# RNAmodel Quick Start Guide

## General

The program RNAmodel V2.8.3 was developed for the modeling RNA based regulation in bacteria by Monte Carlo simulation. The model description and implementation was published in [1-5] as well as examples of the results obtained with it. This document is for a local version of the model that the user can download and install in his or her computer. Online version of the program is also available at http://lab6.iitp.ru/rnamodel, but it suits better for acquaintance with the model rather than for mass experiments.

This local version of the program allows user to estimate probability of premature transcription termination *at fixed relative concentration* of the regulatory amino acid(s). Such estimate is found as the result of modeling multiple trajectories, each one finished by either termination or absence of termination (called “anti-termination”). The probability of termination characterizes the gene expression at a given concentration (single point). In order to obtain a whole plot of the termination probability *dependency* on the amino acid concentration in some range, one has to run the program several times for each value of concentration over that range. The distribution package includes scripts, which help to automate such work.

The following sections describe the distribution contents, installation and run of the program, and recommendation for its effective use. Detailed description of command line options, input and output data is currently available only in Russian.

The program is provided as an executable for x86 architecture. It is intended for a PC with Windows. RNAmodel uses a command line interface; it has to be run from a command processor of the operating system. The programming language is ! , the compiler is Microsoft Visual Studio 2008 Service Pack 1, name of the executable – rnamodel.exe. Target CPU is Intel 32-bit. The operating systems tested were Microsoft Windows XP SP3, Vista, 7, and Microsoft Windows Server 2003 SP2, 2008 R2. Using RNAmodel on other processors and/or operating systems is possible, but may require to re-compile the program or to carry out additional testing. In some cases, source code for Linux can be provided on request.

Project lead: Prof. Vassily A. Lyubetsky, Head of the Laboratory, IITP RAS (Kharkevich Institute)
http://lab6.iitp.ru/ru/contacts.html
Software developer: Dr. Lev I. Rubanov, Leading Researcher, IITP RAS (Kharkevich Institute)
E-mail: rubanov@iitp.ru

## Distribution Package

The software distribution package is a file rnamodelXXX.zip, which contains the following files:

| | |
|---|---|
| ReadmeXXX.doc | Program documentation in Russian (XXX is a version number). |
| QuickStart.doc | This file. |
| rnamodel.exe | Executable module for x86 architecture. |
| vcredist_x86.exe | Redistributable libraries by Microsoft. If your PC has not Microsoft Visual Studio 2008 SP1 installed, you should run this executable once. |
| EcE_trpE, Sden_leuA | Examples of input file with regulatory domain for *trpE* operon of *Escherichia coli* and *leuA* operon of *Shewanella denitrificans*. |
| table.bat | Example script which executes RNAmodel in the concentration range from |

| | |
|---|---|
| | 0 to 1 with step 0.05. First argument must be input file name. This script is for the first use of the program with the given sequence. |
| table2.bat | Example script which executes RNAmodel in the concentration range from 0 to 1 with step 0.05. This script is helpful for subsequent uses of the program with the given sequence so that previous results would not be overwritten. First argument must be input file name, second argument must be an unique ID of the experiment e.g. _1, _2, etc. |
| test.bat | Script to run the program with the above examples. This script invokes the scripts table.bat, table2.bat as well. |
| *.log | Output logs generated by RNAmodel when called for multiple runs and several concentration values. |
| *.txt | Processed multiple run logs for import into Excel. |
| *.out | Output logs generated by RNAmodel when called for single run. |
| *.html | Trajectory files obtained after single run. |
| traject.css | Style sheet for viewing the trajectory files with Web browser. |

## Installation and Checking

1. Create a folder e.g. D:\RNAmodel, and extract the distribution file contents into that folder.
2. If Microsoft Visual Studio 2008 Service Pack 1 has not been installed on the PC, run vcredist_x86.exe from the created folder and follow instructions on the screen.
3. Run Windows command processor (cmd.exe) and switch to the program folder:

   ```
   d:
   cd d:\twobox
   ```

   For your convenience, we suggest to create a shortcut for the command processor on desktop, and then substitute the working directory in the shortcut properties with D:\RNAmodel (or other name you chose at Step 1). After that you will not need to type the above commands each time you use the program.
4. To check the program operation, enter the command rnamodel without parameters. A help on the program arguments will output in the command processor window.
5. To verify the program operation using the script, enter the command test . Sample output files from the distribution will be overwritten, so you should rename or copy them somewhere prior to enter the command. Depending on the PC performance, the entire script processing can take 3-5 minutes or more.
6. Once the script complete, compare result files (*.html, *.log, *.out, *.txt) produced with those from the distribution. Due to different performance and various random factors, the files may not be identical, but the results have to be similar.
7. After that you can prepare source data and scripts for your actual experiments.

## How to use RNAmodel effectively

1. *Source sequence preparation*

   Proper cut of the source sequence from entire leader region of the gene is of key importance for successful modeling. We recommend beginning the sequence from the start codon of the leader peptide gene. Doing so, we ensure correct reading frame and initial conditions consistent with default parameters. Otherwise, you have to specify start of translation position with `-sr` option.

The sequence should be as short as possible because the number of possible states of RNA secondary structure grows quickly as the sequence length increases. However, the sequence must include entire terminator hairpin and U-run followed it. There is no sense in saving more than 15–20 nt after the terminator, because it does not affect RNA polymerase at larger distances.

If terminator/antiterminator is unknown for the sequence, and/or U-run is unobvious, we recommend to run the model with non-cut sequence first, using options `-lmax0 -un10 -o2,` in order to get full list of putative helices and U-runs found by default. You may cancel the program just after start because that list is all you need from the run. Reviewing the list, you can identify mutually exclusive terminator and antiterminator, and choose suitable method and parameters of U-run selection (see item 3 about that). Then you can cut up the sequence as said above.

2. *Maximum size of helix loop*

   The shorter loops allowed, the less putative helices exist in the given sequence, and the faster model works. However, one should remember that the stem of a long hairpin is actually a helix with a long loop. Therefore, user must check whether a stem of antiterminator meet the maximum loop size set by `-lmax` option. Default value 50 is a reasonable compromise, but we met situations requiring `–lmax70` and even `–lmax100`. If the secondary structure is unknown, first proceed as recommended in item 1 (`-lmax0`), then set actual limit.

3. *U-run selection options*

   Exact mechanism of the polymerase slippage off the DNA strand is still unknown. Our model assumes that polymerase can slip only if the transcription point belongs to the U-rich region (U-run for short), and on the condition that a hairpin exists nearby, which affects the polymerase strongly enough.

   Therefore, the presence of U-run nearby the terminator is a necessary condition for the classic attenuation regulation to exist in our model. Failing this condition, the modeling result will be knowingly negative. Thus, prior to run the program for a sequence, the user should ensure that U-run exists. Since there is no conventional definition of U-run, the model provides for two methods. Each one is based on different U-run definition and has own set of parameters. User should choose himself a suitable option for the sequence of interest, and set parameters of that option.

   **First method** uses the following implicit definition of U-run. If there is a region within the sequence that consists of $l$ e $l_{min}$ letters, of which at least $f_{min} \cdot l$ are the letters U/T, then *all letters* of that region are considered belonging to the U-run. Minimum length of U-run, $l_{min}$, and fraction of U's, $f_{min}$, are the program parameters, which are set by the command line options `-lura` and `-u`, respectively. By default, $l_{min} = 5$, $f_{min} = 0.8$.

   The result of such definition is that each letter of the sequence is identified as either belonging or not belonging to the U-run. So the sequence is split into connected components of letters belonging to the U-run; each component is the U-run sought-for. There can be multiple U-runs in a sequence, and the program has an option `-un` allowing to ignore all U-runs but several last ones (by default, only the very last U-run is taken into account). User should cut 3'-end of the sequence so that the last U-run is the nearest one to the terminator, or specify `-un` option.

**Second method** uses the explicit definition of U-run. The U-run is as long as possible segment of the sequence that includes at least $n_{\min}$ letters U/T, such that neighboring U's are separated by $g_{\max}$ or less non-U letters. Also included in the U-run are ($g_{\max}/2$) non-U letters at each end of it. Parameters $n_{\min}$ and $g_{\max}$ are set by command line options `-u` and `-ug`, respectively. Thus, the `-u` option of the program allow user to choose the method desired, because the value of this parameter is less or equal to 1 in the first method, and is greater than 1 in the second method. Default gap length $g_{\max} = 3$, and we recommend to set $n_{\min}$ e 3.

The second method can also result in multiple U-run selection. The user can limit the number of them by `-un` option, like in the first method.

It is difficult to give unambiguous recommendations for selection suitable method. Generally speaking, if an U-run is more similar to the classic one, i.e. consists of many U's with short gaps between them, then the first method will be better. Conversely, if U-run comprises only few U's with longer gaps, then the second method will do. Anyway, once the method and parameters are chosen, we recommend to run the program once with `-o2` option, in order to ensure the U-run is found properly (recall that U-runs are marked in the helix list header).

Another important consideration is that U-run should not reach a loop of the terminator so that the antiterminator hairpin might not slip off the polymerase. Below we provide several examples of the 3'-ends of real sequences, starting from the right shoulder of the terminator, along with suitable option sets. In these examples, the terminator positions are underlined, and U-runs are shaded.

| | |
|---|---|
| GAAGCGGGCUUUUUUGUUUCUAGCUCUUA | by default |
| GAAGCGGGCUUUUUUGUUUCUAGCUCUUA | -u3 |
| GGAUGCGGAGGCUUCCCUCUCUCAUC | -lura10 -u0.5 |
| GGAUGCGGAGGCUUCCCUCUCUCAUC | -u3 |
| GGAGGCUUUUUUUGUACCUG | -lura13 -u.69 |
| GGAGGCUUUUUUCGUAUAUGGAUUC | -lura18 -u.61 |
| AUGGGGGGCUUUUUUAUUUGUAGUUAUUUGUAUUAGUAAUCGA | -un2 |
| AUGGGGGGCUUUUUUAUUUGUAGUUAUUUGUAUUAGUAAUCGA | -lura25 -u.7 |
| AUGGGGGGCUUUUUUAUUUGUAGUUAUUUGUAUUAGUAAUCGA | -u3 |
| GAGCGGGUUUUUUAUUGCCGUUU | -u3 -ug4 |
| AGUCCGGGGGUUUUUUUUACAACUA | -u3 -ug5 |
| GGCCGCCAUCCGCUAGA | -u2 -ug4 |
| GGUCGGUUGCUUUACUUA | -u3 -ug2 |
| CAGGAGGGCCGUACC | -u1 -ug6 |

4. *The number of trajectories*

   For a quick check, whether classical attenuation regulation exists in a sequence, user can run the program with `-z100` option (i.e. at 100 trajectories), but this will give very poor estimate of the termination probability, and can easily lead to a false conclusion. Therefore, we recommend making any conclusion after at least 1000 modeling trajectories for each concentration.

5. The following limits are set in the current version of RNA model:
   - length of the input file name including the path (if present): < 128 characters
   - length of the sequence: < 300 nt
   - number of non-extensible helices in the sequence: < 500
   - number of paired nucleotides within a hairpin: < 64 pairs

- total size of loops and bulges within a hairpin: < 256 nt
- number of hypohelices in a macrostate: < 16

## References

1. V. Lyubetsky, L. Rubanov, A. Seliverstov, S. Pirogov. Model of gene expression regulation in bacteria via formation of RNA secondary structures. *Molecular Biology*, 2006, V. 40, No. 3, p. 440-453.
2. V. Lyubetsky, K. Gorbunov, S. Pirogov, L. Rubanov, A. Seliverstov. An algorithm and search results for a model of gene expression regulation with RNA secondary structures in bacteria. *Information Processes*, 2005, V. 5, No. 5, p. 337-366. http://www.jip.ru/2005/337-366.pdf. (in Russian)
3. V. Lyubetsky, A. Seliverstov. Computation of regulation efficiency of tryptophan biosynthesis in bacteria based on a model of classic attenuation. *Information Processes*, 2006, V. 6, No. 1, p. 55-57. http://www.jip.ru/2006/55-57-2006.pdf. (in Russian)
4. Lyubetsky V., Pirogov S., Rubanov L., Seliverstov A. Modeling Classic Attenuation Regulation of Gene Expression in Bacteria. *Journal of Bioinformatics and Computational Biology*. Vol.5, 1, 2007, p. 155-180.
5. L. Rubanov and V. Lyubetsky. RNAmodel Web Server: Modeling Classic Attenuation in Bacteria. *In Silico Biology*. Vol.7, 3, 2007, p. 285-308.
6. A. Xayaphoummine, T. Bucher, H. Isambert, “Kinefold web server for RNA/DNA folding path and structure prediction including pseudoknots and knots”, *Nucleic Acids Res*. **33** (Web Server issue), W605–10 (2005).