DeCOOT instruction manual

Vyacheslav Tretyachenko, Vaclav Voracek

March 23, 2020

1 Introduction

DeCOOT (degenerate codon optimization tool) serves for automatic design of randomized protein libraries with given amino acid distribution. Inputs of the program involve amino acid distribution, length of resulting proteins in the library and level of degeneracy. Optional parameters are organismal codon bias, removed triplets and codon reassignment. Program outputs a set of degenerate or spiked codons which provide a protein library with desired characteristics. Principle of method lies in stochastic genetic algorithm, therefore solutions for approximations of a given distribution can differ in identities of chosen degenerate codons.

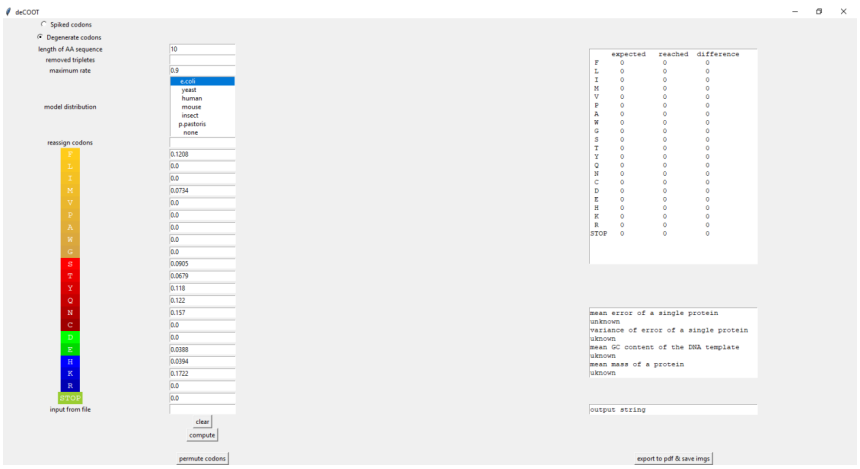

Figure 1: Main window of degenerate codon optimization tool - DeCOOT

2 Installation

DeCOOT can be installed on Linux, Mac and Windows machines. Prerequisites are python 3 and packages listed in requirements.txt. In case that you dont have Python installed we recommend to download Miniconda from https://docs.conda.io/en/latest/miniconda.html which will install Python and few useful packages. After installation:

1. open Anaconda prompt and run the command inside quotation marks "cd xxx", where xxx is the path to the directory with decoot on first use run command inside quotation marks "conda install –file requirements.txt"

2. run the command inside quotation marks "python gui.py"

If Python 3 is already installed on your computer or/and you dont want to install Miniconda. Use pip install command to install all prerequisite packages in requirements.txt file. After the installation change your working directory in terminal by command "cd xxx", where xxx is the path to the directory with DeCOOT and run DeCOOT with "python gui.py" command.

3 Setting up

3.1 Mandatory input

Obligatory inputs for DeCOOT are the length of library proteins and amino acid distributions. These attributes can be defined in "length of AA sequence" and specific amino acid brackets. Amino acid ratios do not need to be normalized to sum to 100 or 1, ratios of amino acids (e.g A 1, T 2, S 2) will be normalized to fractions of 1 automatically (A 0.2, T 0.4, S 0.4). An alternative way for input is provided by "input from file". Amino acid distribution and all other possible constraints can be defined in the file. File is put into the main deCOOT directory and its name is defined in "input from file". Example input can be found in the main DeCOOT directory as example.txt. Lines can be omitted if optional fields are not necessary.

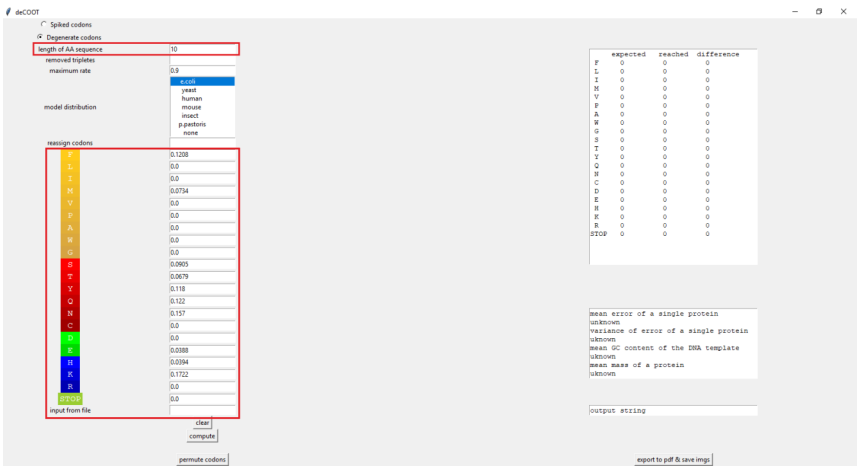

Figure 2: Most important variables for resultant degenerate libraries - length in amino acids and amino acid composition. Both variables can be entered manually or using file input

3.2 input from file

The file for reading the input must be a ".txt" file. The necessary file needs to contain amino acid distribution, which will be further normalized. If an amino acid is not present in input file, it will not appear in the output at all. Further items are optional, if they will not be present in the input file, they will be read from the GUI.

Here we provide an example content of such input file.

```
F= 1
L= 2
A=1
T=5
Q=2.5
N=0.3
D=0.12
H= 5
K=5
length=11
model_distribution=yeast
maximum_rate=0.95
removed_triplets=AAA GGG
reassigned_codons=ACT 2, GTG 3
spiked
```

The frequency of amino acid is captured by its one-letter abbreviation, followed by the equality sign and a numerical value. The first letter on a line has to be the amino-acid name and the second is the equality sign. The numerical value can be inserted in arbitrary form, as it is captured in the example code. Note that there is a "." and not a "," in those numbers.

The next six lines of the example are containing the keywords, which can be used. The keyword must be spelled exactly (even with "_" and not " ") and must be followed immediately by a "=". The only exceptions are keywords "spiked" and "degenerated", which enables/disables the usage of spiked codons.

The other keywords are followed by the same format of input as in the GUI, with the exception of model_distribution, which contains a name of one of distributions in the GUI.

If a line differs in format from those described above (even a superfluous space counts!), then is ignored and may be used for comments.

The order of lines does not matter.

3.3 Degenerate codon type

Next possible input is degenerate codon type used for optimization. DeCOOT offers approximation via degenerate (default) and spiked codons. Optimization with spiked codons might take longer time to converge, but offer higher precision in shorter library lengths or in cases where more solutions with higher degeneracy are required. In longer templates (> 15 amino acids) the difference in precision between degenerate and spiked solutions become negligible.

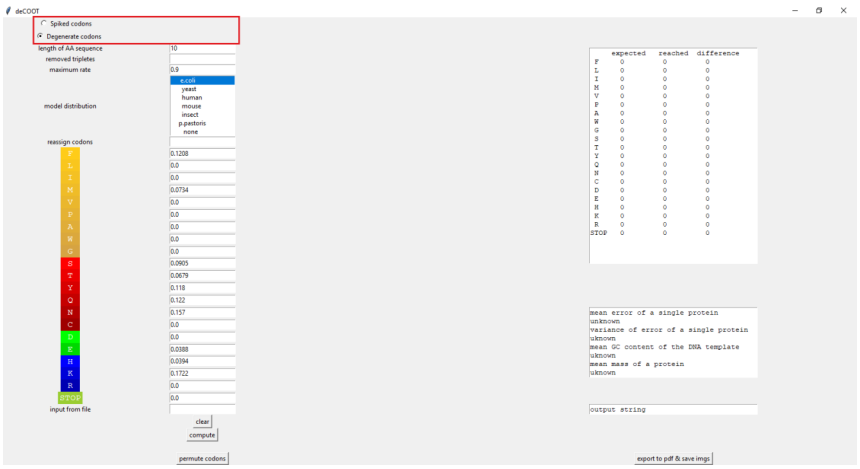

Figure 3: Standard degenerate codons with equimolar base ratios and spiked codons with variable base ratios can be chosen for distribution approximation

3.4 Codon removal and reassignment

Specific codons can be removed from the solution by entering space separated codons into “removed triplets” bracket (for example ”AAA ACT GTG”). DeCOOT will output solutions containing only those degenerate codons which do not code the removed codons. Another option is codon reassignment. If defined in the “reassign codons” in format XXX number (ATG 0.3), codon will be treated as another amino acid (and no longer as Met) with appropriate ratio and will be included into the degenerate output. There may be multiple such reassigned codons. In the input, they are separated by commas (for instance ”ACT 0.074, GTG 0.111”)

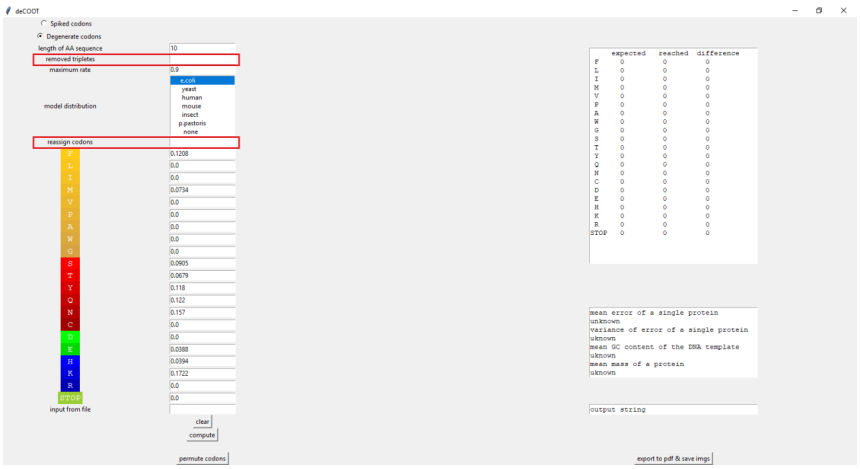

Figure 4: Certain non-degenerate codons can be removed from final degenerate solution or reassigned to encode a different amino acid

3.5 Organism codon bias

Output can be optimized according to organism codon bias, solution will be enriched in codons preferred by the selected model organism.

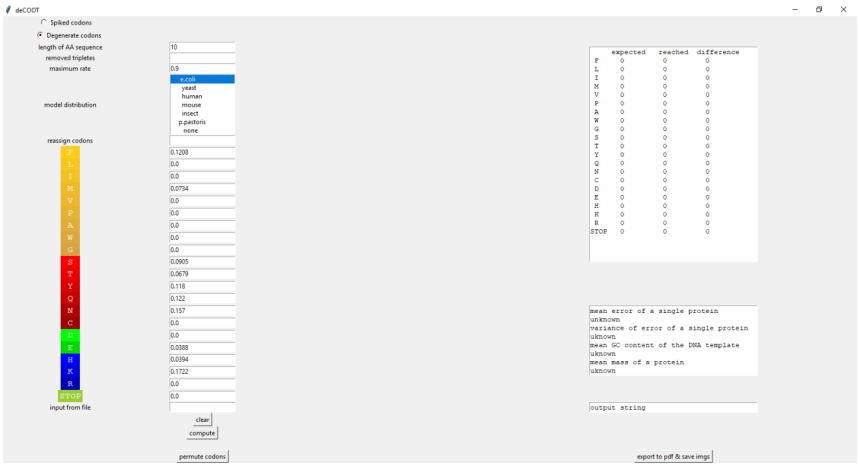

Figure 5: DeCOOT can start approximation from codon optimized seed template. This option increases the probability of organism specific codons enrichment in final degenerate solution

3.6 Degeneracy of solution

Degeneracy of solution can be tuned by "maximum rate" parameter. Rate represents maximum ratio of amino acid representable by degenerate codon (eg codon RGN codes for two arginines, two serines and four glycines, so ratios of R, S and G are 0.25, 0.25 and 0.5 respectively). Default ratio 0.9 ensures that only degenerate codons are included into the solution, maximum ratio 1 will include non-degenerate codons and lower ratios will use more degenerate codons accordingly.

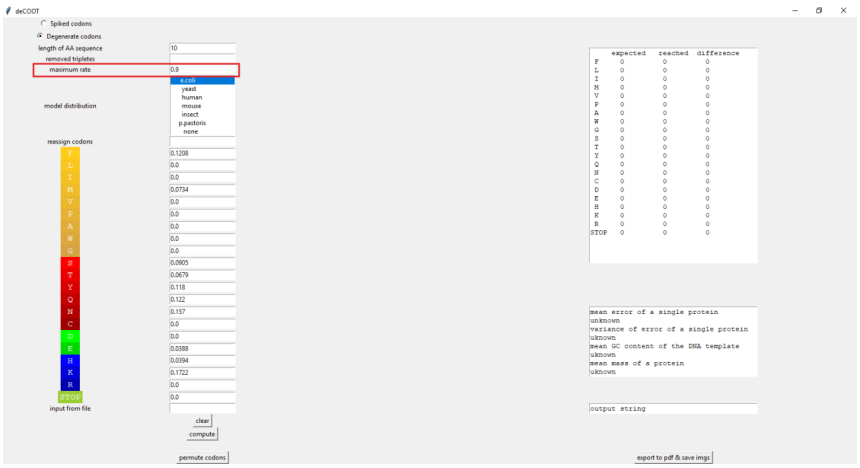

Figure 6: Degeneracy of solution can be fine-tuned, using lower values of maximum rate parameter for higher degeneracy up to 1 for lowest degeneracy where algorithm is allowed to use non-degenerate codons for approximation

3.7 Degeneracy tuning

Degeneracy of solution can be increased by limiting maximum amino acid ratio of codons to lower value. Here is the example solution where the maximum ratio was limited to 0.4 (e.g. maximum representation of a single amino acid by degenerate codon is 40 %). Pie charts provide a visual proof of increased degeneracy compared to all previous outputs with default rate of 0.9. In some cases, higher degeneracy comes with a price of lower precision as shown here. This occurs when input amino acid codons are not very similar and increased degeneracy significantly limits the codon pool for approximation. If precision is important, we recommend to increase maximum amino acid rates or use spiked codons.

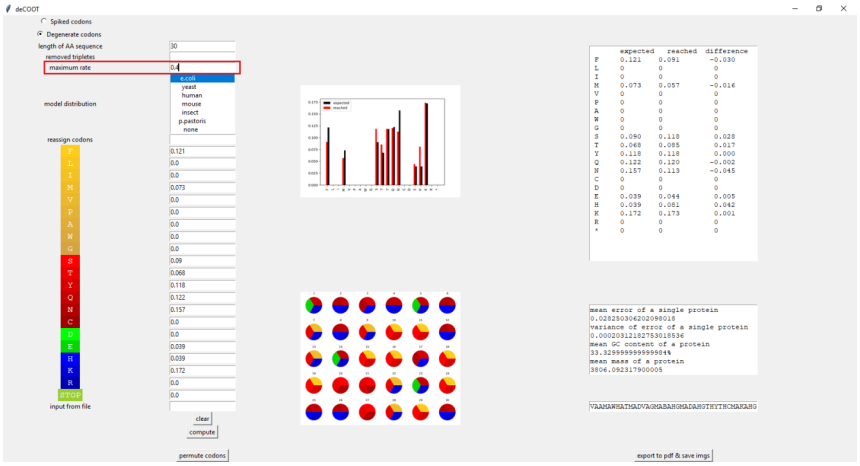

Figure 7: Degeneracy of solution can be fine-tuned, using lower values of maximum rate parameter for higher degeneracy up to 1 for lowest degeneracy wherealgorithm is allowed to use non-degenerate codons for approximation

3.8 Spiked codons

Precision of solution can be improved by inclusion of spiked codons (e.g. codons with variable nucleotide ratios) into design scheme. Here we present how spiked codons improve the precision of the previously shown solution with increased degeneracy. However, not all commercial oligonucleotide synthesis providers are able to follow the spiked codon format and feasibility of the solution as a synthesis template should be consulted individually.

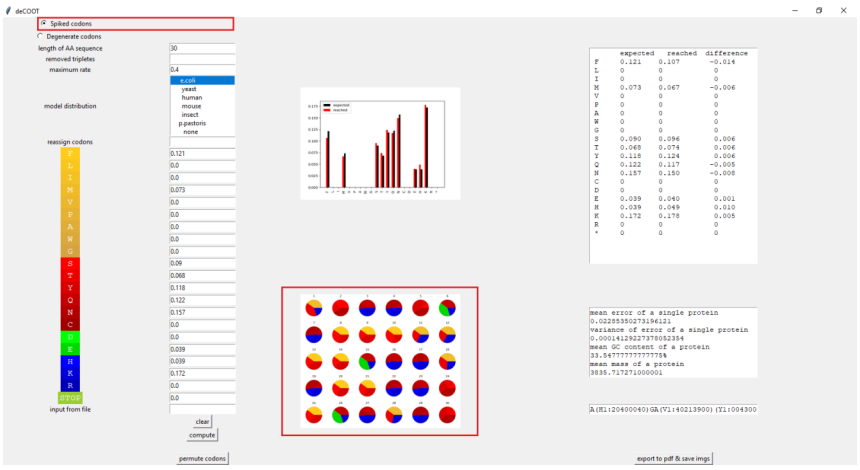

Figure 8: Utilization of spiked codons provides higher precision of optimization. Currently oligonucleotide synthesis providers allow for 4 different spiked codons

4 Output and statistics

When all inputs are entered, optimization is commenced by compute button. Script might freeze for a couple of minutes.

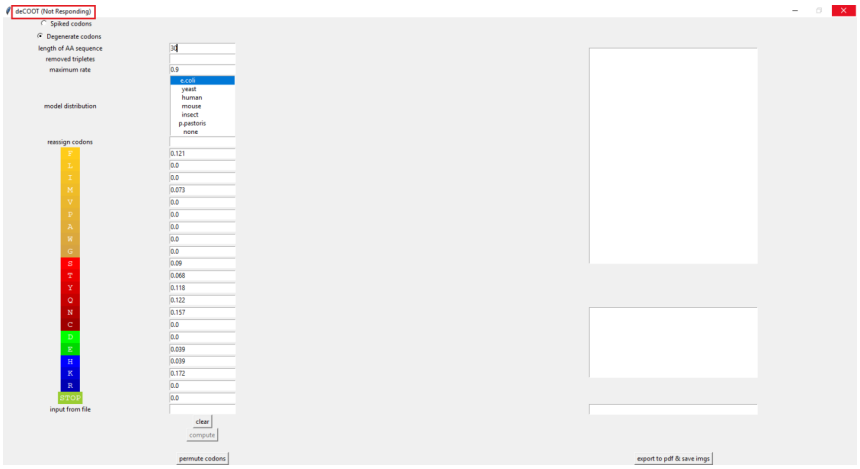

Figure 9: DeCOOT can be computationally intensive especially when large libraries and alphabets are approximated or calculation uses spiked codon option for approximation

4.1 Standard output

Output provides essential statistics on amino acid approximation and differences from input, average difference of solution from input in the means of squared error, variance protein amino acid content in the designed library, GC content of designed DNA library and mean molecular weight of protein product. Degenerate codons are represented by pie charts where slices are contents of hydrophobic (shades of yellow), polar (reds), negatively (greens) and positively charged (blues) amino acids. Numbers above the pie chart represent positions of degenerate codons in the template. Codon location is arbitrary and can be shuffled manually (without DeCOOT) or by the "permute codons" button.

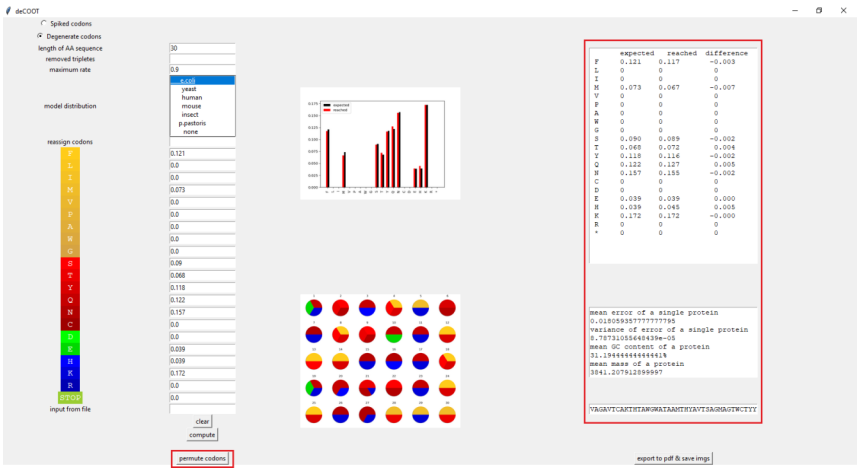

Figure 10: DeCOOT returns deviation from input distribution, GC content of DNA library and average molecular weight of protein library

4.2 PDF report

All statistics shown in the main window are exported to pdf report which contain the degenerate codon or spiked codon sequences for oligonucleotide synthesis.