', 'FILMV'), Grp5=c('AGPST', 'CFWY', 'DEN', 'HKQR', 'ILMV'), Grp6=c('APST', 'CW', 'DEGN', 'FHY', 'ILMV', 'KQR'), Grp7=c('AGST', 'CW', 'DEN', 'FY', 'HP', 'ILMV', 'KQR'), Grp8=c('AST', 'CG', 'DEN', 'FY', 'HP', 'ILV', 'KQR', 'MW'), Grp9=c('AST', 'CW', 'DE', 'FY', 'GN', 'HQ', 'ILV', 'KR', 'MP'), Grp10=c('AST', 'CW', 'DE', 'FY', 'GN', 'HQ', 'IV', 'KR', 'LM', 'P'), Grp11=c('AST', 'C', 'DE', 'FY', 'GN', 'HQ', 'IV', 'KR', 'LM', 'P', 'W'), Grp12=c('AST', 'C', 'DE', 'FY', 'G', 'HQ', 'IV', 'KR', 'LM', 'N', 'P', 'W'), Grp13=c('AST', 'C', 'DE', 'FY', 'G', 'H', 'IV', 'KR', 'LM', 'N', 'P', 'Q', 'W'), Grp14=c('AST', 'C', 'DE', 'FL', 'G', 'H', 'IV', 'KR', 'M', 'N', 'P', 'Q', 'W', 'Y'), Grp15=c('AST', 'C', 'DE', 'F', 'G', 'H', 'IV', 'KR', 'L', 'M', 'N', 'P', 'Q', 'W', 'Y'), Grp16=c('AT', 'C', 'DE', 'F', 'G', 'H', 'IV', 'KR', 'L', 'M', 'N', 'P', 'Q', 'S', 'W', 'Y'), Grp17=c('AT', 'C', 'DE', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'W', 'Y'), Grp18=c('A', 'C', 'DE', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y'), Grp19=c('A', 'C', 'D', 'E', 'F', 'G', 'H', 'IV', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y'), Grp20=c('A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'V', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'W', 'Y')

Value

This function returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (Grp)^k.

References

Zuo, Yongchun, et al. "PseKRAAC: a flexible web server for generating pseudo K-tuple reduced amino acids composition." Bioinformatics 33.1 (2017): 122-124.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat1<-PseKRAAC_T9(seqs=filePrs,type="gap",Grp=4,GapOrLambdaValue=3,k=2)

mat2<-PseKRAAC_T9(seqs=filePrs,type="lambda",Grp=4,GapOrLambdaValue=3,k=2)
```

QSOrder *Quasi Sequence Order*

Description

This function computes the quasi-sequence-order for sequences. It is for amino acid pairs with d distances (d can be any number between 1 and 20). First, it calculates the frequencies of each amino acid ("A", "C",..., "Y"). Then, it normalizes the frequencies by dividing the frequency of an amino acid to the frequency of all amino acids plus the sum of tau values which is multiplied by W. tau values are given by function SOCNumber. For d bigger than 20, it computes tau for d in the range "1 to (nlag-20) * W" and normalizes them like before.

Usage

```
QSOrder(seqs, nlag = 25, W = 0.1, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
nlag	is a numeric value which shows the maximum distance between two amino acids. Distances can be 1, 2, ..., or nlag.
W	(weight) is a tuning parameter.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Details

Please find details about tau in function SOCNumber.

Value

It returns a feature matrix which the number of rows equals to the number of sequences and the number of columns is (nlag*2). For each distance d, there are two values. One value for Granthman and another one for Schneider distance.

Note

For d between 21 to nlag, the function calculates tau values for (d-20) to (nlag-20).

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat<-QSOrder(seqs=filePrs,nlag=25)
```

---

revComp *reverseCompelement*

Description

This function returns the reverse compelement of a dna sequence.

Usage

```
revComp(seq, outputType = "char")
```

Arguments

seq	is a dna sequence.
outputType	this parameter can take two values: 'char' or 'str'. If outputType is 'str', the reverse complement sequence of the input sequence is returned as a string. Otherwise, a vector of characters which represent the reverse complement is returned. Default value is 'char'.

Value

The reverse complement of the input sequence.

Examples

```
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
Seq<-ptmSeqsVect[1]
revCompSeq<-revComp(seq=Seq,outputType="char")
```

SAAC *Splitted Amino Acid Composition(SAAC)*

Description

This function splits the input sequence into three parts. The first part is N-terminal and the third part is C-terminal and middle part contains all amino acids between these two part. N-terminal will be determined by the first numNterm amino acid in the sequences and C-terminal is determined by numCterm of the last amino acids in the sequence. Users should enter numNterm and numCterm parameters. Their default value is 25. The function calculates kAAComposition for each of the three parts.

Usage

```
SAAC(seqs, k = 1, numNterm = 5, numCterm = 5, normalized = TRUE, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	shows which type of amino acid composition applies to the parts. For example, the amino acid composition is applied when k=1 and when k=2, the dipeptide Composition is applied.
numNterm	shows how many amino acids should be considered for N-terminal.
numCterm	shows how many amino acids should be considered for C-terminal.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

It returns a feature matrix. The number of rows is equal to the number of sequences. The number of columns is (3*(20^k)).

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-SAAC(seqs=filePrs,k=1,numNterm=15,numCterm=15)
```

SGAAC *Splitted Group Amino Acid Composition(SGAAC)*

Description

In this function, amino acids are first grouped into a user-defined category. Later, the splitted amino Acid composition is computed. Please note that this function differs from SAAC which works on individual amino acids.

Usage

```
SGAAC(
  seqs,
  k = 1,
  numNterm = 25,
  numCterm = 25,
  Grp = "locFus",
  normalized = TRUE,
  label = c()
)
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
k	shows which type of amino acid composition applies to the parts. For example, the amino acid composition is applied when k=1 and when k=2, the dipeptide Composition is applied.
numNterm	shows how many amino acids should be considered for N-terminal.
numCterm	shows how many amino acids should be considered for C-terminal.
Grp	is a list of vectors containig amino acids. Each vector represents a category. Users can define a customized amino acid grouping, provided that the sum of all amino acids is 20 and there is no repeated amino acid in the groups. Also, users can choose 'cTriad'(conjointTriad), 'locFus', or 'aromatic'. Each option provides specific information about the type of an amino acid grouping.
normalized	is a logical parameter. When it is FALSE, the return value of the function does not change. Otherwise, the return value is normalized using the length of the sequence.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

It returns a feature matrix. The number of rows is equal to the number of sequences. The number of columns is 3*((number of groups)^k).

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")
mat<-SGAAC(seqs=filePrs,k=1,numNterm=15,numCterm=15,Grp="aromatic")
```

SOCNumber *Sequence Order Coupling Number*

Description

This function uses dissimilarity matrices Grantham and Schneider to compute the dissimilarity between amino acid pairs. The distance between amino acid pairs is determined by d which varies between 1 to nlag. For each d, it computes the sum of the dissimilarities of all amino acid pairs. The sum shows the value of tau for a value d. The feature vector contains the values of taus for both matrices. Thus, the length of the feature vector is equal to nlag*2.

Usage

```
SOCNumber(seqs, nlag = 30, label = c())
```

Arguments

seqs	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
nlag	is a numeric value which shows the maximum distance between two amino acids. Distances can be 1, 2, ..., or nlag. Defult is 30.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).

Value

It returns a feature matrix. The number of rows is equal to the number of sequences and the number of columns is (nlag*2). For each distance d, there are two values. One value for Granthman and another one for Schneider distance.

Note

When d=1, the pairs of amino acids have no gap and when d=2, there is one gap between the amino acid pairs in the sequence. It will repeat likewise for other values of d.

Examples

```
filePrs<-system.file("extdata/proteins.fasta",package="ftrCOOL")

mat<-SOCNumber(seqs=filePrs,nlag=25)
```

TriNucIndex *Tri Nucleotide Index (TriNucIndex)*

Description

This function replaces trinucleotides in a sequence with their physicochemical properties in the trinucleotide index file.

Usage

```
TriNucIndex(
  seqs,
  selectedNucIdx = 1:12,
  standardized = TRUE,
  threshold = 0.8,
  label = c(),
  outFormat = "mat",
  outputFileDist = ""
)
```

Arguments

seqs	is a FASTA file containing nucleotide sequences. The sequences start with '>'. Also, seqs could be a string vector. Each element of the vector is a nucleotide sequence.
selectedNucIdx	TriNucIndex function works based on physicochemical properties. Users, select the properties by their ids or indexes in trinucleotide file.
standardized	is a logical parameter. If it is set to TRUE, all the indices will be in their standard format. The default value is TRUE.
threshold	is a number between 0 to 1. In selectedNucIdx, indices with a correlation higher than the threshold will be deleted. The default value is 0.8.
label	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
outFormat	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
outputFileDist	shows the path and name of the 'txt' output file.

Details

There are 12 physicochemical indexes in the trinucleotide database.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length)*(number of selected trinucleotide properties) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
fileLNC<-system.file("extdata/Athaliana1.fa",package="ftrCOOL")
vect<-TriNucIndex(seqs = fileLNC, selectedNucIdx=1:5,threshold=1,outFormat="mat")
```

---

zSCALE *Z-SCALE*

Description

This function converts the amino acids of a sequence to five physicochemical descriptor variables which were developed by Sandberg et al. in 1998. The Z-SCALE function can be applied to encode peptides of equal length.

Usage

```
zSCALE(seqs, label = c(), outFormat = "mat", outputFileDist = "")
```

Arguments

`seqs`	is a FASTA file with amino acid sequences. Each sequence starts with a '>' character. Also, seqs could be a string vector. Each element of the vector is a peptide/protein sequence.
`label`	is an optional parameter. It is a vector whose length is equivalent to the number of sequences. It shows the class of each entry (i.e., sequence).
`outFormat`	(output format) can take two values: 'mat'(matrix) and 'txt'. The default value is 'mat'.
`outputFileDist`	shows the path and name of the 'txt' output file.

Value

The output depends on the outFormat parameter which can be either 'mat' or 'txt'. If outFormat is 'mat', the function returns a feature matrix for sequences with the same length such that the number of columns is (sequence length)*(5) and the number of rows is equal to the number of sequences. It is usable for machine learning purposes. If the outFormat is 'txt', the output is written to a tab-delimited file.

Note

This function is provided for sequences with the same lengths. Users can use 'txt' option in outFormat parameter for sequences with different lengths. Warning: If outFormat is set to 'mat' for sequences with different lengths, it returns an error. Also, when output format is 'txt', label information is not shown in the text file. It is noteworthy that 'txt' format is not usable for machine learning purposes.

Examples

```
ptmSeqsADR<-system.file("extdata/",package="ftrCOOL")
ptmSeqsVect<-as.vector(read.csv(paste0(ptmSeqsADR,"/ptmVect101AA.csv"))[,2])
mat<-zSCALE(seqs = ptmSeqsVect,outFormat="mat")
```

Index