ReadMe for “**Specific Codons Control Cellular Resources and Fitness**” supplementary Matlab® code. Code can be accessed at <https://github.com/nair-lab>. Details on functions and examples of implementations are included below.

This file includes descriptions of how to use the following functions:

* RSCUarray2struct.m
* RSCUstruct.m
* CAI.m
* ENC.m
* Recode\_DNA\_singleAA\_DesiredCodon.m
* RecodeRand.m
* RecodeTarget\_CAI.m
* General\_Sequence\_Struct\_Builder.m

**Calculate CAI:**

Generate two data structures first, one for the sequence of interest, and one for a reference sequence, or set of sequences. If an array of codons and RSCU values is already available for either sequence, the function RSCUarray2struct.m should be used to generate the data structure(s). If you have sequences prepared in frame (ATG to stop codon), you can use the RSCUstruct.m function to create the same type of data structure. You can also use several genes concatenated together in frame as an input (e.g. the entire transcriptome of protein coding genes for an organism of interest).

* **RSCUarray2struct.m**
* For implementation example, see “**Example\_Calculate\_CAI\_ENC.m**”
  + Input1: Cell array of upper case codons and RSCU values without headers, e.g.:

|  |  |
| --- | --- |
| GCG | 1.2921 |
| GCT | 1.1017 |
| Codon | RSCU value |

* + Input2: Name of codon array, e.g. ‘Highly Expressed RSCU Values’
  + Output: RSCU data struct
* **RSCUstruct.m**
* For implementation example, see “**Example\_Calculate\_CAI\_ENC.m**”
  + Input1: Name of the sequence as a character vector, e.g. ‘GFP’.
  + Input2: Character vector of the input sequence, e.g. ‘atgatcata…’
  + Output: RSCU data struct

Once you have two data structures (output of RSCUstruct) of the sequence to be analyzed and the reference RSCU data struct, you can calculate CAI with the CAI.mfunction. Note that the calculated CAI using this function does not take into account the stop codon:

* **CAI.m**
* For implementation example, see “**Example\_Calculate\_CAI\_ENC.m**”
  + Input1: RSCUstruct of target sequence being analyzed
  + Input2: RSCUstruct of the reference RSCU values (e.g. from a set of highly expressed genes or from χ weights).
  + Output: CAI value

**Calculate ENC:**

Calculation of ENC is done using the ENC.mfunction, which uses the method described by Wright (1990):

Wright, F. (1990). The ‘effective number of codons’ used in a gene. *Gene*, *87*(1), 23–29. <https://doi.org/10.1016/0378-1119(90)90491-9>

* **ENC.m**
* For implementation example, see “**Example\_Calculate\_CAI\_ENC.m**”
  + Input: The input is just a struct generated by the function **RSCUstruct.m**. Note that due to the way ENC is mathematically derived, the function is only defined for sequences that have all 18 amino acids with multiple synonymous codons (every AA except met and trp) occur at least twice.
  + Output: ENC value

**Randomize a Sequence with Synonymous Codons**

The function RecodeRand.mwill re-code an entire gene sequence (start to stop codon), where every single codon is re-coded to an alternative synonymous codon. The function also allows user defined exclusions, where you can include a list of codons that should not be re-coded if desired.

* **RecodeRand.m**
* For implementation example, see “**Example\_Recode\_Seq\_Single\_Codon\_and\_Randomize.m**”
  + Input 1: input sequence, given as a character vector of DNA letters.
  + Input 2: exclusions, given as a cell array of codons.
  + Output: new sequence as a character vector.

**Recode a Sequence to Desired Codon**

The function RecodeDNA\_singleAA\_DesiredCodon.m will re-code an entire gene sequence (start to stop codon) where every instance of a particular amino acid is re-coded to a user defined codon.

* **RecodeDNA\_singleAA\_DesiredCodon.m**
* For implementation example, see “**Example\_Recode\_Seq\_Single\_Codon\_and\_Randomize.m**”
  + Input 1: input sequence, given as a character vector of DNA letters.
  + Input 2: target AA, given as the 3 letter amino acid abbreviation (e.g. ala) in lower case that should be re-coded.
  + Input 3: new desired codon, given as a 3 letter codon that every instance of the amino acid should be re-coded to.
  + Output: new sequence as a character vector.

**Recode sequences to a target CAI or χ, then use clustering and PCA to visualize differences**

The function RecodeTarget\_CAI.m will re-code a DNA sequence to a desired target CAI value based on a reference set of RSCU values. There are also additional inputs to constrain ENC and GC content, as well as the option to exclude re-coding specific codons. Note that the tolerance for target CAI is +/- 0.01. Note that a CAI of 0 or 1 and extremes in general cannot be practically achieved.

* **RecodeTarget\_CAI.m**
* For implementation example, see “**Example\_Recode\_Target\_CAIorCHI\_Then\_Analyze.m**”
  + Input1: input sequence as a character vector of DNA bases
  + Input2: exclusions, given as a cell array of codons.
  + Input3: CAI reference struct, given as a struct (generated using the function RSCUstruct or RSCUarray2struct) of the reference RSCU values (e.g. from a set of highly expressed genes or from χ weights).
  + Input4: CAI Target, given as a number value of the target CAI.
  + Input5: GC\_Target, given as a number value of the target GC%.
  + Input6: ENC\_Target, given as a number value of the target ENC.
  + Input7: GCdiff\_thresh, given as a number value of the permitted difference between the GC target and calculated GC%
  + Input8: ENCdiff\_thresh, given as a number value of the permitted difference between the ENC target and calculated ENC
  + Output: new sequence as a character vector, as well as the final CAI, ENC, and GC%.