Package 'sscu'

November 12, 2026

Type Package

Title Strength of Selected Codon Usage

Version 2.42.0

Date 2016-12-1

Author Yu Sun

Maintainer Yu Sun <sunyu1357@gmail.com>

Description The package calculates the indexes for selective stength in codon usage in bacteria species. (1) The package can calculate the strength of selected codon usage bias (sscu, also named as s_index) based on Paul Sharp's method. The method take into account of background mutation rate, and focus only on four pairs of codons with universal translational advantages in all bacterial species. Thus the sscu index is comparable among different species. (2) The package can detect the strength of translational accuracy selection by Akashi's test. The test tabulating all codons into four categories with the feature as conserved/variable amino acids and optimal/non-optimal codons. (3) Optimal codon lists (selected codons) can be calculated by either op_highly function (by using the highly expressed genes compared with all genes to identify optimal codons), or op_corre_CodonW/op_corre_NCprime function (by correlative method developed by Hershberg & Petrov). Users will have a list of optimal codons for further analysis, such as input to the Akashi's test. (4) The detailed codon usage information, such as RSCU value, number of optimal codons in the highly/all gene set, as well as the genomic gc3 value, can be calculate by the optimal_codon_statistics and genomic_gc3 function. (5) Furthermore, we added one test function low_frequency_op in the package. The function try to find the low frequency optimal codons, among all the optimal codons identified by the op_highly function.

Depends R (>= 3.3)

Imports Biostrings (>= 2.36.4), seqinr (>= 3.1-3), BiocGenerics (>= 0.16.1)

Suggests knitr, rmarkdown

VignetteBuilder knitr

LazyLoad yes

License GPL (>= 2)

biocViews Genetics, GeneExpression, WholeGenome

git_url https://git.bioconductor.org/packages/sscu

git_branch RELEASE_3_23

git_last_commit 8b7125b

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

sscu-package 2
akashi_test 3
genomic_gc3 5
low_frequency_op 6
op_corre_CodonW 7
op_corre_NCprime 9
op_highly 10
op_highly_stats 11
s_index 13

Index **16**

`sscu-package` *Strength of Selected Codon Usage*

Description

The package can calculate the indexes for selective stength in codon usage in bacteria species. (1) The package can calculate the strength of selected codon usage bias (sscu, also named as s_index) based on Paul Sharp's method. The method take into account of background mutation rate, and focus only on four pairs of codons with universal translational advantages in all bacterial species. Thus the sscu index is comparable among different species. (2) Translational accuracy selection can be inferred from Akashi's test. The test tabulating all codons into four categories with the feature as conserved/variable amino acids and optimal/non-optimal codons. (3) Optimal codon lists (selected codons) can be calculated by either op_highly function (by using the highly expressed genes compared with all genes to identify optimal codons biased used in the highly expressed genes), or op_corre_CodonW/op_corre_NCprime function (by correlative method developed by Hershberg & Petrov). Users will have a list of optimal codons for further analysis, such as input to the Akashi's test. (4) The detailed codon usage information, such as RSCU value, number of optimal codons in the highly/all gene set, as well as the genomic gc3 value, can be calculate by the optimal_codon_statistics and genomic_gc3 function. (5) Furthermore, we added one test function proportion_index in the package. The function focus on the proportion of optimal codon against its corresponding non-optimal codons for the the four and six codon boxes.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Yu Sun

Maintainer: Yu Sun <sunyu1357@gmail.com>

References

Sharp PM, Bailes E, Grocock RJ, Peden JF, Sockett RE (2005). Variation in the strength of selected codon usage bias among bacteria. Nucleic Acids Research. Sharp PM, Emery LR, Zeng K. 2010. Forces that influence the evolution of codon bias. Philos Trans R Soc Lond B Sci. 365:1203-1212. Hershberg R, Petrov DA. 2009. General rules for optimal codon choice. Plos Genet. 5:e1001115. Akashi H. Synonymous codon usage in Drosophila melanogaster: natural selection and translational accuracy. Genetics 1994 Mar;136(3):927-35. http://drummond.openwetware.org/Akashi's_Test.html Novembre JA. 2002. Accounting for background necleotide composition when measuring codon usage bias. Mol Biol Evol. 19: 1390-1394. https://github.com/jnovembre/ENCprime http://codonw.sourceforge.net/

`akashi_test` *akashi test for codon usage*

Description

The function calculate Akashi's test for translational accuracy selection on coding sequences. Akashi proposed the translational accurary theory for codon usage in 1994. The theory suggest that the optimal codons are codons that translated more accurately than the other codons, and these codons are favored in the important sites, such as the evolutionary conserved amino acid sites, whereas the less conserved amino acids sites are more tolerable to the non-optimal codons. For detailed information, see reference listed below.

Usage

```
akashi_test(contingency_file=NULL)
```

Arguments

`contingency_file`
: a character vector for the filepath of the contingency file, which was generated by the perl script make_contingency_table.pl in the perl_script folder in the package

Details

The function calculate Akashi's test for translational accuracy selection on coding sequences.

The input of the function is a contingency file, you can find an example in the folder akashi_test in the sscu directory. You can either make the file by yourself, or by the perl script make_contingency_table.pl in the akashi_test folder. You can check the detailed usage of the perl script by reading the first few lines in the perl script. The perl script tabulates and calculate the a,b,c,d entries for the coding sequences, and output the four values for each amino acid for each gene, example as the contingency_file_Gvag. The input of the perl script is a folder contains the codon alignments of the genes that you are interested to calculate. For detailed information of Akashi's test, you can either refer to the publication by Akashi, or to the website http://drummond.openwetware.org/Akashi's_Test.html

The function reads and calculates the Z value, p value and odd ratio for the Akashi's test. In addition, it calculates the conserved, variable, optimal and nonoptimal sites for the coding sequences.

Value

a list of numeric vector is returned

`Z`	Z value for Akashi's test
`p`	p value for Akashi's test
`odd_ratio`	odd_ratio for Akashi's test
`conserved_optimal_sites`	total number of conserved optimal sites
`conserved_non_optimal_sites`	total number of conserved non-optimal sites
`variable_optimal_sites`	total number of variable optimal sites
`variable_non_optimal_sites`	total number of variable non-optimal sites
`con_var_ratio`	the ratio of conserved against variable sites

Author(s)

Yu Sun

References

Akashi H. Synonymous codon usage in Drosophila melanogaster: natural selection and translational accuracy. Genetics 1994 Mar;136(3):927-35. http://drummond.openwetware.org/Akashi's_Test.html

Examples

```
# ---------------------------------------------- #
#     Gardnerella vaginalis example              #
# ---------------------------------------------- #

  # Here is an example to calculate the genomic gc3
  # input the one multifasta files to calculate genomic gc3
   akashi_test(system.file("akashi_test/contingency_file_Gvag",package="sscu"))
```

genomic_gc3 *genomic gc3 for an multifasta genomic file*

Description

The function calculates the genomic gc3 for an multifasta genomic CDS file. The function first concatenated all the CDS sequences in the file into one long CDS string, than calculated the gc3 from the GC3 function in seqinr package. You can also use the function to calculate the gc3 for a single gene, or a set of genes, depends what content you put in the input file.

Usage

```
genomic_gc3(inputfile)
```

Arguments

inputfile	a character vector for the filepath of the whole genome cds file

Details

The function calculates the genomic gc3 for an multifasta genomic CDS file. The function first concatenated all the CDS sequences in the file into one long CDS string, than calculated the gc3 from the GC3 function in seqinr package. You can also use the function to calculate the gc3 for a single gene, or a set of genes, depends what content you put in the input file. The result can be used as input for the s_index calculation.

Value

a numeric vector genomic_gc3 is returned

Author(s)

Yu Sun

See Also

GC3 in seqinr library

Examples

```
# ---------------------------------------------- #
#     Lactobacillus kunkeei example              #
# ---------------------------------------------- #

  # Here is an example to calculate the genomic gc3
  # input the one multifasta files to calculate genomic gc3
  genomic_gc3(system.file("sequences/L_kunkeei_genome_cds.ffn",package="sscu"))
```

`low_frequency_op` *the function identify low frequency optimal codons*

Description

The function low_frequency_op identify the low frequency optimal codons. Based on the previous study, in some species, the optimal codons identified by the op_highly function has bit strange patterns: the optimal codons do have lower prequency than the non-optimal codons. It occurs most in the mutation-shifting species such as G. vaginalis. The function can identify these low frequency optimal codons.

Usage

```
low_frequency_op(high_cds_file = NULL, genomic_cds_file = NULL, p_cutoff=0.01)
```

Arguments

`high_cds_file`	a character vector for the filepath of the highly expressed genes
`genomic_cds_file`	a character vector for the filepath of the whole genome cds file
`p_cutoff`	a numeric vector to set the cutoff of p value for the chi.square test, default is set to 0.01

Details

The function low_frequency_op identify the low frequency optimal codons. Based on the previous study (Sun Y et al. Switches in genomic GC content drives shifts of optimal codons under sustained selection on synonymous sites. Genome Biol Evol. 2016 Aug 18.), in some species, the optimal codons identified by the op_highly function has bit strange patterns: the optimal codons do have lower prequency than the non-optimal codons. It occurs most in the mutation-shifting species such as G. vaginalis. The function can identify these low frequency optimal codons.

The function first calculated all the optimal codons statistics by the function op_highly_stats in the package, then tried to find the low frequency optimal codons with the following settings: 1) it has to be an optimal codon 2) The RSCU value for the optimal codon is lower than 0.7 (this is the quite arbitrary setting) 3) the RSCU value for the optimal codon is lower than the corresponding non-optimal codons, which has the same first two nucleotide as the optimal codon but the third position experience the point mutation transition. With this detailed filters, we can identify the low frequency optimal codons in the given genome.

The argument high_cds_file should specific the path for the highly expressed gene dataset. It is up to the users how to define which dataset of highly expressed genes. Some studies use the expression data, or Nc value to divide genes into highly/lowly sets. Other studies use a specific dataset, such as only including the very highly expressed genes (ribosomal genes). In the example, I used the ribosomal genes as the representative for the highly expressed genes.

The arguments, genomic_cds_file, is used to calculate the optimal codons and statistics for highly and all genes.

Same as the op_highly and op_highly_stats, the p value cutoff was set as 0.01.

Value

a list is returned

`low_frequency_optimal_codons`
: a dataframe with statistics for the low frequency optimal codons

`corresponding_codons`
: a dataframe with statistics for the corresponding codons

Author(s)

Yu Sun

References

Sun Y et al. Switches in genomic GC content drives shifts of optimal codons under sustained selection on synonymous sites. Genome Biol Evol. 2016 Aug 18.

unpublished paper from Yu Sun

See Also

the op_highly and op_highly_stats function in the same package

Examples

```
# ---------------------------------------------- #
#     Lactobacillus kunkeei example              #
# ---------------------------------------------- #

  # Here is an example to load the data included in the sscu package
  # input the two multifasta files to calculate sscu
  low_frequency_op(high_cds_file=system.file("sequences/Gvag_highly.ffn",package="sscu"),genomic_cds_file=syst

 # if you want to load your own data, you just specify the file path for your input as these examples
 # low_frequency_op(high_cds_file="/home/yu/Data/codon_usage/bee_endosymbionts/sharp_40_highly_dataset/Bin2.ff
```

`op_corre_CodonW` *Identify optimal codons by using the correlative method from Hershberg & Petrov, the input file is from CodonW*

Description

The function identify the optimal codons based on the correlative method from Hershberg & Petrov. This method take the whole genome into consideration, and predict the optimal codons by making the correlation between the frequency of each codon within each gene and the overall codon bias (Nc or Nc'). The input file include the correspondence analysis output file from the program CodonW (to get the Nc value), and the genomic cds file (to get the codon usage information for each gene).

Usage

```
op_corre_CodonW(genomic_cds_file=NULL, correspondence_file=NULL)
```

Arguments

genomic_cds_file	a character vector for the filepath of the whole genome cds file
correspondence_file	a character vector for the filepath of the correspondence file from the CodonW program

Details

The function identify the optimal codons based on the correlative method from Hershberg & Petrov. This method take the whole genome into consideration, and predict the optimal codons by making the correlation between the frequency of each codon within each gene and the overall codon bias (Nc or Nc'). The input file include the correspondence analysis output file from the program CodonW (to get the Nc value), and the genomic cds file (to get the codon usage information for each gene).

For further details regard how to use CodonW, you can refer to the site http://codonw.sourceforge.net/. Note, you must input the same genomic cds file to CodonW and to the op_corre_CodonW funtion, so that the order and number of genes are consistent in the files.

Value

a character vector for all the optimal codons is returned

Author(s)

Yu Sun

References

Hershberg R, Petrov DA. 2009. General rules for optimal codon choice. Plos Genet. 5:e1001115.
http://codonw.sourceforge.net/

Examples

```
# ---------------------------------------------- #
#     Lactobacillus kunkeei example              #
# ---------------------------------------------- #

  # Here is an example to calculate the genomic gc3
  # input the one multifasta files to calculate genomic gc3
  op_corre_CodonW(genomic_cds_file=system.file("sequences/Gvag_genome_cds.ffn",package="sscu"),correspondence_
```

op_corre_NCprime *Identify optimal codons by using the correlative method from Hershberg & Petrov, the input file is from NCprime*

Description

The function identify the optimal codons based on the correlative method from Hershberg & Petrov. This method take the whole genome into consideration, and predict the optimal codons by making the correlation between the frequency of each codon within each gene and the overall codon bias (Nc or Nc'). The input file include the output file from the program ENCprime (to get the Nc and Nc' value), and the genomic cds file (to get the codon usage information for each gene).

Usage

```
op_corre_NCprime(genomic_cds_file=NULL, nc_file=NULL)
```

Arguments

genomic_cds_file	a character vector for the filepath of the whole genome cds file
nc_file	a character vector for the filepath of the correspondence file from the CodonW program

Details

The function identify the optimal codons based on the correlative method from Hershberg & Petrov. This method take the whole genome into consideration, and predict the optimal codons by making the correlation between the frequency of each codon within each gene and the overall codon bias (Nc or Nc'). The input file include the output file from the program ENCprime (to get the Nc and Nc' value), and the genomic cds file (to get the codon usage information for each gene).

For further details regard how to use ENCprime, you can refer to the site https://github.com/jnovembre/ENCprime. Note, you must input the same genomic cds file to ENCprime and to the op_corre_NCprime funtion, so that the order and number of genes are consistent in the two files.

Value

a character vector for all the optimal codons is returned

Author(s)

Yu Sun

References

Hershberg R, Petrov DA. 2009. General rules for optimal codon choice. Plos Genet. 5:e1001115. Novembre JA. 2002. Accounting for background necleotide composition when measuring codon usage bias. Mol Biol Evol. 19: 1390-1394. https://github.com/jnovembre/ENCprime

Examples

```
# ---------------------------------------------- #
#     Lactobacillus kunkeei example              #
# ---------------------------------------------- #

  # Here is an example to calculate the genomic gc3
  # input the one multifasta files to calculate genomic gc3
  op_corre_NCprime(genomic_cds_file=system.file("sequences/LbDelBA1_genome_cds.ffn",package="sscu"),nc_file=sy
```

`op_highly` *Identify optimal codons by using the highly expressed genes method*

Description

Optimal codons can be defined as codons significantly enriched in the highly expressed genes compared to the lowly expressed genes, or other set of appropriate reference genes. In another word, these codons were favored by translational selection. This function calculate the optimal codon list, thus user could have a general idea of which codons were preferred by selection in the genome.

Usage

```
op_highly(high_cds_file = NULL,ref_cds_file = NULL,p_cutoff = 0.01)
```

Arguments

`high_cds_file`	a character vector for the filepath of the highly expressed genes
`ref_cds_file`	a character vector for the filepath of the reference cds file
`p_cutoff`	a numeric vector to set the cutoff of p value for the chi.square test, default is set to 0.01

Details

Optimal codons can be defined as codons significantly enriched in the highly expressed genes compared to the lowly expressed genes, or other set of appropriate reference genes. In another word, these codons were favored by translational selection, which was strongest among highly expressed genes. This function calculate the optimal codon list with p-values, thus user could have a general idea of which codons were preferred by selection in the genome.

The argument high_cds_file should specific the path for the highly expressed gene dataset. It is up to the users how to define which dataset of highly expressed genes. Some studies use the expression data, or Nc value to divide genes into highly/lowly sets. Other studies use a specific dataset, such as only including the very highly expressed genes (ribosomal genes).

The argument ref_cds_file should specific the path for the lowly expressed gene dataset, or any appropriate dataset. In Sharp PM paper (Forces that influence the evolution of codon bias), he used the all gene data set as neutral reference and also get a list of optimal codons.

The argument p_cutoff set the cutoff for p values in the chi.square test. Only codons are significantly enriched in the highly expressed genes are marked with + symbol in the ouotput tables. The codons are significantly lower presented in the highly expressed genes are marked with - symbol. The codons are not significantly differently presented compared to the reference dataset are marked with NA symbol.

Value

a character vector for all the optimal codons is returned

Author(s)

Yu Sun

References

Sharp PM, Emery LR, Zeng K. 2010. Forces that influence the evolution of codon bias. Philos Trans R Soc Lond B Sci. 365:1203-1212.

Examples

```
# ------------------------------------------------ #
#     Lactobacillus kunkeei example                #
# ------------------------------------------------ #

  # Here is an example to load the data included in the sscu package
 op_highly(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),ref_cds_file=system.file

  # if you want to set the p value cutoff as 0.05
 op_highly(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),ref_cds_file=system.file

 # if you want to load your own data, you just specify the file path for your input as these examples
 # op_highly(high_cds_file = "/home/yu/Data/codon_usage/bee_endosymbionts/sharp_40_highly_dataset/Bin2.ffn",ref
```

---

op_highly_stats *statistics for the optimal codons*

Description

Optimal codons can be defined as codons significantly enriched in the highly expressed genes compared to the lowly expressed genes, or other set of appropriate reference genes (see function op_highly in this package). In another word, these codons were favored by translational selection. This function calculate the optimal codon list with p-values, thus user could have a general idea of which codons were preferred by selection in the genome.

Usage

```
op_highly_stats(high_cds_file = NULL,ref_cds_file = NULL,p_cutoff = 0.01)
```

Arguments

high_cds_file	a character vector for the filepath of the highly expressed genes
ref_cds_file	a character vector for the filepath of the reference cds file
p_cutoff	a numeric vector to set the cutoff of p value for the chi.square test, default is set to 0.01

Details

Optimal codons can be defined as codons significantly enriched in the highly expressed genes compared to the lowly expressed genes, or other set of appropriate reference genes (see function op_highly in this package). In another word, these codons were favored by translational selection, which was strongest among highly expressed genes. This function calculate the optimal codon list with p-values, thus user could have a general idea of which codons were preferred by selection in the genome.

The argument high_cds_file should specific the path for the highly expressed gene dataset. It is up to the users how to define which dataset of highly expressed genes. Some studies use the expression data, or Nc value to divide genes into highly/lowly sets. Other studies use a specific dataset, such as only including the very highly expressed genes (ribosomal genes).

The argument ref_cds_file should specific the path for the lowly expressed gene dataset, or any appropriate dataset. In Sharp PM paper (Forces that influence the evolution of codon bias), he used the all gene data set as neutral reference and also get a list of optimal codons.

The argument p_cutoff set the cutoff for p values in the chi.square test. Only codons are significantly enriched in the highly expressed genes are marked with + symbol in the ouotput tables. The codons are significantly lower presented in the highly expressed genes are marked with - symbol. The codons are not significantly differently presented compared to the reference dataset are marked with NA symbol.

The function also output the rscu value for the high expressed dataset and reference dataset.

Value

a dataframe is returned

rscu_high	rscu value for the highly expressed dataset
rscu_ref	rscu value for the reference dataset
high_No_codon	number of codons found in the highly expressed dataset
high_expect_No_codon	number of expected codons in the highly expressed dataset
ref_No_codon	number of codons found in the reference dataset
ref_expect_No_codon	number of expected codons in the reference dataset
p_value	p value for the chi.square test
symbol	codons are significantly enriched in the highly expressed genes are marked with +; codons are significantly lower presented in the highly expressed genes are marked with -; codons are not significantly differently presented compared to the reference dataset are marked with NA

Author(s)

Yu Sun

See Also

uco in seqinr library for rscu calculation.

Examples

```
# ------------------------------------------------ #
#     Lactobacillus kunkeei example               #
# ------------------------------------------------ #

  # Here is an example to load the data included in the sscu package
 op_highly_stats(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),ref_cds_file=syste

  # if you want to set the p value cutoff as 0.05
 op_highly_stats(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),ref_cds_file=syste

 # if you want to load your own data, you just specify the file path for your input as these examples
 # optimal_codon_statistics(high_cds_file = "/home/yu/Data/codon_usage/bee_endosymbionts/sharp_40_highly_datase
```

---

s_index *S index (Strength of Selected Codon Usage)*

Description

The function sscu calculates the S index (strength of selected codon usage bias) for bacteria species based on Paul Sharp's method. The method take into account of background mutation rate, and focus only on codons with universal translational advantages in all bacterial species. Thus the sscu index can be used to quantify the strength of translational selection and is comparable among different species.

Usage

```
s_index(high_cds_file = NULL, genomic_cds_file = NULL, gc3 = NULL)
```

Arguments

high_cds_file	a character vector for the filepath of the highly expressed genes
genomic_cds_file	a character vector for the filepath of the whole genome cds file
gc3	a numeric vector with gc3 value, eg, 0.5

Details

The function calculates the S index (strength of selected codon usage bias) for bacteria species based on Paul Sharp's method.The method take into account of background mutation rate (in the program, two arguments genomic_cds_file and gc3, are input to calculate mutation), and focus only on codons with universal translational advantages in all bacterial species (in the program, one argument high_cds_file, is input to calculate these codons). Thus the s index can be used to quantify the strength of translational selection and is comparable among different species.

The argument high_cds_file much be specified with the input filepath for the highly expressed genes. The file should be a multifasta file contains 40 highly, including elongation factor Tu, Ts, G, 50S ribosomal protein L1 to L6, L9 to L20, 30S ribosomal protein S2 to S20. This file can be generated by either directly extract these DNA sequence from genbank file, or parse by blast program. For the four amino acids (Phy, Tyr, Ile and Asn), the C-ending codons are always preferred than the U-ending codons. Thus, only these four codons were taken into account in the analyses.

The two arguments, genomic_cds_file or gc3, is used to calculate the genomic mutation rate, and one of them must be specified. The genomic_cds_file should be a multifasta file contains all the coding sequences in the genome, and the function use it to calculate the genomic gc3 and mutation rate. If the gc3 value for the genome is known already, you can specify it in the argument gc3. If both the genomic_cds_file and gc3 arguments are specified, the function will use the genomic_cds_file to calculate mutation rate, and neglect the gc3 argument.

Value

a numeric vector s-index is returned

Author(s)

Yu Sun

References

Sharp PM, Bailes E, Grocock RJ, Peden JF, Sockett RE (2005). Variation in the strength of selected codon usage bias among bacteria. Nucleic Acids Research.

See Also

uco in seqinr library for rscu calculation

Examples

```
# ------------------------------------------------ #
#     Lactobacillus kunkeei example               #
# ------------------------------------------------ #

  # Here is an example to load the data included in the sscu package
  # input the two multifasta files to calculate sscu
 s_index(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),genomic_cds_file=system.fi

  # alternatively, input one multifasta file and gc3 content to calculate sscu
 s_index(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),gc3=0.76)
```

```
# if you want to load your own data, you just specify the file path for your input as these examples
# s_index(high_cds_file="/home/yu/Data/codon_usage/bee_endosymbionts/sharp_40_highly_dataset/Bin2.ffn",genomi
# s_index(high_cds_file="/home/yu/Data/codon_usage/bee_endosymbionts/sharp_40_highly_dataset/Bin2.ffn",gc3=0.
```

Index

akashi_test, 3

GC3, *5*
genomic_gc3, 5

low_frequency_op, 6

op_corre_CodonW, 7
op_corre_NCprime, 9
op_highly, 10
op_highly_stats, 11
optimal_codon_statistics *(*op_highly_stats*)*, 11
optimal_codons *(*op_highly*)*, 10
optimal_codons_table *(*op_highly_stats*)*, 11

s_index, 13
selected_codons *(*op_highly*)*, 10
sscu-package, 2

uco, *13, 14*