sscu user manual (0.99.3)

Yu Sun

2016-03-10

Contents

1. Background

The central dogma is the cornerstone of modern molecular biology, as it explains the information processing flow within a biological system is in the direction of DNA to RNA to protein (crick 1970). An important step in this process is to decode the information stored in the cryptic nucleotides sequences into protein molecules. Sixty-four tri-nucleotide units (codons) are correspond to the coding of twenty amino acids, thus almost all the amino acids has more than one matching codons. The codon choice for an amino acid is not random for most species, and it has been shown that mutation and selection are the two major forces shaping the codon usage pattern. For the majority of the gene, mutation bias is the factor determining the codon choice, but the influence from selection is increased with the genes' expression level. Sharp released a mathematic formula, named as S index, to quantify the strength of selection in the synonymous codon. The method calculates the relative proportion of C- and U-ending codons in the two codon boxes, and also takes into consideration of the genomic mutation bias. The method has been proved to be an effective way to quantify and compare the codon selection among different species. However, no program or pipeline has been released to do the calculation, thus every codon selection study need to write individual code to perform the same task. It is not only waste a lot of time, but also have the possibility to introduce errors during the coding process. In this study, we developed an R package (SSCU) for the standard S index calculation, but it also has other function including genomic gc3 calculation, optimal codon identification and optimal codon index for the four and six codon boxes. This package is useful for the calculation of selective profile in codon usage studies, and it is a good complement to other major codon usage programs, such as CodonW.

2. Function overview

2.1 s_index

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

The function calculate the s index 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 the mutation rate), 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 universally preferred than the U-ending codons. Thus, only these four codons were taken into account in the analyses.

The second 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, or calculated by genomic_gc3 function below, 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.

2.2 genomic_gc3

genomic_gc3(genomic_cds_file)

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 CDS sequences you put in the input file. The result can be used as input for the s_index calculation.

2.3 optimal_codons

optimal_codons(high_cds_file=NULL,ref_cds_file=NULL,p_cutoff=0.05)

If you want to know the detailed codon usage information, i.e. which codons are selected preferred (optimal codons), you can run the function optimal_codons in the package. The optimal codons are defined as codons significantly enriched in the highly expressed genes compared to the lowly expressed genes, or other set of reference genes. The 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 to infer codons used significantly in highly expressed genes.

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.

2.4 optimal_index

optimal_index(high_cds_file=NULL,genomic_cds_file=NULL)

The function optimal_index is developed in this study, which estimate the relative amount of GC-ending optimal codon for the four and six codon boxes codon in a given mutational background. The function has similar mathematical formula as sscu and also take into account of background mutation rate, thus is comparable with the S index. However, since the set of GC-ending optimal codons are likely to be different among different species, the index can not be compared among different species.

The argument high_cds_file must 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 arguments, genomic_cds_file, is used to calculate the genomic mutation rate (gc3). 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.

Noted, most of the AT biased genomes do not have any GC-ending optimal codons for the four and six codon boxes, thus the function will report NA as output.

3 Workflow

Here is an example of how to use these functions from the package. We will use the genomic CDS file from the Lactobacillus. kunkeei and Gardnerella vaginalis as exemple for the workflow. These files are included in the sequences folder in the package.

First, we can calculate the genomic gc3 for the L. kunkeei to have a general statistics about the genome. We show how to do it here with genomic_gc3 function:

```r
library(sscu)
genomic_gc3(system.file("sequences/L_kunkeei_genome_cds.ffn",package="sscu"))
```

```
## [1] 0.3083719
```

The genomic gc3 content is 0.308, which is a low gc genome.

Then, we can further check the codon usage pattern in detailes for L. kunkeei by using the optimal_codons function:

```r
head(optimal_codons(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),
               ref_cds_file=system.file("sequences/L_kunkeei_genome_cds.ffn",package="sscu")))
```

```
##     codon aa rscu_high rscu_ref high_No_codon high_expect_No_codon
## TTT   TTT  F      0.66     1.10            58                   88
## TTC   TTC  F      1.34     0.90           118                   88
## TTA   TTA  L      2.58     2.54           198                   77
## TTG   TTG  L      0.56     1.09            43                   77
## TCT   TCT  S      1.10     1.04            64                   58
## TCC   TCC  S      0.12     0.38             7                   58
##     ref_No_codon ref_expect_No_codon p_value symbol
## TTT        10540                9539   0.001      -
## TTC         8538                9539   0.005      +
## TTA        16254                6399   0.895     NA
## TTG          6971                    6399   0.000      -
## TCT          5263                    5062   0.792     NA
## TCC          1937                    5062   0.002      -
```

We can see several statistics for each codon, including RSCU values, actual and expected number of codons, the p value of chi.square test. The symbol indicates if the codon is highly (optimal codons), no difference, or lowly presented in the highly expressed genes compared to the reference gene set. Thus, you can get a list of optimal codons for the output. The output has 64 lines with each line for each codon, and we only show the first six lines for simplicity.

Next, if we want to see the selective profile for the species, which is the major function in the package. We can use the s_index function to do the calculation. We can use either the genomic gc3 content 0.308 as we got previously:

```r
s_index(gc3=0.308,
        high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"))
```

```
## [1] 1.541898
```

or you can directly input the CDS file in the function.

```r
s_index(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),
        genomic_cds_file=system.file("sequences/L_kunkeei_genome_cds.ffn",package="sscu"))
```

```
## [1] 1.540154
```

We can get the S index as 1.54 from both method. S index = 0 indicates that, in the highly expressed genes, the relative frequency of C/U-ending codons are exactly the same as the genomic gc3 (mutational pattern). S index > 0.3 indicate C-ending codons are used in significantly higher frequency than the U-ending codons, which further indicates that translational selection is affecting codon usage. In this case, S index of 1.54 suggests strong stranslational selection in L. kunkeei, which is even higher than E. coli (1.49).

The translational selection in the two codon boxes is very strong in L. kunkeei, the frequency is C-ending codons is significantly higher than U-ending codons, even taking into consideration of the genomic mutation bias. For the four and six codon boxes, the selection is generally very low, so the codon usage frequency is generally follow the mutation bias pattern. But the four and six codon boxes also have some optimal codons, are they strong enough to deviate the codon frequency from the mutational pattern? In this study, we develop a function called optimal_index to estimate the relative frequency between GC/AU-ending codons in the optimal codons in the four and six codon boxes.

Here is an example of L. kunkeei genome:

```r
optimal_index(high_cds_file=system.file("sequences/L_kunkeei_highly.ffn",package="sscu"),
          genomic_cds_file=system.file("sequences/L_kunkeei_genome_cds.ffn",package="sscu"))
```

```
## [1] NA
```

It indicates that the four and six codon boxes does not have any GC-ending optimal codons. This is general pattern for most species, suggesting muation is the dominant force for the codon choice for the four and six codon boxes.

This is another example of G. vaginalis:

```r
optimal_index(high_cds_file=system.file("sequences/Gvag_highly.ffn",package="sscu"),
           genomic_cds_file=system.file("sequences/Gvag_genome_cds.ffn",package="sscu"))
```

```
## $optimal_mutation_index
## [1] 0.01422826
##
## $s_index
## [1] 0.9697102
##
## $ratio
## [1] 0.01467269
```

It has GC-ending optimal codons, but the optimal index is only 0.014. The intepretation of optimal index is very similar to S index: the value 0 means the frequency of optimal codons is very close to the mutational pattern. Thus, for the G. vaginalis, although the translational selection can be detected in the four and six codon boxes (as the optimal codons), but they are so weak do alter the mutational dominated codon frequency. In another words, these optimal codons are low frequency optimal codons.

Again, we can use the function optimal_codons to get the detailed codon usage data:

```r
head(optimal_codons(high_cds_file=system.file("sequences/Gvag_highly.ffn",package="sscu"),
               ref_cds_file=system.file("sequences/Gvag_genome_cds.ffn",package="sscu")))
```

```
##     codon aa rscu_high rscu_ref high_No_codon high_expect_No_codon
## TTT   TTT  F      0.41     1.28            46                  112
## TTC   TTC  F      1.59     0.72           177                  112
## TTA   TTA  L      0.17     1.24            16                   92
## TTG   TTG  L      2.08     1.66           192                   92
## TCT   TCT  S      2.18     2.07           135                   62
## TCC   TCC  S      1.74     0.47           108                   62
##     ref_No_codon ref_expect_No_codon p_value symbol
## TTT        10639                8316   0.000      -
## TTC         5992                8316   0.000      +
## TTA         8260                6686   0.000      -
## TTG        11095                6686   0.080     NA
## TCT        11188                5396   0.687     NA
## TCC         2524                5396   0.000      +
```

By checking the optimal codon detailed information, it is clear several optimal codons do have low proportion among the highly expressed genes. For example GUC for Valine, GCC for Alanine, CUC for Leucine are all in low proportion compare to the corresponding non-optimal U-ending codons.