Package ‘cubar’

January 9, 2024

**Title**  Codon Usage Bias Analysis

**Version**  0.5.0

**Description**  A suite of functions for rapid and flexible analysis of codon usage bias. It provides in-depth analysis at the codon level, including relative synonymous codon usage (RSCU), tRNA weight calculations, machine learning predictions for optimal or preferred codons, and visualization of codon-anticodon pairing. Additionally, it can calculate various gene-specific codon indices such as codon adaptation index (CAI), effective number of codons (ENC), fraction of optimal codons (Fop), tRNA adaptation index (tAI), mean codon stabilization coefficients (CSCg), and GC contents (GC/GC3s/GC4d). It also supports both standard and non-standard genetic code tables found in NCBI, as well as custom genetic code tables.

**License**  MIT + file LICENSE


**BugReports**  https://github.com/mt1022/cubar/issues

**Encoding**  UTF-8

**LazyData**  true

**LazyDataCompression**  bzip2

**Imports**  Biostrings (>= 2.60.0), IRanges (>= 2.34.0), data.table (>= 1.14.0), ggplot2 (>= 3.3.5), rlang (>= 0.4.11)

**Depends**  R (>= 4.1.0)

**Suggests**  knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder**  knitr

**RoxygenNote**  7.2.3

**Config/testthat/edition**  3

**NeedsCompilation**  no

**Author**  Hong Zhang [aut, cre, cph] (<https://orcid.org/0000-0002-4064-9432>)

**Maintainer**  Hong Zhang <mt1022.dev@gmail.com>

**Repository**  CRAN

**Date/Publication**  2024-01-09 04:00:02 UTC
**R topics documented:**

- `aa2codon` .................................................. 2
- `check_cds` .................................................. 3
- `count_codons` .............................................. 4
- `create_codon_table` ...................................... 5
- `est_csc` ..................................................... 5
- `est_optimal_codons` ...................................... 6
- `est_rscu` .................................................... 7
- `est_trna_weight` .......................................... 8
- `get_cai` ..................................................... 9
- `get_codon_table` ........................................... 10
- `get_cscg` ................................................... 10
- `get_enc` .................................................... 11
- `get_fop` ...................................................... 12
- `get_gc` ...................................................... 12
- `get_gc3s` .................................................... 13
- `get_gc4d` .................................................... 14
- `get_tai` ...................................................... 14
- `human_mt` ................................................... 15
- `plot_ca_pairing` .......................................... 16
- `rev_comp` ................................................... 16
- `seq_to_codons` ............................................ 17
- `show_codon_tables` ....................................... 17
- `yeast_cds` .................................................. 18
- `yeast_exp` .................................................. 18
- `yeast_half_life` .......................................... 19
- `yeast_trna_gcn` .......................................... 20

**Index**

<table>
<thead>
<tr>
<th>aa2codon</th>
<th>amino acids to codons</th>
</tr>
</thead>
</table>

**Description**

A data.frame of mapping from amino acids to codons

**Usage**

`aa2codon`

**Format**

A data.frame with two columns: `amino_acid` and `codon`.

- **amino_acid** amino acid corresponding to the codon
- **codon** codon identity
check_cds

Source

It is actually the standard genetic code.

Examples

aa2codon

---

check_cds | Quality control of CDS

Description

check_cds performs quality control of CDS sequences by filtering some peculiar sequences and optionally remove start or stop codons.

Usage

```r
check_cds(
  seqs,
  codon_table = get_codon_table(),
  min_len = 6,
  check_len = TRUE,
  check_start = TRUE,
  check_stop = TRUE,
  check_istop = TRUE,
  rm_start = TRUE,
  rm_stop = TRUE,
  start_codons = c("ATG")
)
```

Arguments

- `seqs`: input CDS sequences
- `codon_table`: codon table matching the genetic code of `seqs`
- `min_len`: minimum CDS length in nt
- `check_len`: check whether CDS length is divisible by 3
- `check_start`: check whether CDSs have start codons
- `check_stop`: check whether CDSs have stop codons
- `check_istop`: check internal stop codons
- `rm_start`: whether to remove start codons
- `rm_stop`: whether to remove stop codons
- `start_codons`: vector of start codons
count_codons

Value

DNAStringSet of filtered (and trimmed) CDS sequences

Examples

# CDS sequence QC for a sample of yeast genes
s <- head(yeast_cds, 10)
print(s)
check_cds(s)

count_codons  Count occurrences of different codons

Description

count_codons tabulates the occurrences of all the 64 codons in input CDSs

Usage

count_codons(seqs, ...)

Arguments

seqs  CDS sequences, DNAStringSet.
...
  additional arguments passed to ‘Biostrings::trinucleotideFrequency’.

Value

  matrix of codon (column) frequencies of each CDS (row).

Examples

# count codon occurrences
 cf_all <- count_codons(yeast_cds)
 dim(cf_all)
 cf_all[1:5, 1:5]
 count_codons(yeast_cds[1])
create_codon_table

create custom codon table from a data frame

description
create_codon_table creates codon table from data frame of aa to codon mapping.

usage
create_codon_table(aa2codon)

arguments
aa2codon a data frame with two columns: amino_acid (Ala, Arg, etc.) and codon.

value
a ‘data.table’ with four columns: aa_code, amino_acid, codon, and subfam.

examples
head(aa2codon)
create_codon_table(aa2codon = aa2codon)

est_csc

Estimate Codon Stabilization Coefficient

description
get_csc calculate codon occurrence to mRNA stability correlation coefficients (Default to Pearson’s).

usage
est_csc(
  seqs,
  half_life,
  codon_table = get_codon_table(),
  cor_method = "pearson"
)

arguments
seqs CDS sequences of all protein-coding genes. One for each gene.
half_life data.frame of mRNA half life (gene_id & half_life are column names).
codon_table a table of genetic code derived from ‘get_codon_table’ or ‘create_codon_table’.
cor_method method name passed to ‘cor.test’ used for calculating correlation coefficients.
Value
data.table of optimal codons.

References

Examples

# estimate yeast mRNA CSC
est_csc(yeast_cds, yeast_half_life)

codons_opt <- est_optimal_codons(yeast_cds)
codons_opt <- codons_opt[qvalue < 0.001 & coef < 0]
codons_opt
**est_rscu**

**Estimate RSCU**

**Description**

`est_rscu` returns the RSCU value of codons

**Usage**

```r
est_rscu(cf, weight = 1, pseudo_cnt = 1, codon_table = get_codon_table())
```

**Arguments**

- `cf`: matrix of codon frequencies as calculated by `count_codons()`.
- `weight`: a vector of the same length as `seqs` that gives different weights to CDSs when count codons. For example, it could be gene expression levels.
- `pseudo_cnt`: pseudo count to avoid dividing by zero. This may occur when only a few sequences are available for RSCU calculation.
- `codon_table`: a table of genetic code derived from `get_codon_table` or `create_codon_table`.

**Value**

a data.table of codon info and RSCU values

**References**


**Examples**

```r
# compute RSCU of all yeast genes
cf_all <- count_codons(yeast_cds)
est_rscu(cf_all)

# compute RSCU of highly expressed (top 500) yeast genes
heg <- head(yeast_seq[order(-yeast_seq$fpkm), ], n = 500)
cf_heg <- count_codons(yeast_cds[heg$gene_id])
est_rscu(cf_heg)
```
est_trna_weight  Estimate tRNA weight \( w \)

Description

est_trna_weight compute the tRNA weight per codon for TAI calculation. This weight reflects relative tRNA availability for each codon.

Usage

```r
est_trna_weight(
  trna_level,
  codon_table = get_codon_table(),
  s = list(WC = 0, IU = 0, IC = 0.4659, IA = 0.9075, GU = 0.7861, UG = 0.6295)
)
```

Arguments

- **trna_level**: named vector of tRNA level (or gene copy numbers), one value for each anti-codon. vector names are anticodons.
- **codon_table**: a table of genetic code derived from ‘get_codon_table’ or ‘create_codon_table’.
- **s**: list of non-Waston-Crick pairing penalty.

Value

data.table of tRNA expression information.

References


Examples

```r
# estimate codon tRNA weight for yeasts
est_trna_weight(yeast_trna_gcn)
```
Description

`get_cai` calculates Codon Adaptation Index (CAI) of each input CDS

Usage

`get_cai(cf, rscu)`

Arguments

- `cf`: matrix of codon frequencies as calculated by `count_codons()`.
- `rscu`: rscu table containing CAI weight for each codon. This table could be generated with `est_rscu` or prepared manually.

Value

a named vector of CAI values

References


Examples

```
# estimate CAI of yeast genes based on RSCU of highly expressed genes
heg <- head(yeast_exp[order(-yeast_exp$fpkm), ], n = 500)
cf_all <- count_codons(yeast_cds)
cf_heg <- cf_all[heg$gene_id, ]
rscu_heg <- est_rscu(cf_heg)
cai <- get_cai(cf_all, rscu_heg)
head(cai)
hist(cai)
```
get_codon_table

Description

get_codon_table creates a codon table based on the given id of genetic code in NCBI.

Usage

get_codon_table(gcid = "1")

Arguments

gcid a string of genetic code id. run 'show_codon_tables()' to see available codon tables.

Value

a 'data.table' with four columns: aa_code, amino_acid, codon, and subfam.

Examples

# Standard genetic code
get_codon_table()

# Vertebrate Mitochondrial genetic code
get_codon_table(gcid = '2')

get_cscg

Mean Codon Stabilization Coefficients

Description

get_cscg calculates Mean Codon Stabilization Coefficients of each CDS.

Usage

get_cscg(cf, csc)

Arguments

cf matrix of codon frequencies as calculated by 'count_codons()'.
csc table of Codon Stabilization Coefficients as calculated by 'est_csc()'.

Value

a named vector of cscg values.
get_enc

References

Examples

# estimate CSCg of yeast genes
yeast_csc <- est_csc(yeast_cds, yeast_half_life)
cf_all <- count_codons(yeast_cds)
cscg <- get_cscg(cf_all, csc = yeast_csc)
head(cscg)
hist(cscg)

get_enc

Calculate ENC

Description
get_enc computes ENC of each CDS

Usage
get_enc(cf, codon_table = get_codon_table())

Arguments
cf          matrix of codon frequencies as calculated by ‘count_codons()’.
codon_table codon_table a table of genetic code derived from ‘get_codon_table’ or ‘create_codon_table’.

Value
vector of ENC values, sequence names are used as vector names

References

Examples

# estimate ENC of yeast genes
cf_all <- count_codons(yeast_cds)
enc <- get_enc(cf_all)
head(enc)
hist(enc)
get_fop  

Fraction of optimal codons (Fop)

Description

get_fop calculates the fraction of optimal codons (Fop) of each CDS.

Usage

get_fop(seqs, codon_table = get_codon_table())

Arguments

seqs  
CDS sequences of all protein-coding genes. One for each gene.

codon_table  
a table of genetic code derived from ‘get_codon_table’ or ‘create_codon_table’.

Value

a named vector of fop values.

References


Examples

# estimate Fop of yeast genes
fop <- get_fop(yeast_cds)
head(fop)
hist(fop)

get_gc  

GC contents

Description

Calculate GC content of the whole sequences.

Usage

get_gc(cf)
get_gc3s

Arguments

cf          matrix of codon frequencies as calculated by 'count_codons()'.

Value

a named vector of GC contents.

Examples

# estimate GC content of yeast genes
cf_all <- count_codons(yeast_cds)
gc <- get_gc(cf_all)
head(gc)
hist(gc)

get_gc3s

GC contents at synonymous 3rd codon positions

Description

Calculate GC content at synonymous 3rd codon positions.

Usage

get_gc3s(cf, codon_table = get_codon_table())

Arguments

cf          matrix of codon frequencies as calculated by 'count_codons()'.
codon_table a table of genetic code derived from 'get_codon_table' or 'create_codon_table'.

Value

a named vector of GC3s values.

References


Examples

# estimate GC3s of yeast genes
cf_all <- count_codons(yeast_cds)
bc3s <- get_gc3s(cf_all)
head(bc3s)
hist(bc3s)
get_gc4d

*GC contents at 4-fold degenerate sites*

**Description**

Calculate GC content at synonymous position of codons (using four-fold degenerate sites only).

**Usage**

```r
get_gc4d(cf, codon_table = get_codon_table())
```

**Arguments**

- `cf`: matrix of codon frequencies as calculated by `count_codons()`.
- `codon_table`: a table of genetic code derived from `get_codon_table` or `create_codon_table`.

**Value**

a named vector of GC4d values.

**Examples**

```r
# estimate GC4d of yeast genes
cf_all <- count_codons(yeast_cds)
gc4d <- get_gc4d(cf_all)
head(gc4d)
hist(gc4d)
```

get_tai

*Calculate TAI*

**Description**

get_tai calculates tRNA Adaptation Index (TAI) of each CDS

**Usage**

```r
get_tai(cf, trna_w)
```

**Arguments**

- `cf`: matrix of codon frequencies as calculated by `count_codons()`.
- `trna_w`: tRNA weight for each codon, can be generated with `est_trna_weight()`.
Value

a named vector of TAI values

References


Examples

# calculate TAI of yeast genes based on genomic tRNA copy numbers
w <- est_trna_weight(yeast_trna_gcn)
cf_all <- count_codons(yeast_cds)
tai <- get_tai(cf_all, w)
head(tai)
hist(tai)

human_mt

human mitochondrial CDS sequences

Description

CDSs of 13 protein-coding genes in the human mitochondrial genome extracted from ENSEMBL Biomart

Usage

human_mt

Format

a DNAStringSet of 13 sequences

Source

<https://www.ensembl.org/index.html>

Examples

head(human_mt)
plot_ca_pairing  
*Plot codon-anticodon pairing relationship*

**Description**

plot_ca_pairing returns the RSCU value of codons

**Usage**

```r
plot_ca_pairing(codon_table = get_codon_table(), plot = TRUE)
```

**Arguments**

- `codon_table`: a table of genetic code derived from `get_codon_table` or `create_codon_table`.
- `plot`: whether to plot the pairing relationship

**Value**

a data.table of codon info and RSCU values

**Examples**

```r
ctab <- get_codon_table(gcid = '2')
pairing <- plot_ca_pairing(ctab)
head(pairing)
```

rev_comp  
*Reverse complement*

**Description**

rev_comp creates reverse complemented version of the input sequence

**Usage**

```r
rev_comp(seqs)
```

**Arguments**

- `seqs`: input sequences, DNAStringSet or named vector of sequences

**Value**

reverse complemented input sequences as a DNAStringSet.
**seq_to_codons**

**Examples**

```r
# reverse complement of codons
rev_comp(Biostrings::DNAStringSet(c('TAA', 'TAG')))
```

**Description**

seq_to_codons converts a coding sequence to a vector of codons

**Usage**

```r
seq_to_codons(seq)
```

**Arguments**

- `seq` DNAString, or an object that can be coerced to a DNAString

**Value**

a character vector of codons

**Examples**

```r
# convert a CDS sequence to a sequence of codons
seq_to_codons('ATGTGGTAG')
seq_to_codons(yeast_cds[[1]])
```

**show_codon_tables**

**Description**

show_codon_tables print a table of available genetic code from NCBI through 'Biostrings::GENETIC_CODE_TABLE'.

**Usage**

```r
show_codon_tables()
```

**Value**

No return value (NULL). Available codon tables will be printed out directly.
Examples

```r
# print available NCBI codon table IDs and descriptions.
show_codon_tables()
```

---

**yeast_cds**

**yeast CDS sequences**

**Description**

CDSs of all protein-coding genes in Saccharomyces_cerevisiae

**Usage**

`yeast_cds`

**Format**

a DNAStringSet of 6600 sequences

**Source**


**Examples**

```r
head(yeast_cds)
```

---

**yeast_exp**

**yeast mRNA expression levels**

**Description**

Yeast mRNA FPKM determined from rRNA-depleted (Ribozero) total RNA-Seq libraries. RUN1_0_WT and RUN2_0_WT (0 min after RNA Pol II repression) were averaged and used here.

**Usage**

`yeast_exp`

**Format**

a data.frame with 6717 rows and three columns:

- `gene_id` gene ID
- `gene_name` gene name
- `fpkm` mRNA expression level in Fragments per kilobase per million reads
**Source**


**References**


**Examples**

head(yeast_exp)

---

<table>
<thead>
<tr>
<th>yeast_half_life</th>
<th>Half life of yeast mRNAs</th>
</tr>
</thead>
</table>

**Description**

Half life of yeast mRNAs in Saccharomyces cerevisiae calculated from rRNA-deleted total RNAs by Presnyak et al.

**Usage**

yeast_half_life

**Format**

a data.frame with 3888 rows and three columns:

- **gene_id** gene id
- **gene_name** gene name
- **half_life** mRNA half life in minutes

**Source**

<https://doi.org/10.1016/j.cell.2015.02.029>

**References**


**Examples**

head(yeast_half_life)
yeast_trna_gcn

Description
Yeast tRNA gene copy numbers (GCN) by anticodon obtained from gtRNAdb.

Usage
yeast_trna_gcn

Format
a named vector with a length of 41. Value names are anticodons.

Source
<http://gtrnadb.ucsc.edu/genomes/eukaryota/Scere3/sacCer3-mature-tRNAs.fa>

References

Examples
yeast_trna_gcn
Index

* datasets
  * aa2codon, 2
  * human_mt, 15
  * yeast_cds, 18
  * yeast_exp, 18
  * yeast_half_life, 19
  * yeast_trna_gcn, 20

aa2codon, 2
check_cds, 3
count_codons, 4
create_codon_table, 5

est_csc, 5
est_optimal_codons, 6
est_rscu, 7
est_trna_weight, 8

get_cai, 9
get_codon_table, 10
get_cscg, 10
get_enc, 11
get_fop, 12
get_gc, 12
get_gc3s, 13
get_gc4d, 14
get_tai, 14

human_mt, 15

plot_ca_pairing, 16

rev_comp, 16

seq_to_codons, 17
show_codon_tables, 17

yeast_cds, 18
yeast_exp, 18
yeast_half_life, 19
yeast_trna_gcn, 20