Package 'kodonz'

February 10, 2019

Title A Comprehensive Codon Analysis Toolbox

Version 0.1.0

Description Codon analysis toolbox

Depends R (>= 3.4.3)

License GPL (==2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Imports seqinr

R topics documented:

arg_check *Arguments checks*

Description

Performs the usual checks of arguments consistency of codon sequences and codon table to be used

Usage

```
arg_check(x, mode, y)
```

Arguments

x	a char array of codons
mode	a char array denoting the expected mode of the argument x
y	a number denoting the codon table to be used. 0 = standard codon table, 2 = vertibrate mitchodrial, 3 = yeast mitochondrial, 4 = mold, protozoan, and Coelenterate mitochondrial and Mycoplasma/Spiroplasma, 5 = invertebrate mitochandrial, 6 = Ciliate, Dasycladacean and Hexamita nuclear, 9 = Echinoderm and flatworm mitochondrial, 10 = Euplotid nuclear, 11 = bacterial, Archaeal and plant plastid, 12 = alternative yeast nuclear, 13 = Ascidian mitochondrial, 14 = alternative flatworm mitochondrial, 16 = Chlorophycean mitochondrial, 21 = Trematode mitochondrial, 22 = Scenedesmus obliquus mitochondrial, 23 = Thraustochytrium mitochondrial, 24 = Pterobranchia mitochondrial, 25 = candidate division SR1 and Gracilibacteria, 26 = Pachysolen tannophilus nuclear, 29 = Mesodinium nuclear, 30 = Peritrich nuclear.

Value

a numeric matrix of the appropriate codon table

augc_freq *Overall AT and GC content*

Description

Calculate the overall AT and GC content

Usage

```
augc_freq(x)
```

Arguments

x	a list of KZsqns objects

Value

a numeric matrix of AU and GC content

base_freq *Overall nucleotides frequency*

Description

Calculate the overall nucleotides frequency composition

Usage

```
base_freq(x)
```

Arguments

x	a list of KZsqns objects.

Value

a n-by-4 numeric matrix, where n is the length of list x

Examples

```
data(CodonTable0)
x = vector('list', 5) # Creating an empty list of length 5

for(i in 1:5){
x[[i]] = CodonTable0[sample(1:64, 10*i, TRUE),1]
attr(x[[i]], 'class') = 'KZsqns'
}

cat(base_freq(x))
```

cai *Codon adaptation index*

Description

Calculates the codon adaptation index (CAI) for each codon

Usage

```
cai(x, ref_spp = "n", ref, y = 0, threthold = 0.005,
  lower_bound = 0.005)
```

Arguments

x	a KZsqns object, for which CAI will be calculated
ref_spp	a char sequence of species name to use. "ecoli" = *Escherichia coli*; "bsubt" = *Bacillus subtilis*; "scere" = *Saccharomyces cerevisiae*; When specified, it overides the value set by *ref*
ref	a named numeric array of relative adaptiveness (0 to 1) of each codon from a species-specific reference set of very highly expressed genes. The array should have a length of 64 with 0 filled in at the stop codons. The name of each number is the corresponding char array of codon. For example, "AAA" or "Aaa" represent the same codon. Upper case letters can be mixed with lower case letters.
y	an optional number denoting the codon table to be used. 0 = standard codon table (default), 2 = vertibrate mitchodrial (See dna2aa for additional options). When an option other than those mentioned above is provided, the standard codon table will be used
threthold	a number below which the adaptiveness will be forced to *lower_bound*
lower_bound	a number to which values of adaptiveness below the *threshold* will be forced

Details

When any reference codon has a relative usage value of zero and the codon sequence to be calculated has such a codon, the CAI for this sequence would be zeroed out. *lower_bound* and *threthold* are there to prevent such an undesired result. There is a helper method *cai_crt* to calculate the reference table from a given reference set of highly expressed genes. It is the intention of the authors to keep this helper method seperate from the CAI method, and the user can device their own rules about how to construct a reference table by creating their own reference table or helper method.

Value

a number denoting the CAI for the provided sequence

Source

The reference set of very highly expressed genes for *E. coli*, *B. subtilis* and *S. cerevisiae* [1][2].

References

[1] Sharp, P. M., & Li, W.-H. (1986). Codon usage in regulatory genes in Escherichia coli does not reflect selection for 'rare'codons. Nucleic acids research, 14(19), 7737-7749. [2] Shields, D. C., & Sharp, P. M. (1987). Synonymous codon usage in Bacillus subtilis reflects both translational selection and mutational biases. Nucleic acids research, 15(19), 8023-8040.

---

cai_crt *Custom CAI reference table*

Description

A helper method to calculate the codon adaptiveness index reference table based on a list of highly expressed genes

Usage

```
cai_crt(x, y, method = c("sharp", "bulmer"))
```

Arguments

x	a list of KZsqns object, from which the reference table will be calculated
y	y an optional number denoting the codon table to be used. 0 = standard codon table (default), 2 = vertibrate mitchodrial (See dna2aa for additional options). When an option other than those mentioned above is provided, the standard codon table will be used
method	a charactor string denoting which method to use for codons absent from the reference set: "bulmer", set the adaptiveness of the codon to 0.01; "sharp", in calculating the adaptiveness of an absent codon, use a frequency of 0.5 instead

Details

If any codon is missing from the reference set, the adaptiveness value of that codon need to be adjusted in order to obtain valid CAI for sequences that have that codon in their sequence. The sharp's method is the default option [1], but the user may choose to use bulmer's method [2].

References

[1] Sharp, P. M., & Li, W.-H. (1987). The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. Nucleic acids research, 15(3), 1281-1295. [2] Bulmer, M. (1988). Are codon usage patterns in unicellular organisms determined by selection-mutation balance? Journal of Evolutionary Biology, 1(1), 15-26.

---

comp_ly *Composition analysis*

Description

Calculates nucleotide composition of a list of DNA sequenses

Usage

```
comp_ly(x)
```

---

dna2aa *Translating a codon sequence to into animo-acids*

Description

This function translate an array of codons into the corresponding amino acid sequence using the codon table of choice.

Usage

```
dna2aa(dnaSequence, aaNoLetters = 3, codonTable = 0)
```

Arguments

dnaSequence	a character array of codons
aaNoLetters	a number to assign the format of animo acid. 1 = one-letter abbreviation, 3 = three-letter abbreviation. If it defaults into the 3-letter abbreviation when no number or a different number is specified
codonTable	a number to assign the type of codon table to be used for the translation. 0 = standard codon table, 2 = vertibrate mitchodrial, 3 = yeast mitochondrial, 4 = mold, protozoan, and Coelenterate mitochondrial and Mycoplasma/Spiroplasma, 5 = invertebrate mitochondrial, 6 = Ciliate, Dasycladacean and Hexamita nuclear, 9 = Echinoderm and flatworm mitochondrial, 10 = Euplotid nuclear, 11 = bacterial, Archaeal and plant plastid, 12 = alternative yeast nuclear, 13 = Ascidian mitochondrial, 14 = alternative flatworm mitochondrial, 16 = Chlorophycean mitochondrial, 21 = Trematode mitochondrial, 22 = Scenedesmus obliquus mitochondrial, 23 = Thraustochytrium mitochondrial, 24 = Pterobranchia mitochondrial, 25 = candidate division SR1 and Gracilibacteria, 26 = Pachysolen tannophilus nuclear, 29 = Mesodinium nuclear, 30 = Peritrich nuclear. If no number is specified or an option other than those provided above is specified, there will be a warning and the standard codon table will be used

Details

Codon tables were based on the list compiled by Andrzej Elzanowski and Jim Ostell [1] at NCBI, Bethesda, Maryland, USA. The numbering of codon tables respects the order in Elzanowski and Ostell [1] (except the standard code, which was numbered 0 in this package). There are three adddditional codon tables listed in Elzanowski and Ostell [1] (27 Karyorelict nuclear, 28 Condylostoma nuclear and 31 Blastocrithidia nuclear), but were excluded in this package, because these codon tables include codons that can be either intepreted as a AA coding codon or stop codon, and the AA sequence can not be determined based on the DNA sequence alone. The codon table data were available as "CodonTable#", where "#" denote the numeric order of the codon table. For example, "CodonTable21" is the Trematode mitochonrial code table.

Value

a character array of animo-acids

References

[1] https://www.ncbi.nlm.nih.gov/Taxonomy/Utils/wprintgc.cgi#SG13 [retrieved on 2/4/2019]

Examples

```
data('CodonTable0')
sqs = sample(1:64, 100, replace=T)
codonSequence = CodonTable0[sqs,1] # an arbitrary sequence of codons is genereated based on the standard codon t
cat("The sequence of aa based on the standard codon table:\n")
cat(CodonTable0[sqs,2])
cat("The translated aa sequence:\n")
cat(dna2aa(codonSequence, 1, 0))
```

enc *Effective number of codons*

Description

This function calculate the effective number of codons (ENC) using Wright's formula.

Usage

```
enc(x, y = 0, formula = "w")
```

Arguments

x	a KZsqns object
y	a number to assign the type of codon table to be used for the translation. 0 = standard codon table, 2 = vertibrate mitchodrial (See dna2aa for additional options). If no number is specified or an option other than those provided above is specified, there will be a warning and the standard codon table will be used
formula	a char array to denote the formula for calculating ENC. Options include: 'w': Wright's formula, 'f4-': Fuglsang's Ncf4- formula.

Value

an numeric array of the ENCs

References

Fuglsang, A., 2006 Estimating the "Effective Number of Codons": The Wright way of determining codon homozygosity leads to superior estimates. Genetics 172:1301-1307 Wright, F., 1990 The 'effective number of codons' used in a gene. Gene 87: 23-29.

Examples

```
dnalist = vector('list', 2)
dnalist[[1]] = CodonTable2[sample(1:64, 1e4, TRUE),1]
attr(dnalist[[1]], 'class') <- 'KZsqns'
dnalist[[2]] = CodonTable2[sample(1:64, 1e4, TRUE),1]
attr(dnalist[[2]], 'class') <- 'KZsqns'
enc(x = dnalist, y = 2) # Two numbers should be around 60 as there are 60 animo acid coding codons and 4 stop codon

dnalist = vector('list', 2)
dnalist[[1]] = CodonTable2[sample(1:64, 1e4, TRUE),1]
attr(dnalist[[1]], 'class') <- 'KZsqns'
dnalist[[2]] = CodonTable2[sample(1:64, 1e4, TRUE),1]
attr(dnalist[[2]], 'class') <- 'KZsqns'
enc(x = dnalist, y = 4) # Two numbers should be around 62 as there are 60 animo acid coding codons and 2 stop codon
```

enc_exp *Expected number of ENC given GC3 content*

Description

Calculates the expected ENC based on Wright's formula given GC3 percentage

Usage

```
enc_exp(gc3 = 0)
```

Arguments

gc3	a number of GC3 content

Value

a number of the expected ENC given the GC3 content

gc3 *Calculate GC3 value of a codon sequence*

Description

Outputs an array of numbers, representing the proportion of codons with G/C bases in the third position

Usage

```
gc3(x)
```

Arguments

x	a list of KZsqns objects.

Value

a number array of GC3s.

Examples

```
data('CodonTable0')
x = vector('list', 5) # Creating an empty list of length 5
for(i in 1:5){
  x[[i]] = CodonTable0[sample(1:64, 10*i, TRUE),1]
  attr(x[[i]], 'class') = 'KZsqns'
}

cat(gc3(x))
```

gcx *GCx of a list of genes*

Description

Calculates the GC1,2,3 and GC12 contents of a list of genes

Usage

```
gcx(x, pos = 1)
```

Arguments

x	a list of KZsqns objects.
pos	the position of the codon for which GCx is calculated. 1: GC1; 2:GC2; 3: GC3; 4:GC12

Value

a matrix of the GCx of the list of genes

load.fasta *A function to load fasta files*

Description

Loads the specified fasta file

Usage

```
load.fasta(file = "n", x, ...)
```

Arguments

file	Name of the fasta file
x	a character array of DNA sequence. Use "A", "T", "G" and "C" to denote the nucleatide composition. When a fasta file is also specified, the fasta file takes priority, and this character sequence is ignored
...	additional parameters you might want to pass to the seqinr::read.fasta function

Value

A list of objects of class KZsqns

Examples

```
dna1 = load.fasta("mitochondrial01.fasta", as.string = TRUE) # not run
```

n3_freq *Nucleotide at the third position of the codon*

Description

Calculates the nucleotide frequency at the third position of the codon

Usage

```
n3_freq(x)
```

Arguments

x	a list of KZsqns objects.

Value

a matrix of nucleotide composition at the third position

nc_fuglsang4m *Effective number of codons based on Fuglsang's formula 4-*

Description

Effective number of codons based on Fuglsang's formula 4-

Usage

```
nc_fuglsang4m(x, cTable)
```

nc_wright *Effective number of codons based on Wright's formula*

Description

Effective number of codons based on Wright's formula

Usage

```
nc_wright(x, cTable)
```

pminmax *%MinMax of a coding DNA sequence*

Description

Calculates the %MinMax (Clarke and Clark, 2008) of a DNA sequence coding a protein of interest.

Usage

```
pminmax(x, z, cut, y = 0, spp = "")
```

Arguments

x	a char array of the codons for which %MinMax will be calculated
z	the width of the sliding window.
cut	a named numeric array of the codon usage table of the host species. Each entry is the number of codons of genes from the reference species, and the name is the case insensitive three letter base char string of the codon.
y	a number denoting the codon table to be used. When *cut* is specified, y also needs to be specified. 0 = standard codon table (default), 2 = vertibrate mitchodrial (See dna2aa for additional options). When an option other than those mentioned above is provided, the standard codon table will be used
spp	a character string denoting the host species name on which the codon usage table is based. Sixteen build-in reference tables are available: 'ecoli', E. coli (standard code); 'insect', insect (standard code); 'yeast', yeast (standard code); 'celegans', C. elegans (standard code); 'drosophila', Drosophila melanogaster (standard code); 'human', human (standard code); 'mouse', mouse (standard code); 'rat', rat (standard code); 'pig', pig (standard code); 'pichia', Pichia pastoris (standard code); 'arabidopsis', Arabidopsis thaliana (standard code); 'streptomyces', Streptomyces (standard code); 'zea', Zea mays (standard code); 'nicotiana', Nicotiana tabacum (standard code); 'saccharomyces', Saccharomyces cerevisiae (standard code); 'cricetulus', Cricetulus griseus (standard code). When specified, this option overides cut.

Details

Frequently used codon frequency table in 16 expression host organisms were built-in based on Genscript [1]. In addition, when your host organism is not one of those built-in, you can supply a custom array of codon usage table and the corresponding codon table can be supplied using cut and y to calculate %MinMax statistic for additional host organisms.

References

[1] https://www.genscript.com/tools/codon-frequency-table [retreaved 2/4/2019]

rm_ndc *Remove non-degenerate and stop codons*

Description

Removes non-redundant and stop codons based on the specified codon table. For some compositional analysis, it is advisable to remove non-redundant amino acid coding codons and stop codons.

Usage

```
rm_ndc(x, y = 0, keep.stop = TRUE)
```

Arguments

x	a list of KZsqns objects
y	a vector of numbers denoting the codon table to be used for each KZsqns object. 0 = standard codon table (default), 2 = vertibrate mitchodrial (See dna2aa for additional options). When an option other than those mentioned above is provided, the standard codon table will be used. When the length of this vector is smaller than the number of KZsqns objects supplied, standard codon tables will be used for those remaining KZsqns objects
keep.stop	an option indication whether stop codons should also be kept. TRUE: keep stop codons if they are degenerate (default); FALSE: remove stop codons

rrt *Random reverse translation*

Description

Randomly reverse translate an animo acid sequence into a DNA sequence according to a codon usage table and a codon table

Usage

```
rrt(x, cut, y = 0, spp)
```

Arguments

x	a char array of animo acids (e.g., either 'G' or 'Gly' is acceptable but do not mix one letter symbols with three letter symbols in the same array) or codons (e.g., 'GGG'). The first element of the array will be used to determine the type of the input
cut	a named numerical array of codon usage table. Case-insensitive three letter codon names are used. Numbers could be the actuall codon counts or scaled ratios
y	a number denoting the codon table to be used. When *cut* is specified, *y* also needs to be specified. 0 = standard codon table (default), 2 = vertibrate mitchodrial, 3 = ?, 4 = ?. When an option other than those mentioned above is provided, the standard codon table will be used

spp — a char string of the host species name for which the codon usage table will be used. Four build-in reference tables are available: 'ecoli', E. coli; 'insect', insect; 'yeast', yeast; 'celegans', C. elegans. When specified, this option overides *cut*. These build-in codon usage tables are taken from https://www.genscript.com/tools/codon-frequency-table.

Value

a char array of random reverse translated codons

---

rscu *Relative synonymous codon usage*

Description

Calculates the relative synomynous codon usage (RSCU) for each codon.

Usage

```
rscu(x, y = 0, stop = FALSE)
```

Arguments

x — a char array of codons

y — an optional number denoting the codon table to be used. 0 = standard codon table (default), 2 = vertibrate mitchodrial (See dna2aa for additional options). When an option other than those mentioned above is provided, the standard codon table will be used

stop — a optional boolean denoting whether the RSCU values for the stop codons should be returned. FALSE: NAs will be returned (default); TRUE: the RSCUs will be returned

Details

The RSCU is calculated for each animo acid coding codon present in the provided codon sequence. When any animo acid is missing from the sequence, each coding codon for that animo acid will have NA returned as their RSCU. The RSCU of stop codons also have NAs in their position as the default setting. To include RSCUs for the stop codons in the returned array, set *stop* as TRUE.

Value

a numeric array of RSCUs for each animo acid

scuo *Synonymous codon usage order*

Description

Calculates the synonymous codon usage order of a codon sequence based on a codon table.

Usage

```
scuo(x, y = 0)
```

Arguments

x	a char sequence of codons
y	a number to assign the type of codon table to be used for the translation. 0 = standard codon table, 2 = vertibrate mitchodrial, . If no number is specified or an option other than those provided above is specified, there will be a warning and the standard codon table will be used

Index