Bioinformatics, 36(9), 2020, 2920–2922
doi: 10.1093/bioinformatics/btaa039
Advance Access Publication Date: 23 January 2020
Applications Note

Structural bioinformatics

incaRNAfbinv 2.0: a webserver and software with motif control for fragment-based design of RNAs

Matan Drory Retwitzer[1,*], Vladimir Reinharz [2,3], Alexander Churkin[4], Yann Ponty[5], Jérôme Waldispühl[6] and Danny Barash[1,*]

[1]Department of Computer Science, Ben Gurion University of the Negev, Beer Sheva 84105, Israel, [2]Department of Computer Science, Université du Québec à Montréal, Montreal, H2X 3Y7, Canada, [3]Institute for Basic Science, Daejeon 34126, South Korea, [4]Software Engineering Department, Sami Shamoon College of Engineering, Beer-Sheva 84100, Israel, [5]Laboratoire d'Informatique de l'École Polytechnique (LIX CNRS UMR 7161), Ecole Polytechnique, Palaiseau 91120, France and [6]School of Computer Science, McGill University Montréal H3A 0E9, Canada

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on May 27, 2019; revised on November 25, 2019; editorial decision on December 23, 2019; accepted on January 15, 2020

Abstract

Summary: RNA design has conceptually evolved from the inverse RNA folding problem. In the classical inverse RNA problem, the user inputs an RNA secondary structure and receives an output RNA sequence that folds into it. Although modern RNA design methods are based on the same principle, a finer control over the resulting sequences is sought. As an important example, a substantial number of non-coding RNA families show high preservation in specific regions, while being more flexible in others and this information should be utilized in the design. By using the additional information, RNA design tools can help solve problems of practical interest in the growing fields of synthetic biology and nanotechnology. `incaRNAfbinv 2.0` utilizes a fragment-based approach, enabling a control of specific RNA secondary structure motifs. The new version allows significantly more control over the general RNA shape, and also allows to express specific restrictions over each motif separately, in addition to other advanced features.

Availability and implementation: `incaRNAfbinv 2.0` is available through a standalone package and a web-server at https://www.cs.bgu.ac.il/incaRNAfbinv. Source code, command-line and GUI wrappers can be found at https://github.com/matandro/RNAsfbinv.

Contact: matandro@cs.bgu.ac.il or dbarash@cs.bgu.ac.il

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

`incaRNAfbinv 2.0` (`incaRNAtion` with RNA fragment-based inverse) is a standalone python package and a webserver for RNA design. It allows its user to specify an RNA secondary structure and design sequences based on user-defined constraints under an inverse folding paradigm. Inverse folding has been used to design RNA sequences for a variety of applications. A recent review can be found in Churkin *et al.* (2018), which also highlights the use of sequence design to search for novel non-coding RNAs (ncRNAs) using a structure-based methodology (Dotu *et al.*, 2013; Retwitzer *et al.*, 2015).

Inverse folding problems were addressed using a variety of approaches. Initial approaches, following `RNAinverse` (Hofacker *et al.*, 1994), used adaptive random walk to minimize base pair distance as a primary target while more recent proposals, such as `RNAiFold` (Garcia-Martin *et al.*, 2015) and `antaRNA` (Kleinkauf *et al.*, 2015) use constraint programming and ant colony optimization, respectively. A common feature of those programs is a focus on designing sequences that minimize the base pair distance to the target structure, computed using available tools, such as `RNAfold` (Hofacker, 2003) or `mfold` (Zuker, 2003). Other approaches like `NUPACK` (Zadeh *et al.*, 2011) that utilizes minimization of the ensemble defect have also been tried. One of the sub-topics that has gained recent interest is that of multiple target RNA design (Hammer *et al.*, 2017, 2019; Lyngsö *et al.*, 2012; Siederdissen *et al.*, 2013; Taneda, 2015).

The first `RNAfbinv` program (Weinbrand *et al.*, 2013) focused on fragment-based design, while base pair distance was used as a secondary target. It used simulated annealing to design a sequence matching a general shape and additional parameters such as the target energy. The input dot bracket structure representation was transformed into a Shapiro string representation (Shapiro, 1988), equivalent to a coarse grained tree-graph. The Shapiro

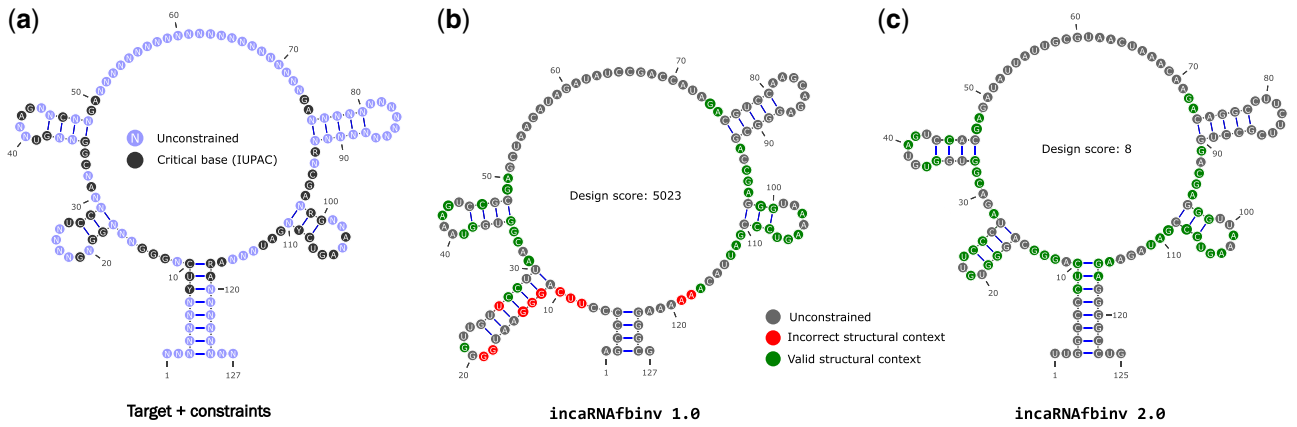

Fig. 1. Illustration of the differences between `incaRNAfbinv 1.0` and `incaRNAfbinv 2.0` for the FMN riboswitch aptamer. Design scores appear in the center of the illustration. (**a**) Target sequence and structure. Critical nucleic acid bases indicated in black. (**b**) Typical design output of `incaRNAfbinv 1.0`. Sequence constraints were satisfied but not always properly in the correct structural context (red color). The best alignment fails to match the U in the 5' of initial stem, the A in the 3' of the initial stem and the triple G in the 5' most segment of the multi loop summing up to a penalty of 5000. The rest is due to insertions and size differences of various motifs. (**c**) Typical design output of `incaRNAfbinv 2.0` (note that the designed sequence is 2nt-shorter than the input). Sequence constraints are not only satisfied, but also shifted to match their original structural context. These small shifts in sizes amass a penalty of 8. The figure was generated by VARNA (Darty *et al.*, 2009)

representation depicted the general shape of the structure by listing the motifs and their relation to each other. The Shapiro string comparison allowed `RNAfbinv` to reach the general shape as intended, yet ignored other features of the motifs. Nevertheless, design cases require the preservation of specific nucleic acid bases within a structural context. Although sequence preservation was available as a feature in `RNAfbinv` (Retwitzer *et al.*, 2016; Weinbrand *et al.*, 2013), it was disconnected from the fragment design. This allowed for designs where critical nucleic acid bases matched their specified position, yet would sometimes be shifted to an unintended fragment, as shown in Figure 1.

2 Methodology

`incaRNAfbinv 2.0` starts with the `incaRNAtion` weighted sampling approach (Reinharz *et al.*, 2013), followed by a four-step look-ahead simulated annealing process. Although superficially similar to the strategy implemented by the initial `RNAfbinv` program, the content of each step has considerably evolved in both the general objective function and the comparison method. This allows for a correct design procedure in which constraints are added within a given structural context, a key feature for many functional RNAs.

Indeed, in each step, the current candidate sequence and predicted structure are compared with the target sequence structure, using a tree comparison. The sequence is folded using `RNAfold` (Lorenz *et al.*, 2011), and the structure is then converted to its tree form, where each node represents a single motif. The nodes also hold sequence information relevant to the motif (Supplementary Fig. S1). The designed candidate tree is then compared with the target tree.

In RNAfbinv 1.0, the objective function's main component is the number of mismatched motifs between the candidate and target sequences. In the new version, the target structure contains two types of nodes: bounded motifs (Stems) or unbounded motifs (Loops, Bulges or Exterior region), with only motifs of the same type being comparable. Matching two nodes leads to a score being calculated, based on global sequence alignment between the sequences attached to the node. Sequence alignment is done per sequence segment by order. For example, an interior-loop has two unpaired segments, while a bulge motif only has one. When the two nodes are matched, only one of the interior-loop segments will be aligned, while the other will be treated as deleted. By default, deleting an 'N' nucleotide from the target motif will result in a small penalty, while deletion of any other IUPAC symbols introduces a very large penalty. An optional feature enables the use of sequence motifs by writing consecutive lower case IUPAC symbols in the target sequence. Those motifs only exist within their structural context. Insertion and deletions inside such regions will result in a slightly larger penalty.

For target tree T and candidate tree C the design score is:

$$\begin{aligned}\text{ObjectiveScore}(T,C) = \text{TreeAlign}(T,C) + |\Delta G_T - \Delta G_C| \\ + |\text{neutrality}_T - \text{neutrality}_C| \times 100\end{aligned}$$

The $\text{TreeAlign}(T,C)$ score is computed by dynamic programming as

$$\min \begin{cases} \text{SeqAlign}(T,C) + \text{CombineChildren}(T,C) \\ \min\limits_{u \in \text{children}(T)} \text{TreeAlign}(u,C) + \text{Del}(T) + \sum_{w \in \text{children}(T),\, w \neq u} \delta(w) \\ \min\limits_{v \in \text{children}(C)} \text{TreeAlign}(T,v) + \text{Del}(C) + \sum_{w \in \text{children}(C),\, w \neq v} \delta(w) \end{cases}$$

$\text{CombineChildren}(T,C)$ finds the best forest alignment between the children of T and C while maintaining order. $\text{SeqAlign}(T,C)$ is the global alignment score of the sequences in nodes T and C. The cost $\text{Del}(C)$ (resp. $\delta(w)$) of removing a node C (resp. subtree w) includes a penalty for the removal of the attached sequences, a fixed penalty for each removed motifs and an additional penalty for removing a target motif marked as preserved. A detailed description of the dynamic programming algorithm including extended formulas (Supplementary Figs S2–S4) and exact penalty values (Supplementary Table S1) are available in the Supplementary Material. The penalties reflect the importance of different features within the design such that sequence constraints are a top priority and motif-based structure is a close second. Additional features, such as base pair distance, free energy and mutational robustness, have the least effect on the design.

Motif-based comparison also allows for variable length results as in Figure 1, in which case designed sequences might be shorter or longer than the original input. The maximum length difference can be set by the user. Eventually the alignment score is combined with additional features previously supported by `incaRNAfbinv 1.0`.

3 Conclusion

`incaRNAfbinv 2.0` allows for the design of sequences that match the shape of a target secondary structure while applying sequence constraints in their proper structural context, which was not possible in the initial version. An example is presented using the FMN riboswitch aptamer (Fig. 1), where some of the preserved nucleic acids are critical for ligand binding within the multi-loop and

hairpin loops. These constrains are properly handled with `incaRNAfbinv 2.0`, thus making our webserver significantly better than before for designing functional ncRNAs.

Funding

This work was supported by a Joint Research Projects grant from the Israeli Ministry of Science & Technology (MOST) and the French *Centre National de la Recherche Scientifique* (CNRS).

Conflict of Interest: none declared.

References

Churkin,A. *et al*. (2018) Design of RNAs: comparing programs for inverse RNA folding. *Brief. Bioinform*., **19**, 350–358.

Darty,K. *et al*. (2009) VARNA: interactive drawing and editing of the RNA secondary structure. *Bioinformatics*, **25**, 1974–1975.

Dotu,I. *et al*. (2013) Using RNA inverse folding to identify IRES-like structural subdomains. *RNA Biol*., **10**, 1842–1852.

Garcia-Martin,J.A. *et al*. (2015) RNAiFold 2.0: a web server and software to design custom and Rfam-based RNA molecules. *Nucleic Acids Res*., **43**, W513–W521.

Hammer,S. *et al*. (2017) RNAblueprint: flexible multiple target nucleic acids sequence design. *Bioinformatics*, **33**, 2850–2858.

Hammer,S. *et al*. (2019) Fixed-parameter tractable sampling for RNA design with multiple target structures. *BMC Bioinformatics*, **20**, 209.

Hofacker,I.L. *et al*. (1994) Fast folding and comparison of RNA secondary structures. *Monatsh. Chem*., **125**, 167–188.

Hofacker,I.L. (2003) Vienna RNA secondary structure server. *Nucleic Acids Res*., **31**, 3429–3431.

Kleinkauf,R. *et al*. (2015) AntaRNA: ant colony based RNA sequence-design. *Bioinformatics*, **31**, 3114–3121.

Lorenz,R. *et al*. (2011) ViennaRNA Package 2.0. *Algorithm. Mol. Biol*., **6**, 26.

Lyngsö,R.B. *et al*. (2012) Frnakenstein: multiple target inverse RNA folding. *BMC Bioinformatics*, **13**, 260.

Reinharz,V. *et al*. (2013) A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotide distribution. *Bioinformatics*, **29**, i308–i315.

Retwitzer,M.D. *et al*. (2015) An efficient minimum free energy structure-based search method for riboswitch identification based on inverse RNA folding. *PLoS One*, **10**, e0134262.

Retwitzer,M.D. *et al*. (2016) IncaRNAfbinv: a webserver for the fragment-based design of RNA sequences. *Nucleic Acids Res*., **44**, W308–W314.

Shapiro,B.A. (1988) An algorithm for comparing multiple RNA secondary structures. *Bioinformatics*, **4**, 387–393.

Siederdissen,H. *et al*. (2013) Computational design of RNAs with complex energy landscapes. *Bioploymers*, **99**, 1124–1136.

Taneda,A. (2015) A multiple-objective optimization for RNA design with multiple target secondary structures. *BMC Bioinformatics*, **16**, 280.

Weinbrand,L. *et al*. (2013) RNAfbinv: an interactive Java application for fragment-based design of RNA sequences. *Bioinformatics*, **29**, 2938–2940.

Zadeh,J.N. *et al*. (2011) Nucleic acid sequence design via efficient ensemble defect optimization. *J. Comp. Chem*., **32**, 439–452.

Zuker,M. (2003) Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*., **31**, 3406–3415.