

Methods in
Molecular Biology 2586

Springer Protocols

Risa Karakida Kawaguchi
Junichi Iwakiri *Editors*



RNA Structure Prediction

 Humana Press

METHODS IN MOLECULAR BIOLOGY

Series Editor

John M. Walker

School of Life and Medical Sciences

University of Hertfordshire, Hatfield

Hertfordshire, UK

For further volumes:

<http://www.springer.com/series/7651>

For over 35 years, biological scientists have come to rely on the research protocols and methodologies in the critically acclaimed *Methods in Molecular Biology* series. The series was the first to introduce the step-by-step protocols approach that has become the standard in all biomedical protocol publishing. Each protocol is provided in readily-reproducible step-by-step fashion, opening with an introductory overview, a list of the materials and reagents needed to complete the experiment, and followed by a detailed procedure that is supported with a helpful notes section offering tips and tricks of the trade as well as troubleshooting advice. These hallmark features were introduced by series editor Dr. John Walker and constitute the key ingredient in each and every volume of the *Methods in Molecular Biology* series. Tested and trusted, comprehensive and reliable, all protocols from the series are indexed in PubMed.

RNA Structure Prediction

Edited by

Risa Karakida Kawaguchi

Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA

Junichi Iwakiri

*Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences,
The University of Tokyo, Chiba, Japan*

 **Humana Press**

Editors

Risa Karakida Kawaguchi
Cold Spring Harbor Laboratory
Cold Spring Harbor, NY, USA

Junichi Iwakiri
Department of Computational Biology
and Medical Sciences
Graduate School of Frontier Sciences
The University of Tokyo
Chiba, Japan

ISSN 1064-3745

Methods in Molecular Biology

ISBN 978-1-0716-2767-9

<https://doi.org/10.1007/978-1-0716-2768-6>

ISSN 1940-6029 (electronic)

ISBN 978-1-0716-2768-6 (eBook)

© Springer Science+Business Media, LLC, part of Springer Nature 2023

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Humana imprint is published by the registered company Springer Science+Business Media, LLC, part of Springer Nature.

The registered company address is: 1 New York Plaza, New York, NY 10004, U.S.A.

Preface

Since the 1960s, RNA secondary structure analysis has been performed for a wide variety of groups of RNAs, including rRNAs and RNA virus genomes [1], and this has been followed by RNA 3D structure analysis [2]. Owing to the instability of RNA structures, the number of the detected structures is much smaller for RNAs than for proteins, specifically 1523 RNA-only structures compared with 151,577 protein-only structures in the Protein Data Bank (PDB) [3] as of 2020. In silico structure prediction is, therefore, a powerful approach to overcoming these limitations and revealing a comprehensive view of RNA structures. The most prevalent and high-performance structure prediction method is based on a thermodynamic model that takes the primary sequence as an input and predicts a representative structure according to its stability. Free energy changes are estimated as a metric of structure stability by approximation of those for each sub-structure using the melting temperature for short oligos or based on structures in databases [4–6]. To compute the probability of RNA secondary structures based on thermodynamic models, it is necessary to compute the sum of the exponential of the free energy changes. This is formulated as $\sum_{\zeta \in \Omega(x)} e^{dG(\zeta, x)/RT}$, where

$dG(\zeta, x)$ is the free energy change of sequence x and structure ζ , R is the gas constant, T is the absolute temperature, and $\Omega(x)$ represents the set of all possible structures of sequence x . $dG(\zeta, x)$ for every possible subsequence x can be efficiently computed by a dynamic programming (DP) technique using the subproblems $dG(\zeta, x')$, where x' is a subsequence of x . After computing the DP variables obtained for the full sequence, the partition function and probability of the structure can be computed following a canonical distribution. Previous benchmark studies have carried out the comprehensive assessment of RNA secondary and 3D prediction tools with multiple structure databases [7, 8], indicating not only sufficiently high prediction accuracy but also computational limitation in robustness and performance stability for many prediction methods.

In terms of accuracy and computational time, one of the main limitations of in silico structure analysis is the prediction of long RNAs. For example, the estimation of free energy changes is assumed to show low accuracy for distant base pairs because models are fitted to results obtained from short RNAs. The rapid exponential increase in possible structures relative to sequence length is also a critical problem for feasibility. While the improvement of applicability and performance has been actively studied for in silico structure analysis [9], experimental techniques have also been applied to solve those problems at the same time, for example, high-throughput structure probing methods including SHAPE-seq [10] and DMS-seq [11]. These methods utilize high-throughput sequencing to capture cleavage or modification events introduced by chemical reagents or enzymes as a signature of reactive sites regardless of the length or condition of RNA samples. This is based on the principle that the probability of these events is correlated with the probability of being unpaired in RNA secondary structures. By comparing the determined and predicted accessibilities, both scores have been found to be consistent at a transcriptome-wide level [9]. Owing to the various sources of biological and technical noise, there is a challenge in extracting reproducible results from high-throughput data including such noise [12, 13]. However, the integration of multiple datasets relying on different techniques may uncover the true transcriptome-wide RNA structure landscape by overcoming data-specific biases [14].

Moreover, high-throughput structure probing methods can be applied to a variety of conformational analyses. A careful selection of reagents and experimental settings can further enhance the comprehensiveness of detectability of base reactivity within a specific time range [15]. For the detection of the co-accessibility of multiple bases, a structure probing method variant called mutational profiling has been developed (e.g., SHAPE-MaP [16] and DMS-MaP [17]). In such methods, the mutations on the same read are used as a clue to determine which structure or combination of structures is observed *in vivo* [18, 19]. The secondary structure is known to spontaneously fold in a manner deeply dependent on the primary sequence but to be perturbed by triggers that can include other molecules' binding, modification, and different cell states (e.g., temperature, pH, or metal ion; [11, 19, 20]). Analyzing the inconsistency between the predicted structures and experimentally determined reactivities is, therefore, an efficient approach to discover the existence of *in vivo*-specific regulation that causes structure disruption. Recently, a probing method that captures the folding of growing nascent RNA during transcription has been developed [21, 22]. Because RNA secondary structure is formed simultaneously during transcription [23], the structure comparison between nascent and matured RNAs potentially reveals the formation of a structure differing from the initial or stable form, suggesting that the influence of thermodynamic fluctuations may affect RNA folding kinetics.

Compared with RNA secondary structure prediction, RNA 3D structure prediction is recognized as a more challenging problem due to the more numerous degrees of freedom for 3D structures, such as distances and angles of each base pair, resulting in high computational demand. Despite such difficulties, improvements in RNA structure prediction have been attempted for 3D structures using information from experimentally validated structures via X-crystallography and cryo-EM. The accumulation of RNA 3D structure information is also expected to help clarify the kinetic mechanisms of RNA and protein binding [24].

For both RNA secondary structure and 3D structure analyses, machine learning has focused on combining classical structure prediction methods with the vast amounts of experimental structure data recently obtained. One successful example has utilized determined RNA structures in order to improve prediction accuracy of existing RNA secondary structure prediction methods. While some studies have applied such data to improving the parameters in thermodynamic models [25–27], another strategy cooperatively improves prediction models by aligning the predicted base pairs and detected accessibilities to be consistent [28]. The application of deep learning models has enabled the implementation of a new strategy to tackle complex problems, such as high-resource demands or high-dimensionality input features. For example, a deep neural network model achieved a dramatic advancement of binding motif identification required for RNA interaction determination. By considering the higher-order influences of base combinations for RNA and protein binding, this model enables accurate binding site prediction as well as a deeper elucidation of RNA binding regulation mechanisms [29]. As such, machine learning methods, accumulated data, and their resulting accurate structure predictions have a strong synergistic effect in moving the field of RNA biology forward by expanding the targets of RNA structure analysis, from short non-coding RNAs (ncRNAs) to long or coding RNAs, with a variety of goals.

Organization of This Book

In this series, we introduce recent progress in RNA structure prediction and its application from a broad viewpoint, particularly over the last several years. Here, we introduce the topics covered by 16 chapters and discuss some additional topics that are closely related to RNA structure prediction, such as RNA inverse folding. Because of advancements in experimental protocols and devices (e.g., nanopore sequencing [30]), the integration of new types of data as well as new analysis techniques is necessary. Hence, the variety of topics contained in this series is hoped to serve as a simple guide for both experimental and computational RNA researchers.

RNA Secondary Structure Prediction

RNA secondary structure prediction is a problem that involves predicting the combinations of base pairing between complementary bases, namely A with U and G with C, as well as for G–U wobble pairs. The gold-standard method for RNA secondary structure prediction is a thermodynamic model in which the free energy change of the structure is approximated by the experimentally obtained parameters for each partial structure. The highly stable structure is then selected as the best prediction result that takes into account the landscape of all possible structures through use of a different metric, such as minimum free energy (MFE), maximum expected accuracy, and centroid structure computation [31]. Because of the enormous number of possible base pairs, reliable secondary structure prediction is obtained in exchange for the considerable computation time required to find stable structures whose free energy changes are substantially small. For example, one can assume that each base of a sequence of 100 nucleotides is randomly sampled from four types of bases. If a base can bind to its complementary base at any position, each pair can form a base pair with a probability of $6/16 = 0.375$. Given this probability, the order of all possible structures for a sequence of 100 bases can reach roughly 10^{55} . In a thermodynamic model, however, the probability of the unstable structures or base pairs is low enough that they rarely affect the results of the structure prediction. One strategy that ignores those low-probability structures can reduce the computation area to be explored while maintaining the precision of structure prediction.

An efficient enumeration of all possible structures can be conducted by DP. The computational time of a DP algorithm depends on the sequence length but can be varied depending on the structure types to be included. For example, pseudoknot structures consist of base pairs crossing over each other. The computational time of DP-based structure prediction with pseudoknots is $O(N^6)$ while the computational time for structures without pseudoknots is $O(N^3)$, where N is the sequence length [32]. This makes it difficult to analyze RNAs longer than several tens of nucleotides, although pseudoknots can function as a key motif for biological regulation [33]. Kimchi et al. (2019) tackled this problem by developing *LandscapeFold*, which enumerates structures including pseudoknots based on a polymer physics model [34]. It can predict the MFE structure as well as the distribution of free energy changes for all possible structures rather than utilizing a DP algorithm based on thermodynamic models. The structure analysis of *LandscapeFold* can be a powerful approach to analyze the structures of functional short RNAs, such as ligands or aptamers. In addition to the prediction of structures including pseudoknots, another problem is that an appropriate metric is required to compare multiple RNA structures. The novel tool *planeGraph2tree* clusters RNA structures with pseudoknots based on the PEELING

algorithm [35]. Its input is a plane graph obtained from each RNA secondary structure. `planeGraph2tree` identifies a topological centroid of the graph and constructs a topological centroid tree. The distance between two structures is then obtained by the minimum cost of editing operations.

For the structure analysis of long RNAs, even the prediction of pseudoknot-free structures is infeasible because of the computational time and precision problem. Computation of local structures instead of global structures is a solution to accelerate DP computation. `Rfold` is a model in which a maximum constraint is set for the base pair distance in order to analyze the local structure stability of RNA [36]. The model can be applied for the computation of a variety of structure metrics: stem probability, accessibility, and each loop-type probability. However, the problems of over- and underflow as well as computational time exist for long RNAs including mRNAs and pre-mRNAs. *ParasoR* is an algorithm based on the `Rfold` model, and its DP algorithm is modified and distributed to multiple computational nodes for a local structure analysis [9]. Similar to global structure prediction, it can compute a variety of structure scores for all possible structures under the constraint of a maximum span for base pairs. This platform is also applied to an efficient simulation of RNA secondary structures, for example, a dynamic conformational change caused by a single point mutation. *Radium* has been developed to detect mutations that can disrupt a large part of a secondary structure, called riboSNIches [37], even within long RNAs. To further improve the computation of global structure prediction based on a DP algorithm, *LinearFold* accelerates MFE structure prediction using beam search, which ignores only the low-probability results to avoid a substantial decrease in prediction accuracy [38]. The acceleration depends on the beam size that defines the stack size of the partial structures to be considered for the next DP computation. The computational complexity of *LinearFold* MFE structure prediction is $O(Nb \log b)$, where N is the sequence length and b is the user-defined beam size. This strategy can also be applied to RNA–RNA interaction prediction as implemented in *LinearCofold*, which is introduced in this series.

As such, a wide variety of RNA secondary structure prediction tools have been developed in terms of their possible structures, metrics to evaluate the structures, or approaches to extract representative structures. Several platforms have been developed for structure analysis using multiple tools simultaneously, including ViennaRNA [39], Freiburg RNA Tools [40], and *Rtools* [5]. *Rtools* is a web server that can analyze a query sequence with eight different applications for RNA secondary structure analysis. `CentroidFold` and `CentroidHomfold` predict the representative structure for the query according to the γ -centroid estimators. `RintD` and `RintW` are tools to visualize the distribution of the secondary structures over the Hamming distance from the reference structures, indicating the stability of the reference structures despite thermodynamic fluctuation. Some other workbenches for RNA-seq analysis also provide tools for RNA secondary structure analysis, and these can be used for the quick analysis of target transcripts as well as ncRNAs [41].

While existing structure prediction models can achieve accurate prediction in general, the accuracy of structure prediction may occasionally be decreased for certain RNAs, such as long RNAs or RNAs in vivo. Previous studies have attempted to improve models and parameters through machine-learning approaches. The classical thermodynamic models contain several thousands of parameters, for example, with 7850 parameters used in the full Turner model [42]. Optimizing those models with a large number of parameters poses a risk of overfitting. *MXfold* is a method to train model parameters for the free energy change approximation based on a structured support vector machine [43]. Combining with L1 regularization, *MXfold* has been shown to produce the best prediction accuracy while

avoiding overfitting. Other than the structure of single RNAs, the prediction of RNA–RNA interactions has been actively studied because those interactions are tightly related to the function of ncRNAs, such as miRNAs, siRNAs, and long ncRNAs. While there is substantial space to explore for potential binding regions genome-wide, *RIblast* and *RIsearch2* apply a seed-and-extend-based alignment strategy to speed up and improve the discovery of highly complementary regions that can form a stable structure with the queried RNA [44]. By changing the setting for the seed search step to use a fast computation library, it enables the efficient discovery of the candidate regions that can form stable structures with base pairs. As an introduction of various RNA–RNA interaction predictions, Fukunaga et al. have provided a comprehensive survey on the prediction web services including their *LncRRsearch* [45].

Application of RNA Secondary Structure Prediction

Thanks to highly accurate structure prediction, the comparison of predicted structures can be further applied to the functional analysis of RNAs. In particular, the comparison of predicted structure stability with experimentally determined accessibility has the potential to reveal the existence of the external factors that cause structure alteration including RNA binding proteins or base modification [46]. While RNA secondary structure can be experimentally determined by several approaches (e.g., X-ray crystal structure analysis, cryo-electron microscopy, and NMR), they require the appropriate concentrations of a single RNA and suffer from problems of feasibility and throughput limitations. A *high-throughput structure probing method* is an approach that can overcome the throughput and coverage problems of existing conformational analysis methods. By using a high-throughput sequencing technique, this method detects RNA modification at reactive sites. Takizawa has introduced a means of inferring the reactivity of each base from high-throughput structure probing data using the computational methods PROBer [47], BUMHMM [48], and reactIDR [13]. In Chap. 13, the author applied these pipelines to discover the structural constraints on the RNA genome of influenza virus [14].

Similarly, the combination of RNA binding protein (RBP) pulldown and high-throughput RNA sequence analysis has been widely applied at transcriptome-wide to reveal the mechanism of RBP and (m)RNA. However, this strategy is susceptible to biased backgrounds and false positives derived from the pulldown assay step, even when UV cross linking is included, as is used in CLIP-seq and its alternatives [49]. For the accurate inference of sequential and structural RBP binding motifs, an artificial library of random short RNAs is utilized in SELEX [50] and RNACompete [51], enabling an efficient motif analysis with high coverage. These methods solve the practical problem of capturing the desired aptamers with a binding affinity that is high enough for the target RBP among a pool of random RNAs. *ResidualBind*, introduced in this series, is a deep learning framework for inferring RBP binding motifs from experimental RBP binding data [52]. The outstanding characteristic of ResidualBind is that it can perform a global importance analysis for the existence of motifs to improve the model interpretability of their deep learning model.

The functional domains of RNAs, particularly of ncRNAs, have been discovered by examining the conservation of not only sequences but also secondary structures required to interact with other molecules to perform biological roles. One example of the potential conservation signature is a pattern found in RNA structure-aware alignment [53]. Specifically, it is defined as a co-occurrence of two mutations at paired bases that does not disrupt their pairing [54]. Walter Costa et al. (2019) developed a novel approach, called *SSS-test*, to evaluate the significance of positive and negative selection on RNA structures based on

mutational conservation signatures [55]. SSS-test is designed to perform a statistical test of the consensus structure by comparing the predicted structures with and without mutational variants. This strategy can be used to analyze the structural constraints on RNAs with unknown functions. To infer the consensus structure, which is essential for a conservation analysis to understand their common function, this book introduces *TOPAS*, which is a novel algorithm for performing network-based alignment of RNA secondary structures [56]. In *TOPAS*, the input of RNA sequences and base pairing probability matrix are integrated to construct topological RNA structure networks. Using probabilistic network alignment techniques, *TOPAS* can find a structurally sound alignment with a low computational complexity of $O(n^2)$.

Other Applications of RNA Secondary Structure Prediction

Corresponding with the continuous improvement of structure prediction methods as well as the ongoing accumulation of experimental structure data, RNA structure prediction has become a progressively more practical tool for studying a wide variety of forms of downstream regulation by functional RNAs. Although it is beyond the scope of this series, one of the remarkable contributions of machine learning technology and RNA structure analysis is the expansion of the field of RNA design. In contrast to structure prediction, in which the sequence is folded to form a stable structure, the problem of “inverse folding” is predicting the sequences that can form a desired structure. This problem has been actively studied as the application of RNA structure prediction because it is closely connected to the engineering of RNAs. For example, an RNA aptamer is a molecule that can bind to a target molecule with a strong affinity and thus can be applied to protein detection for research and diagnostic purposes as well as utilized as a therapeutic drug [57]. Because the binding affinity of RNA aptamers highly depends on the secondary structure, the results of inverse folding can be used as the first filtering process for identifying reliable candidates followed by experimental enrichment-based selection.

On the other hand, the inverse folding problem requires substantial computational resources owing to the enormous sequence and structure space to be explored [58]. To solve this problem with heuristics, previous studies have applied a wide variety of approaches such as constraint programming [59], genetic algorithms [60], and ant colony optimization [61]. More recently, methods that demand either more reference data or sophisticated exploration strategies have been applied to the problem, including fragment-based searches of the sequences [62], Monte Carlo tree search [63], and the efficient sampling of stable structures [64]. There is also the challenge of finding solutions that completely differ from human intuition [65]. By incorporating datasets from such different sources, the feasibility of inverse folding for versatile applications is expected to improve.

As such, the advancement of computational strategies and machine learning methods will provide us with further applications of RNA secondary structure prediction for medical research, such as the prediction of pathogenic mutations [66] or splicing efficiency prediction [67]. Because RNA secondary structure prediction is closely related to 3D structure prediction and frequently used as input, the continued improvement of RNA secondary structure prediction would benefit 3D structure prediction and vice versa.

RNA 3D Structure Databases

Similar to protein structure data, 3D structures of RNA can be deposited in the PDB, which is freely accessed from all over the world. As the name suggests, the PDB is a database that mainly maintains and distributes 3D structures of proteins. Additionally, several types of ancillary information, such as primary sequence, secondary structure (e.g., helix, sheet), family/domain annotations, and many external links to other useful databases, are also provided for each protein entry, whereas such useful information is not provided for RNA entries. To complement the data contents of RNA/DNA entries in the PDB, RNA and DNA structural data has been mirrored and distributed by the Nucleic Acids Database (NDB) since 1992 [68]. The NDB is a database that specializes in distributing RNA- and DNA-specific structural data, including base pairs, base stacking, and classification of base-pairing hydrogen bonds.

It is worth mentioning that various types of base-pairing interactions are observed in the known 3D structures of RNAs, including A–U and G–U Watson–Crick-type pairing, G–U wobble base pairing, and non-Watson–Crick-type base pairing. These base pairs are categorized into 12 types based on the combination of interacting edges in nucleobases (i.e., Watson–Crick, Hoogsteen, and sugar) and the relative orientation of glycosidic bonds (i.e., *cis* or *trans*). According to a statistical analysis of the determined RNA 3D structures [69], 76% of base pairs are *cis* Watson–Crick/Watson–Crick-type base pairs known as canonical Watson–Crick base pairs, and the remaining 24% of base pairs are non-Watson–Crick type, which demonstrates the important roles of non-canonical base-pairs in the appropriate folding of 3D structures of RNA.

Computational Prediction and Design of RNA 3D Structure

Most functional RNAs exhibit biological functions that require their appropriate folding into characteristic 3D structures. With the rapid growth of sequencing technology, a huge number of RNAs expressed in living cells have been identified. For example, the number of human transcripts registered in the GENCODE database (release 38) is more than 200,000 with currently available primary sequences [70]. However, experimental determination of RNA tertiary structures is far less advanced than the growth of primary sequence information. According to statistics released by the PDB, where most 3D structure data on biomolecules (including protein, DNA, and RNA) is deposited, currently, the number of protein-only 3D structures exceeds 150,000 entries, whereas the number of RNA-only structures has only reached 1500 entries. In order to fill the large gap between the abundance of available primary sequence data and the paucity of 3D structure data, accurate computational methods of predicting 3D structures of RNA are urgently needed.

Computational prediction of 3D structures of RNA can be categorized into the following three groups: template-based modeling, fragment/motif assembly, and folding simulation. The first type of prediction methods, template-based modeling is based on prediction using, as a template, an existing RNA tertiary structure with a primary sequence that is highly similar to the query sequence. For this method, the template RNA 3D structure and the pairwise sequence alignment between the template and query RNAs are required as input data. Similar to the template-based modeling frequently used for protein 3D structure prediction, this method is effective if highly homologous template RNAs can be found in a database. However, the majority of available structural data for template RNAs is limited to

several types of ncRNAs, such as tRNA, rRNA, snRNA, and ribozymes, in the PDB and NDB.

In fragment/motif assembly, the second type of prediction method, known three-dimensional structures of RNA are decomposed into small or partial RNA fragments that can be mixed and combined like building blocks to predict an RNA 3D structure corresponding to the query RNA sequence. In terms of fragment size used in the prediction methods, the prediction tools can be categorized into two groups corresponding to fragment size: medium-sized fragment (often referred as motif) tools, such as RNAComposer [71], MC-fold [72], 3dRNA [73], and Vfold3D [74], and small-sized fragment tools (utilizing only a few nucleotides or base pairs in a helix), such as FARFAR2 [75]. Similar to template-based modeling, in general, prediction based on larger RNA fragments with primary sequences that are highly homologous to the users' input sequence yields more reliable prediction. However, appropriately large RNA fragments are not frequently found in the existing RNA structure databases. This reflects the trade-off between fragment size (i.e., prediction accuracy) and data availability.

In folding simulations, the third type of prediction method, 3D structures of RNA are predicted through the simulation of RNA folding dynamics for an unfolded initial RNA structure to obtain a well-folded 3D structure. For this type of prediction, each nucleotide in an RNA molecule is represented by 3–7 coarse-grained beads rather than all-atom RNA models in order to reduce the computational cost of the simulation. Monte Carlo simulation is used for sampling the conformation of the coarse-grained RNA structure. The following two types of energy functions are used for the simulation process: physical chemistry-based energy functions and knowledge-based statistical potential. Physical chemistry-based energy functions describe various types of interactions, such as hydrogen bonds formed in base pairs and electrostatic repulsion between the backbone phosphates [76]. Knowledge-based statistical potential is derived from the statistical analysis of the existing RNA 3D structures [77]. The general computation time of the folding simulation depends on the length of the query RNA, and, hence, tends to be relatively longer than those of other methods. In this book, we invited Watkins and Das to present their method *FARFAR2*, which successfully predicts native-like three-dimensional structures of RNA, as one of the current state-of-the-art fragment assembly methods [74].

As another extension of the prediction of 3D structures of RNA, the rational design of arbitrary RNA structures has potential to broaden the field of RNA applications in biotechnology and nanotechnology. However, designing RNA 3D-structure is a difficult task that requires expert knowledge of RNA 3D structures. For example, there is the task of RNA molecule design to obtain structures containing two helices and one junction, thus connecting two distinct functional RNAs into a single RNA molecule. Even this task is necessarily a time-consuming trial and error process. In this book, we invited Jurich and Yesselman to introduce their unique software toolkit *RNAMake* [78], which enables non-expert users to build/design RNA 3D structures in several practical situations.

Representative RNA 3D Structure Dataset and RNA-Puzzles

To develop prediction methods for 3D structures of RNA, various types of structural and statistical information, such as canonical and non-canonical base pairs, small-to-medium-sized RNA fragments for assembly, and backbone torsions, must be extracted from a wide variety of known RNA 3D structures. In PDB and NDB, however, current RNA entries are highly biased towards several types of RNAs. At the same time, there are multiple redundant

entries for these specific types of RNAs, which can sometimes be problematic for developing and training prediction methods. Currently, a dataset comprised of only non-redundant RNA 3D structures is maintained and distributed by the BGSU RNA site (<http://rna.bgsu.edu/rna3dhub/>) as a representative set, and it is updated weekly to include the most recently determined RNA 3D structures. This essential resource has been widely applied to many previous studies of RNA structural prediction methods [79].

Because RNA 3D structure prediction is a challenging problem, continuous efforts to develop and improve prediction methods are required. In the field of protein structure prediction, a blind test of community-wide 3D structure prediction methods called Critical Assessment Structure Prediction (CASP) has been held since 1994, and many researchers from around the world participate in each CASP round [80, 81]. This continuous management of CASP encourages researchers to develop and improve their methods for predicting protein 3D structures. In fact, the prediction accuracy of protein structures has greatly improved in recent decades. Similar to CASP, a framework for blind testing the prediction of 3D structures of RNA and the evaluation of predicted structures, called RNA-Puzzles, has been organized as a community effort since 2011 [8, 82].

As well as developing methods for predicting the 3D structure of RNA, it is important to evaluate the accuracy of prediction methods in an appropriate manner. Especially for 3D structures of RNA, the evaluation of Watson–Crick and non-Watson–Crick type base-pair prediction is a topic that is important for the development of more accurate structure prediction methods. Several metrics that have been used to evaluate protein 3D structure methods (e.g., RMSD, root-mean-square deviations) have been shown to be unsuitable for this purpose for RNA [83]. In addition, the coordinates of several atoms and partial fragments are occasionally missing in RNA structural data in the PDB, and these are referred to as missing atoms/residues, though they are included in the actual RNA molecules. It should also be noted that it is necessary to handle such incomplete data in practical applications of RNA 3D structure prediction. In this book, Magnus and Miao introduce their useful software named *RNA-Puzzles toolkit* [84], which provides a framework for editing and handling such incomplete RNA 3D structural data, and for evaluating prediction results such as those of RNA-Puzzles.

Cold Spring Harbor, NY, USA
Kashiwa, Japan

Risa Karakida Kawaguchi
Junichi Iwakiri

References

1. Gomas PJ, Tamm I (1963) The secondary structure of Reovirus RNA. *Proc Natl Acad Sci*. <https://doi.org/10.1073/pnas.49.5.707>
2. Kim S (1976) Three-dimensional structure of transfer RNA. *Prog Nucleic Acid Res Mol Biol*. [https://doi.org/10.1016/s0079-6603\(08\)60070-7](https://doi.org/10.1016/s0079-6603(08)60070-7)
3. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The protein data bank. *Nucleic Acids Res* 28(1):235–242. <https://doi.org/10.1093/nar/28.1.235>
4. Zuker M, Stiegler P (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res* 9(1):133–148. <https://doi.org/10.1093/nar/9.1.133>
5. Hamada M, Ono Y, Kiryu H, Sato K, Kato Y, Fukunaga T, Mori R, Asai K (2016) Rtools: a web server for various Secondary structural analyses on single RNA sequences. *Nucleic Acids Res* 44(W1):W302–W307. <https://doi.org/10.1093/nar/gkw337>

6. Lu ZJ, Gloor JW, Mathews DH (2009) Improved RNA secondary structure prediction by maximizing expected pair accuracy. *RNA* 15(10):1805–1813. <https://doi.org/10.1261/rna.1643609>
7. Puton T, Kozłowski LP, Rother KM, Bujnicki JM (2014) CompaRNA: a server for continuous benchmarking of automated methods for RNA secondary structure prediction. *Nucleic Acids Res* 42(8):5403–5406. <https://doi.org/10.1093/nar/gku208>
8. Miao Z, Adamiak RW, Antczak M, Boniecki MJ, Bujnicki JM, Chen S et al (2020) RNA-puzzles round IV: 3D structure predictions of four ribozymes and two aptamers. *RNA* 26(8):982–995. <https://doi.org/10.1261/rna.075341.120>
9. Kawaguchi R, Kiryu H (2016) Parallel computation of genome-scale RNA secondary structure to detect structural constraints on human genome. *BMC Bioinformatics* 17(1):203. <https://doi.org/10.1186/s12859-016-1067-9>
10. Lucks JB, Mortimer SA, Trapnell C, Luo S, Aviran S, Schroth GP, Pachter L, Doudna JA, Arkin AP (2011) Multiplexed RNA structure characterization with selective 2'-hydroxyl acylation analyzed by primer extension sequencing (SHAPE-Seq). *Proc Natl Acad Sci U S A* 108(27):11063–11068. <https://doi.org/10.1073/pnas.1106501108>
11. Rouskin S, Zubradt M, Washietl S, Kellis M, Weissman JS (2014) Genome-wide probing of RNA structure reveals active unfolding of mRNA structures in vivo. *Nature* 505(7485):701–705. <https://doi.org/10.1038/nature12894>
12. Sexton AN, Wang PY, Rutenberg-Schoenberg M, Simon MD (2017) Interpreting reverse transcriptase termination and mutation events for greater insight into the chemical probing of RNA. *Biochemistry* 56(35):4713–4721. <https://doi.org/10.1021/acs.biochem.7b00323>
13. Kawaguchi R, Kiryu H, Iwakiri J, Sese J (2019) reactIDR: evaluation of the statistical reproducibility of high-throughput structural analyses towards a robust RNA Structure prediction. *BMC Bioinformatics* 20(Suppl 3):130. <https://doi.org/10.1186/s12859-019-2645-4>
14. Takizawa N, Higashi K, Kawaguchi RK, Gotoh Y, Suzuki Y, Hayashi T, Kurokawa K (2021) Comprehensive in vitro structure probing analysis of the Influenza A virus identifies a functional RNA structure involved in replication and segment interactions. *bioRxiv*. <https://doi.org/10.1101/2020.03.05.975870>
15. Strobel EJ, Yu AM, Lucks JB (2018) High-throughput determination of RNA structures. *Nat Rev Genet*. <https://doi.org/10.1038/s41576-018-0034-x>
16. Siegfried NA, Busan S, Rice GM, Nelson JNE, Weeks KM (2014) RNA motif discovery by SHAPE and mutational profiling (SHAPE-MaP). *Nat Methods* 11(9):959–965. <https://doi.org/10.1038/nmeth.3029>
17. Zubradt M, Gupta P, Persad S, Lambowitz AM, Weissman JS, Rouskin S (2017) DMS-MaPseq for genome-wide or targeted RNA structure probing in vivo. *Nat Methods* 14(1):75–82. <https://doi.org/10.1038/nmeth.4057>
18. Li H, Aviran S (2018) Statistical modeling of RNA Structure profiling experiments enables parsimonious reconstruction of structure landscapes. *Nat Commun* 9(1):606. <https://doi.org/10.1038/s41467-018-02923-8>
19. Mustoe AM, Lama NN, Irving PS, Olson SW, Weeks KM (2019) RNA Base-pairing complexity in living cells visualized by correlated chemical probing. *Proc Natl Acad Sci U S A* 116(49):24574–24582. <https://doi.org/10.1073/pnas.1905491116>
20. Denesyuk NA, Thirumalai D (2015) How do metal ions direct ribozyme folding? *Nat Chem* 7(10):793–801. <https://doi.org/10.1038/nchem.2330>
21. Schärffen L, Neugebauer KM (2021) Transcription regulation through nascent RNA folding. *J Mol Biol*:166975. <https://doi.org/10.1016/j.jmb.2021.166975>
22. Yu AM, Gasper PM, Cheng L, Lai LB, Kaur S, Gopalan V, Chen AA, Lucks JB (2021) Computationally reconstructing cotranscriptional RNA folding from experimental data reveals rearrangement of non-native folding intermediates. *Mol Cell* 81(4):870–83.e10. <https://doi.org/10.1016/j.molcel.2020.12.017>
23. Lai D, Proctor JR, Meyer IM (2013) On the importance of cotranscriptional RNA structure formation. *RNA* 19(11):1461–1473. <https://doi.org/10.1261/rna.037390.112>
24. Bell DR, Weber JK, Yin W, Huynh T, Duan W, Zhou R (2020) In silico design and validation of high-affinity RNA aptamers targeting epithelial cellular adhesion molecule dimers. *Proc Natl Acad Sci U S A* 117(15):8486–8493. <https://doi.org/10.1073/pnas.1913242117>
25. Do CB, Woods DA, Batzoglou S (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* 22(14):e90–e98. <https://doi.org/10.1093/bioinformatics/btl246>

26. Andronescu M, Condon A, Hoos HH, Mathews DH, Murphy KP (2010) Computational approaches for RNA energy parameter estimation. *RNA* 16(12):2304–2318. <https://doi.org/10.1261/rna.1950510>
27. Singh J, Hanson J, Paliwal K, Zhou Y (2019) RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning. *Nat Commun* 10(1):5407. <https://doi.org/10.1038/s41467-019-13395-9>
28. Wirecki TK, Merdas K, Bernat A, Boniecki MJ, Bujnicki JM, Stefaniak F (2020) RNAProbe: a web server for normalization and analysis of RNA structure probing data. *Nucleic Acids Res* 48(W1):W292–W299. <https://doi.org/10.1093/nar/gkaa396>
29. Alipanahi B, Delong A, Weirauch MT, Frey BJ (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat Biotechnol* 33(8):831–838. <https://doi.org/10.1038/nbt.3300>
30. Aw JGA, Lim SW, Wang JX, Lambert FRP, Tan WT et al (2021) Determination of isoform-specific RNA structure with nanopore long reads. *Nat Biotechnol* 39(3):336–346. <https://doi.org/10.1038/s41587-020-0712-z>
31. Hamada M, Kiryu H, Sato K, Mituyama T, Asai K (2009) Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics* 25(4):465–473. <https://doi.org/10.1093/bioinformatics/btn601>
32. Jabbari H, Wark I, Montemagno C, Will S (2018) Knotty: efficient and accurate prediction of complex RNA pseudoknot structures. *Bioinformatics* 34(22):3849–3856. <https://doi.org/10.1093/bioinformatics/bty420>
33. Brierley I, Pennell S, Gilbert RJC (2007) Viral RNA pseudoknots: versatile motifs in gene expression and replication. *Nat Rev Microbiol* 5(8):598–610. <https://doi.org/10.1038/nrmicro1704>
34. Kimchi O, Cragolini T, Brenner MP, Colwell LJ (2019) A polymer physics framework for the entropy of arbitrary pseudoknots. *Biophys J* 117(3):520–532. <https://doi.org/10.1016/j.bpj.2019.06.037>
35. Wang F, Akutsu T, Mori T (2020) Comparison of pseudoknotted RNA secondary structures by topological centroid identification and tree edit distance. *J Comput Biol* 27(9):1443–1451. <https://doi.org/10.1089/cmb.2019.0512>
36. Kiryu H, Kin T, Asai K (2008) Rfold: an exact algorithm for computing local base pairing probabilities. *Bioinformatics* 24(3):367–373. <https://doi.org/10.1093/bioinformatics/btm591>
37. Wan Y, Qu K, Zhang QC, Flynn RA, Manor O et al (2014) Landscape and variation of RNA secondary structure across the human transcriptome. *Nature* 505(7485):706–709. <https://doi.org/10.1038/nature12946>
38. Huang L, Zhang H, Deng D, Zhao K, Liu K, Hendrix DA, Mathews DH (2019) LinearFold: linear-time approximate RNA folding by 5'-to-3' dynamic programming and beam search. *Bioinformatics* 35(14):i295–i304. <https://doi.org/10.1093/bioinformatics/btz375>
39. Gruber AR, Bernhart SH, Lorenz R (2015) The ViennaRNA web services. *Methods Mol Biol* 1269:307–326. https://doi.org/10.1007/978-1-4939-2291-8_19
40. Raden M, Ali SM, Alkhnabashi OS, Busch A, Costa F, Davis JA, Eggenhofer F, Gelhausen R, Georg J, Heyne S, Hiller M, Kundu K, Kleinkauf R, Lott SC, Mohamed MM, Mattheis A, Miladi M, Richter AS, Will S, Wolff J, Wright PR, Backofen R (2018) Freiburg RNA tools: a central online resource for RNA-focused research and teaching. *Nucleic Acids Res* 46(W1):W25–W29. <https://doi.org/10.1093/nar/gky329>
41. Fallmann J, Videm P, Bagnacani A, Batut B, Doyle MA, Klingstrom T, Eggenhofer F, Stadler PF, Backofen R, Grüning B (2019) The RNA workbench 2.0: next generation RNA data analysis. *Nucleic Acids Res* 47(W1):W511–W515. <https://doi.org/10.1093/nar/gkz353>
42. Andronescu MS, Pop C, Condon AE (2010) Improved free energy parameters for RNA pseudoknotted secondary structure prediction. *RNA* 16(1):26–42. <https://doi.org/10.1261/rna.1689910>
43. Akiyama M, Sato K, Sakakibara Y (2018) A max-margin training of RNA secondary structure prediction integrated with the thermodynamic model. *J Bioinforma Comput Biol* 16(6):1840025. <https://doi.org/10.1142/S0219720018400255>
44. Fukunaga T, Hamada M (2017) RIBlast: an ultrafast RNA-RNA interaction prediction system based on a seed-and-extension approach. *Bioinformatics* 33(17):2666–2674. <https://doi.org/10.1093/bioinformatics/btx287>

45. Fukunaga T, Iwakiri J, Ono Y, Hamada M (2019) LncRRsearch: a web server for lncRNA-RNA interaction prediction integrated with tissue-specific expression and subcellular localization data. *Front Genet* 10:462. <https://doi.org/10.3389/fgene.2019.00462>
46. Spitale RC, Flynn RA, Zhang QC, Crisalli P, Lee B et al (2015) Structural imprints in vivo decode RNA regulatory mechanisms. *Nature* 519(7544):486–490. <https://doi.org/10.1038/nature14263>
47. Li B, Tambe A, Aviran S, Pachter L (2017) PROBER provides a general toolkit for analyzing sequencing-based toeprinting assays. *Cell Syst* 4(5):568–74.e7. <https://doi.org/10.1016/j.cels.2017.04.007>
48. Selega A, Sirocchi C, Iosub I, Granneman S, Sanguinetti G (2017) Robust statistical modeling improves sensitivity of high-throughput RNA structure probing experiments. *Nat Methods* 14(1):83–89. <https://doi.org/10.1038/nmeth.4068>
49. Hafner M, Katsantoni M, Köster T, Marks J, Mukherjee J, Staiger D, Ule J, Zavolan M (2021) CLIP and complementary methods. *Nat Rev Method Primers* 1(1). <https://doi.org/10.1038/s43586-021-00018-1>
50. Tuerk C, Gold L (1990) Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. *Science* 249(4968):505–510. <https://doi.org/10.1126/science.2200121>
51. Ray D, Kazan H, Chan ET, Castillo LP, Chaudhry S, Talukder S, Blencowe BJ, Morris Q, Hughes TR (2009) Rapid and systematic analysis of the RNA recognition specificities of RNA-binding proteins. *Nat Biotechnol* 27(7):667–670. <https://doi.org/10.1038/nbt.1550>
52. Koo PK, Majdandzic A, Ploenzke M, Anand P, Paul SB (2021) Global importance analysis: an interpretability method to quantify importance of genomic features in deep neural networks. *PLoS Comput Biol* 17(5): e1008925. <https://doi.org/10.1371/journal.pcbi.1008925>
53. Wright ES (2020) RNAconTest: comparing tools for noncoding RNA multiple sequence alignment based on structural consistency. *RNA* 26(5):531–540. <https://doi.org/10.1261/rna.073015.119>
54. Kalvari I, Nawrocki EP, Ontiveros-Palacios N, Argasinska J, Lamkiewicz K et al (2021) Rfam 14: expanded coverage of metagenomic, viral and microRNA families. *Nucleic Acids Res* 49(D1):D192–D200. <https://doi.org/10.1093/nar/gkaa1047>
55. Walter Costa MB, Höner Zu Siederdisen C, Dunjić M, Stadler PF, Nowick K (2019) SSS-test: a novel test for detecting positive selection on RNA secondary structure. *BMC Bioinformatics* 20(1):151. <https://doi.org/10.1186/s12859-019-2711-y>
56. Chen CC, Jeong H, Qian X, Yoon BJ (2019) TOPAS: network-based structural alignment of RNA sequences. *Bioinformatics* 35(17):2941–2948. <https://doi.org/10.1093/bioinformatics/btz001>
57. Ni S, Zhuo Z, Pan Y, Yu Y, Li F et al (2021) Recent progress in aptamer discoveries and modifications for therapeutic applications. *ACS Appl Mater Interfaces* 13(8):9500–9519. <https://doi.org/10.1021/acscami.0c05750>
58. Fornace ME, Porubsky NJ, Pierce NA (2020) A unified dynamic programming framework for the analysis of interacting nucleic acid strands: enhanced models, scalability, and speed. *ACS Synth Biol* 9(10):2665–2678. <https://doi.org/10.1021/acssynbio.9b00523>
59. Garcia-Martin JA, Dotu I, Clote P (2015) RNAiFold 2.0: a web server and software to design custom and Rfam-based RNA molecules. *Nucleic Acids Res* 43(W1):W513–W521. <https://doi.org/10.1093/nar/gkv460>
60. Taneda A (2011) MODENA: a multi-objective RNA inverse folding. *Adv Appl Bioinformatics Chem* 4:1. <https://doi.org/10.2147/aabc.s14335>
61. Kleinkauf R, Mann M, Backofen R (2015) antaRNA: ant colony-based RNA sequence design. *Bioinformatics* 31(19):3114–3121. <https://doi.org/10.1093/bioinformatics/btv319>
62. Retwitzer MD, Reinharz V, Churkin A, Ponty Y, Waldspühl J, Barash D (2020) incaRNAfbinv 2.0: a webserver and software with motif control for fragment-based design of RNAs. *Bioinformatics* 36(9): 2920–2922. <https://doi.org/10.1093/bioinformatics/btaa039>
63. Yang X, Yoshizoe K, Taneda A, Tsuda K (2017) RNA inverse folding using Monte Carlo tree search. *BMC Bioinformatics* 18(1):468. <https://doi.org/10.1186/s12859-017-1882-7>
64. Hammer S, Wang W, Will S, Ponty Y (2019) Fixed-parameter tractable sampling for RNA design with multiple target structures. *BMC Bioinformatics* 20(1):209. <https://doi.org/10.1186/s12859-019-2784-7>
65. Rother M, Milanowska K, Puton T, Jeleniewicz J, Rother K, Bujnicki JM (2011) ModeRNA server: an online tool for modeling RNA 3D structures. *Bioinformatics* 27(17):2441–2442. <https://doi.org/10.1093/bioinformatics/btr400>

66. Halvorsen M, Martin JS, Broadaway S, Laederach A (2010) Disease-associated mutations that alter the RNA structural ensemble. *PLoS Genet* 6(8):e1001074. <https://doi.org/10.1371/journal.pgen.1001074>
67. Mikl M, Hamburg A, Pilpel Y, Segal E (2019) Dissecting splicing decisions and cell-to-cell variability with designed sequence libraries. *Nat Commun* 10(1):4572. <https://doi.org/10.1038/s41467-019-12642-3>
68. Berman HM, Olson WK, Beveridge DL, Westbrook J, Gelbin A, Demeny T, Hsieh SH, Srinivasan AR, Schneider B (1992) The nucleic acid database. A comprehensive relational database of three-dimensional structures of nucleic acids. *Biophys J* 63(3):751–759. [https://doi.org/10.1016/S0006-3495\(92\)81649-1](https://doi.org/10.1016/S0006-3495(92)81649-1)
69. Stombaugh J, Zirbel CL, Westhof E, Leontis NB (2009) Frequency and isostericity of RNA base pairs. *Nucleic Acids Res* 37(7):2294–2312. <https://doi.org/10.1093/nar/gkp011>
70. Frankish A, Diekhans M, Ferreira AM, Johnson R, Jungreis I et al (2019) GENCODE reference annotation for the human and mouse genomes. *Nucleic Acids Res* 47(D1):D766–D773. <https://doi.org/10.1093/nar/gky955>
71. Popenda M, Szachniuk M, Antczak M, Purzycka KJ, Lukasiak P, Bartol N, Blazewicz J, Adamiak RW (2012) Automated 3D Structure composition for large RNAs. *Nucleic Acids Res* 40(14):e112. <https://doi.org/10.1093/nar/gks339>
72. Parisien M, Major F (2008) The MC-fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature* 452(7183):51–55. <https://doi.org/10.1038/nature06684>
73. Zhao Y, Huang Y, Gong Z, Wang Y, Man J, Xiao Y (2012) Automated and fast building of three-dimensional RNA structures. *Sci Rep* 734. <https://doi.org/10.1038/srep00734>
74. Xu X, Zhao P, Chen SJ (2014) Vfold: a web server for RNA structure and folding thermodynamics prediction. *PLoS One* 9(9):e107504. <https://doi.org/10.1371/journal.pone.0107504>
75. Watkins AM, Rangan R, Das R (2020) FARFAR2: improved De Novo Rosetta prediction of complex global RNA folds. *Structure* 28(8):963–76.e6. <https://doi.org/10.1016/j.str.2020.05.011>
76. Cragolini T, Laurin Y, Derreumaux P, Pasquali S (2015) Coarse-grained HiRE-RNA model for ab initio RNA folding beyond simple molecules, including noncanonical and multiple base pairings. *J Chem Theory Comput* 11(7):3510–3522. <https://doi.org/10.1021/acs.jctc.5b00200>
77. Boniecki MJ, Lach G, Dawson WK, Tomala K, Lukasz P, Soltysinski T, Rother KM, Bujnicki JM (2016) SimRNA: a coarse-grained method for RNA folding simulations and 3D Structure prediction. *Nucleic Acids Res* 44(7):e63. <https://doi.org/10.1093/nar/gkv1479>
78. Yesselman JD, Eiler D, Carlson ED, Gotrik MR, d’Aquino AE et al (2019) Computational design of three-dimensional RNA structure and function. *Nat Nanotechnol* 14(9):866–873. <https://doi.org/10.1038/s41565-019-0517-8>
79. Leontis NB, Zirbel CL (2012) Nonredundant 3D Structure datasets for RNA knowledge extraction and benchmarking. *Nucleic Acids Mol Biol*. https://doi.org/10.1007/978-3-642-25740-7_13
80. Moult J, Pedersen JT, Judson R, Fidelis K (1995) A large-scale experiment to assess protein structure prediction methods. *Proteins* 23(3):ii–v. <https://doi.org/10.1002/prot.340230303>
81. Kryshtafovych A, Schwede T, Topf M, Fidelis K, Moult J (2019) Critical assessment of methods of protein structure prediction (CASP)-round XIII. *Proteins* 87(12):1011–1020. <https://doi.org/10.1002/prot.25823>
82. Cruz JA, Blanchet MF, Boniecki M, Bujnicki JM, Chen SJ et al (2012) RNA-puzzles: a CASP-like evaluation of RNA three-dimensional structure prediction. *RNA* 18(4):610–625. <https://doi.org/10.1261/rna.031054.111>
83. Parisien M, Cruz JA, Westhof E, Major F (2009) New metrics for comparing and assessing discrepancies between RNA 3D structures and models. *RNA* 15(10):1875–1885. <https://doi.org/10.1261/rna.1700409>
84. Magnus M, Antczak M, Zok T, Wiedemann J, Lukasiak P, Cao Y, Bujnicki JM, Westhof E, Szachniuk M, Miao Z (2020) RNA-puzzles toolkit: a computational resource of RNA 3D structure benchmark datasets, structure manipulation, and evaluation tools. *Nucleic Acids Res* 48(2):576–588. <https://doi.org/10.1093/nar/gkz1108>

Contents

<i>Preface</i>	<i>v</i>
<i>Contributors</i>	<i>xxi</i>
1 Rtools: A Web Server for Various Secondary Structural Analyses on Single RNA Sequences.	1
<i>Yukiteru Ono and Kiyoshi Asai</i>	
2 Linear-Time Algorithms for RNA Structure Prediction	15
<i>He Zhang, Liang Zhang, Kaibo Liu, Sizhen Li, David H. Mathews, and Liang Huang</i>	
3 Genome-Wide RNA Secondary Structure Prediction	35
<i>Risa Karakida Kawaguchi and Hisanori Kiryu</i>	
4 Nucleic Acid Structure Prediction Including Pseudoknots Through Direct Enumeration of States: A User's Guide to the LandscapeFold Algorithm.	49
<i>Ofer Kimchi, Michael P. Brenner, and Lucy J. Colwell</i>	
5 Metrics for RNA Secondary Structure Comparison.	79
<i>Feiqi Wang, Tatsuya Akutsu, and Tomoya Mori</i>	
6 RNA Secondary Structure Prediction Based on Energy Models	89
<i>Manato Akiyama and Kengo Sato</i>	
7 RNA Secondary Structure Alteration Caused by Single Nucleotide Variants	107
<i>Risa Karakida Kawaguchi and Hisanori Kiryu</i>	
8 Evolutionary Conservation of RNA Secondary Structure	121
<i>Maria Beatriz Walter Costa</i>	
9 Network-Based Structural Alignment of RNA Sequences Using TOPAS	147
<i>Chun-Chi Chen, Hyundoo Jeong, Xiaoning Qian, and Byung-Jun Yoon</i>	
10 Fast RNA-RNA Interaction Prediction Methods for Interaction Analysis of Transcriptome-Scale Large Datasets.	163
<i>Tsukasa Fukunaga and Michiaki Hamada</i>	
11 Web Services for RNA-RNA Interaction Prediction	175
<i>Tsukasa Fukunaga, Junichi Iwakiri, and Michiaki Hamada</i>	
12 ResidualBind: Uncovering Sequence-Structure Preferences of RNA-Binding Proteins with Deep Neural Networks	197
<i>Peter K. Koo, Matt Ploenzke, Praveen Anand, Steffan Paul, and Antonio Majdandzic</i>	
13 RNA Structure Determination by High-Throughput Structural Analysis	217
<i>Naoki Takizawa</i>	

14	RNA 3D Modeling with FARFAR2, Online	233
	<i>Andrew M. Watkins and Rhiju Das</i>	
15	Automated 3D Design and Evaluation of RNA Nanostructures with RNAMake.....	251
	<i>Chris P. Jurich and Joseph D. Yesselman</i>	
16	RNA 3D Structure Comparison Using RNA-Puzzles Toolkit	263
	<i>Marcin Magnus and Zhichao Miao</i>	
	<i>Index</i>	287

Contributors

- MANATO AKIYAMA • *Department of Biosciences and Informatics, Keio University, Yokohama, Japan*
- TATSUYA AKUTSU • *Bioinformatics Center, Institute for Chemical Research, Kyoto University, Kyoto, Japan*
- PRAVEEN ANAND • *Dana Farber Cancer Institute, Boston, MA, USA*
- KIYOSHI ASAI • *Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences, University of Tokyo, Tokyo, Japan*
- MICHAEL P. BRENNER • *School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA*
- CHUN-CHI CHEN • *Department of Electrical Engineering, National Chiayi University, Chiayi City, Taiwan*
- LUCY J. COLWELL • *Department of Chemistry, University of Cambridge, Cambridge, UK*
- RHIJU DAS • *Department of Biochemistry, Stanford University School of Medicine, Stanford, CA, USA; Biophysics Program, Stanford University, Stanford, CA, USA*
- TSUKASA FUKUNAGA • *Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan; Graduate School of Advanced Science and Engineering, Waseda University, Tokyo, Japan*
- MICHIAKI HAMADA • *Graduate School of Advanced Science and Engineering, Waseda University, Tokyo, Japan; Department of Electrical Engineering and Bioscience, Faculty of Science and Engineering, Waseda University, Tokyo, Japan; Computational Bio Big-Data Open Innovation Laboratory (CBBB-OIL), National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan*
- LIANG HUANG • *School of Electrical Engineering & Computer Science, Oregon State University, Corvallis, OR, USA*
- JUNICHI IWAKIRI • *Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences, The University of Tokyo, Chiba, Japan*
- HYUNDOO JEONG • *Department of Mechatronics Engineering, Incheon National University, Incheon, Republic of Korea*
- CHRIS P. JURICH • *Department of Chemistry, University of Nebraska-Lincoln, Lincoln, NE, USA*
- RISA KARAKIDA KAWAGUCHI • *Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA*
- OFER KIMCHI • *School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA; Lewis-Sigler Institute for Integrative Genomics, Princeton University, Princeton, NJ, USA*
- HISANORI KIRYU • *Department of Computational Biology and Medical Sciences, The University of Tokyo, Tokyo, Japan*
- PETER K. KOO • *Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA*
- SIZHEN LI • *School of Electrical Engineering & Computer Science, Oregon State University, Corvallis, OR, USA*
- KAIBO LIU • *Baidu Research USA, Sunnyvale, CA, USA*

- MARCIN MAGNUS • *ReMedy-International Research Agenda Unit, Centre of New Technologies, University of Warsaw, Warsaw, Poland*
- ANTONIO MAJDANDZIC • *Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA*
- DAVID H. MATHEWS • *Dept. of Biochemistry & Biophysics, Center for RNA Biology, Rochester, NY, USA; Dept. of Biostatistics & Computational Biology, University of Rochester Medical Center, Rochester, NY, USA*
- ZHICHAO MIAO • *European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI), Hinxton, UK; Department of Anesthesiology, Translational Research Institute of Brain and Brain-Like Intelligence, Shanghai Fourth People's Hospital Affiliated to Tongji University School of Medicine, Shanghai, China*
- TOMOYA MORI • *Bioinformatics Center, Institute for Chemical Research, Kyoto University, Kyoto, Japan*
- YUKITERU ONO • *Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences, University of Tokyo, Tokyo, Japan*
- STEFFAN PAUL • *Bioinformatics Program, Harvard Medical School, Boston, MA, USA*
- MATT PLOENZKE • *Department of Biostatistics, Harvard University, Cambridge, MA, USA*
- XIAONING QIAN • *Department of Electrical and Computer Engineering, Texas A & M University, College Station, TX, USA; Computational Science Initiative, Brookhaven National Laboratory, Upton, USA*
- KENGO SATO • *School of System Design and Technology, Tokyo Denki University, Tokyo, Japan*
- NAOKI TAKIZAWA • *Laboratory of Virology, Institute of Microbial Chemistry (BIKAKEN), Tokyo, Japan*
- MARIA BEATRIZ WALTER COSTA • *Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, University of Leipzig, Leipzig, Germany; Institute of Laboratory Medicine, Clinical Chemistry und Molecular Diagnostics, University of Leipzig Medical Center, Leipzig, Germany*
- FEIQI WANG • *Bioinformatics Center, Institute for Chemical Research, Kyoto University, Kyoto, Japan*
- ANDREW M. WATKINS • *Department of Biochemistry, Stanford University School of Medicine, Stanford, CA, USA; Prescient Design, Genentech, South San Francisco, CA, USA*
- JOSEPH D. YESSELMAN • *Department of Chemistry, University of Nebraska-Lincoln, Lincoln, NE, USA*
- BYUNG-JUN YOON • *Department of Electrical and Computer Engineering, Texas A & M University, College Station, TX, USA; Computational Science Initiative, Brookhaven National Laboratory, Upton, USA*
- HE ZHANG • *Baidu Research USA, Sunnyvale, CA, USA; School of Electrical Engineering & Computer Science, Oregon State University, Corvallis, OR, USA*
- LIANG ZHANG • *Baidu Research USA, Sunnyvale, CA, USA; School of Electrical Engineering & Computer Science, Oregon State University, Corvallis, OR, USA*



Chapter 1

Rtools: A Web Server for Various Secondary Structural Analyses on Single RNA Sequences

Yukiteru Ono and Kiyoshi Asai

Abstract

Predicting the secondary structures of RNA molecules is an essential step to characterize their functions, but the thermodynamic probability of any prediction is generally small. On the other hand, there are a few tools for calculating and visualizing various secondary structural information from RNA sequences. We implemented a web server that calculates in parallel various features of secondary structures: different types of secondary structure predictions, the marginal probabilities for local structural contexts, accessibilities of the subsequences, the energy changes by arbitrary base mutations, and the measures for validations of the predicted secondary structures. The web server is available at <http://rtools.cbrc.jp>, which integrates software tools, CentroidFold, CentroidHomfold, IPknot, CapR, Raccess, Rchange, RintD, and RintW.

Key words RNA secondary structure analysis, Web server, Structure alignment, Pseudoknot

1 Introduction

The secondary structure of RNA molecules, as well as the primary sequences, is one of the fundamental features to characterize their functions. While experimental probing techniques have been invented to reveal RNA secondary structure *in vivo* and *in vitro*, computational *in silico* predictions are still useful because of their reasonable speed and costs. There are many tools and web servers to predict the secondary structure of given RNA sequences, but the probabilities of the predicted structures based on the thermodynamic Boltzmann distribution are generally very small. Therefore, it is often inappropriate to proceed to downstream analyses based on the predicted secondary structures.

There are a few computational tools to compensate imperfect prediction of secondary structures by calculating and visualizing additional structural information. Among such structural information, the most widely recognized is the base-pairing probability matrix, the matrix of marginal probabilities that each pair of the nucleotides forms a base pair. We can expand the concept to

marginal probabilities of the other local structural contexts, stem probabilities, loop probabilities, bulge probabilities, and so on as implemented in CapR [1].

To capture the landscape of the probability distribution of the secondary structures, the distributions on Hamming distance from the reference structures are useful. RintD [2] and RintW [3] calculate the probability distribution of secondary structures and base-pairing probabilities on Hamming distance.

An RNA molecule identifies its target RNA/DNA by base-pair interactions, which consist of mutually reverse complementary subsequences. Raccess [4] is a tool to identify the “free” bases in probability distribution of the secondary structures, to identify the available regions for interacting with other molecules. It is often of interest how the RNA secondary structures change according to the mutation of the bases. Rchange [5] is a tool to calculate the change of the entropy and internal energy by the mutation of the bases.

We have implemented a web server, rtools [6], to operate various analyses above in parallel to help the recognition of the whole picture of the secondary structure of the given RNAs.

2 Design of the Web Server

The interface of rtools is quite simple (Fig. 1), where you enter RNA sequence that you want to analyze in the main input box. The input RNA sequence must be single FASTA format and less than or equal to 400 nt. And then, you press “Submit” button, and rtools quickly returns various analyses for the sequence on the result page (Fig. 2). At the top of the result page, the progress of analysis of each tool is displayed. When the analysis is completed, “completed” is displayed, and you can jump to the analysis result of the tool by clicking it. The result page shows the analysis results of all the tools connected in a single long page. On the result page, for any tool, you can change the parameters and rerun by pressing “Update” button (Fig. 3), and the results of other tools remain the same. The analysis results (e.g., base-pairing probability matrix and various graphs) are offered as the following format – TEXT, PNG, PDF, and EPS – which can be easily utilized in your further analysis and your research paper. You obtain “Request ID” in the result page and can always access the results by using the ID. However, rtools only stores results for 30 days, so be sure to download any results you want to keep. You can select only the tools you want to run by checking and unchecking the checkboxes at each tool name.

Bioinformatics Web Server for RNA

Enter your RNA sequence (single FASTA format, <= 400nt)

Samples: [Hammerhead ribozyme](#), [tRNA](#), [H/ACA snoRNA](#), [TPP-riboswitch](#)

Softwares

CentroidFold

Prediction of RNA secondary structure.

: specify the inference engine

: weight of base pairs

Rtips IPknot: IP-based prediction of RNA pseudoKNOTS

Prediction of RNA pseudoknot based on maximizing expected accuracy.

: specify the inference engine

(NUPACK can be selected only when RNA length is <= 100nt)

: weight of base pairs (level 2)

CapR

Calculation of probabilities that each RNA base position is located within each secondary structural context.

: maximal span of base pairs

RintD

Validation of RNA secondary structure.

structure.1

predicted by CentroidFold

: weight of base pairs

Your structure (dot-bracket format without sequence)

structure.2

predicted by RNAfold

Your structure (dot-bracket format without sequence)

CentroidHomfold

Prediction of RNA secondary structure by using homologous sequences.

: specify the inference engine for secondary structure

: specify the inference engine for pairwise alignment

: weight of base pairs

: E-value for homology search against Rfam

: number of homologous sequences

Rchange

Calculation of energy changes of RNA secondary structures after base mutations.

: maximal span of base pairs

Raccess

Calculation of structural accessibility of RNA transcripts.

: maximal span of base pairs

RintW

Capturing alternative secondary structures of RNA.

: specify the inference engine for canonical structure

: weight of base pairs

Fig. 1 The top page of rtools web server

3 Implemented Computational Tools

This section introduces eight tools that are used by rtools web server. The analysis results of tRNA are used as examples of the figures offered by rtools.

3.1 CentroidFold

CentroidFold based on a generalized centroid estimator is one of the most accurate tools for predicting RNA secondary structures [7]. CentroidFold offers secondary structure and base-pairing probability (BPP) matrix as analysis results (Figs. 4 and 5). The predicted secondary structure is colored according to BPPs. In text format, secondary structure is a dot-bracket format with a FASTA-like header line, indicating a secondary structure. In a dot-bracket format, each dot represents an unpaired base, and opening and

Bioinformatics Web Server for RNA


HOME	HELP
------	------

Request ID:

>AL138651.1/64525-64597

GGGGAUGUAGCUCAUAUGGUAGAGCGCUCGCUUUGCAUGCGAGAGGCACAGGGUUCGAUUCCUGCAUCUCCA

[go to results](#)

CentroidFold	: Completed
CentroidHomfold	: Completed
 tips IPknot: IP-based prediction of RNA pseudoKNOTs	: Completed
Rchange	: Completed
CapR	: Completed
Raccess	: Completed
RintD	: Completed
RintW	: Completed

Download: [all results](#)

Fig. 2 The result page of rtools web server

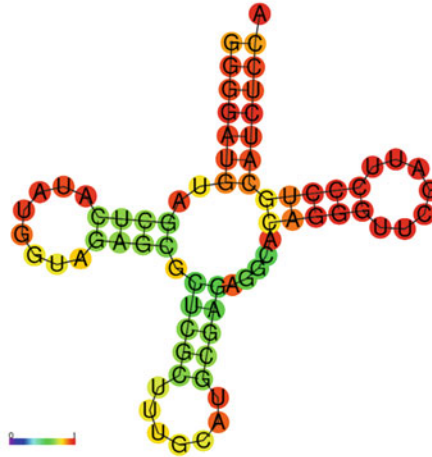
closing brackets represent a base pair. BPP matrix (upper triangle) is presented together with the predicted secondary structure (lower triangle). Detailed values of BPP can be obtained in text format.

The algorithm of CentroidFold is based on γ -centroid estimators [8], a type of maximizing expected accuracy (MEA) estimators. The γ -centroid estimators generally have better expected accuracy than the minimum free energy (MFE) predictions, in terms of the number of true positives and true negatives of base pairs [7]. On average, it is known that CentroidFold achieves better performance with respect to accurate predictions of base pairs than other tools [9].

CentroidFold can utilize one of the following models as inference engine of secondary structure: McCaskill model [10] with Turner 2004 energy parameters [11] that were determined experimentally; McCaskill model with Boltzmann likelihood (BL) parameters [12] obtained by retraining energy parameters using machine learning procedures; and CONTRAfold model [13] based on a machine learning based model called conditional random fields (CRFs). The interface allows us to select from these energy models. In CentroidFold, ViennaRNA package [14] is utilized to compute BPP matrix for McCaskill model. User can specify

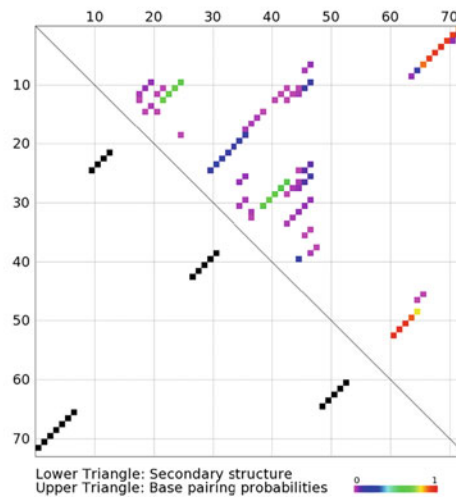
CentroidFold

Secondary structure



Download: [TEXT](#) [PNG](#) [PDF](#) [EPS](#)

Base pairing probability plot



Download: [TEXT](#) [PNG](#) [PDF](#) [EPS](#)

: specify the inference engine
 : weight of base pairs

Fig. 3 The analysis result of CentroidFold on the result page

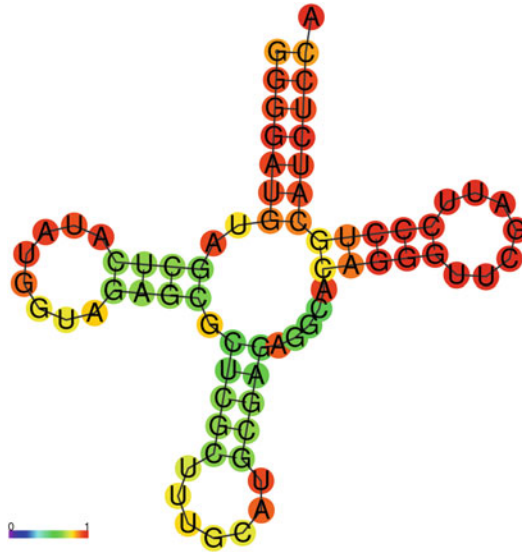


Fig. 4 The secondary structure of tRNA predicted by CentroidFold

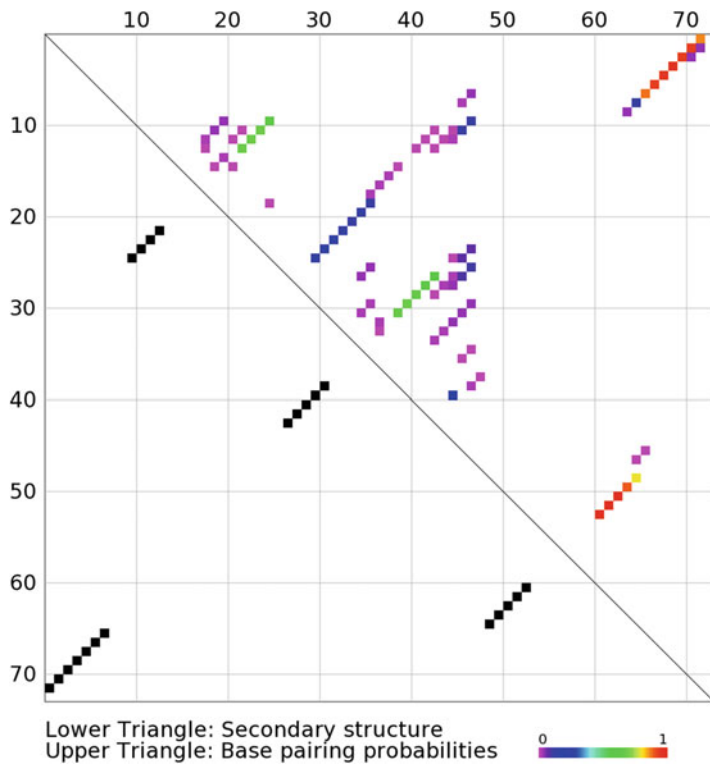


Fig. 5 The base-pairing probability matrix of Fig. 4 structure

weight of base pair, which is the rate of true base pairs in each decomposed substructure. In general, increasing the weight is expected to increase the predictive sensitivity, and decreasing the weight is expected to increase the positive predictive value.

3.2 *CentroidHomfold*

CentroidHomfold [15] predicts RNA secondary structures by employing automatically collected homologous sequences of the target. Homologous sequences are collected from Rfam [16] using LAST [17]. If homologous sequences are available, CentroidHomfold can predict secondary structures for the target sequence more accurately than CentroidFold using homologous sequence information with the probabilistic consistency transformation for BPPs.

As inference engine of secondary structure, CentroidHomfold can utilize the same models as CentroidFold. As inference engine of pairwise alignment, CentroidHomfold can utilize one of the following models: CONTRAlign model [18] based on a probabilistic model known as a pair CRF and ProbCons model [19] that is a pair-hidden Markov model-based progressive alignment algorithm based on probabilistic consistency. Weight of base pair can be specified in the same way as CentroidFold. In addition, user can specify E-value and the number of homologous sequences as parameters of alignment to Rfam.

3.3 *IPknot*

IPknot [20] predicts RNA secondary structures including a wide class of pseudoknots, while CentroidFold predicts only pseudoknot-free RNA secondary structures. IPknot offers secondary structure as analysis result, the figure of which is created by VARNA [21] (Fig. 6).

IPknot runs fast and predicts the MEA structure using integer programming (IP) with threshold cut. In general, secondary structure prediction including pseudoknots is more computationally expensive than pseudoknot-free prediction. While the time complexity of IPknot is, at worst, not polynomial order, it has been empirically proven that IPknot is fast enough for long RNA sequences. IPknot can also predict the consensus secondary structure when a multiple alignment of RNA sequences is given. IPknot's original server (<http://rtips.dna.bio.keio.ac.jp/ipknot/>) provides the interface of this prediction, but rtools does not.

IPknot can utilize the same model as CentroidFold, and in addition, NUPACK (Dirks-Pierce model) [22] can also be utilized. In IPknot, ViennaRNA package is utilized to compute BPP matrix for McCaskill model. Weight of base pair can be specified in the same way as CentroidFold.

3.4 *Rchange*

Rchange [5] computes entropy and internal energy changes of secondary structures for single-point mutated sequences. Rchange offers a figure which shows the upper and lower bound of the relative changes of the ensemble free energy (Fig. 7). The upper bound values indicate the largest energy increase caused by a single

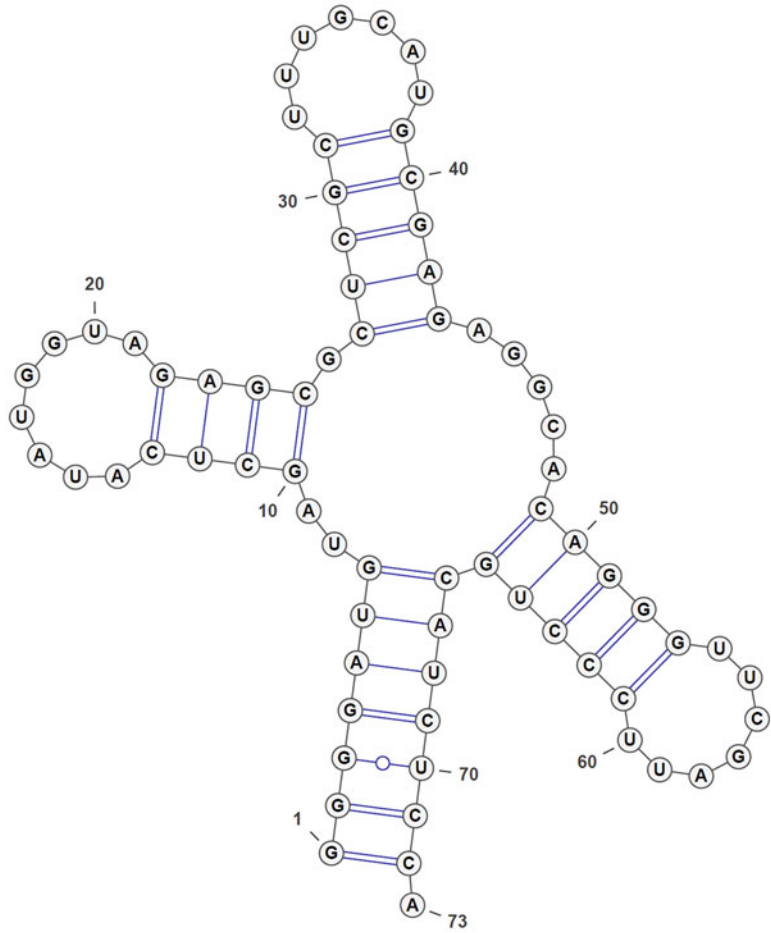


Fig. 6 The secondary structure of tRNA predicted by IPknot using VARNA

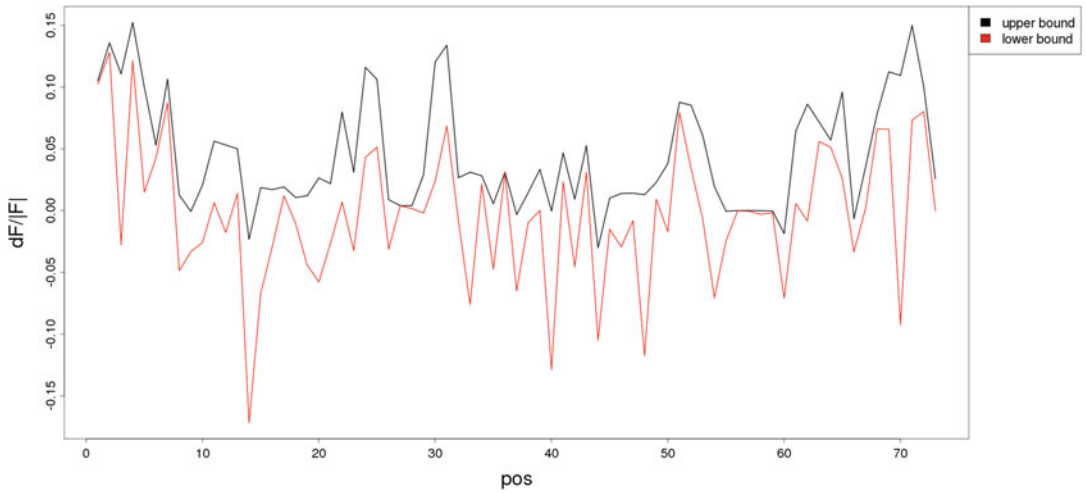


Fig. 7 The upper and lower bound of the relative changes of the ensemble free energy by single mutations of tRNA computed by Rchange

mutation. The lower bound values indicate the largest energy decrease caused by a single mutation. Rchange's original output is in text format and can be downloaded. In the output, entropy changes, internal energy changes, and Helmholtz free energy changes by three mutations (e.g., T, G, C for A) at each position are summarized in one line.

User can specify maximal span of base pairs. The maximal span constraint avoids predicting large number of spurious base pairs between distant positions while still allowing to predict global structures and reducing the computational complexities which are $O(NW^2)$ (where N is sequence length and W is maximal span of base pairs) for single mutations [5].

3.5 CapR

CapR [1] computes probabilities that each RNA base position is located within each secondary structural context for long RNA sequences. CapR offers two figures which show a structural profile of an RNA base as a set of six probabilities that the base belongs to each context (Fig. 8). The six contexts of RNA structures are (i) bulge loop, (ii) exterior loop, (iii) hairpin loop, (iv) internal loop, (v) multibranch loop, and (vi) stem part. The first figure shows each probability, and the second figure combines similar contexts (Bulge+Internal, Exterior+Multibranch).

CapR efficiently computes marginal probabilities that each RNA base position is located within each secondary structural context. Those marginal probabilities are computed according to a probability distribution of secondary structures of a given RNA sequence, provided by McCaskill (energy) model. The computational complexities of CapR is also $O(NW^2)$ [1].

User can specify maximal span of base pairs in the same way as specified in Rchange.

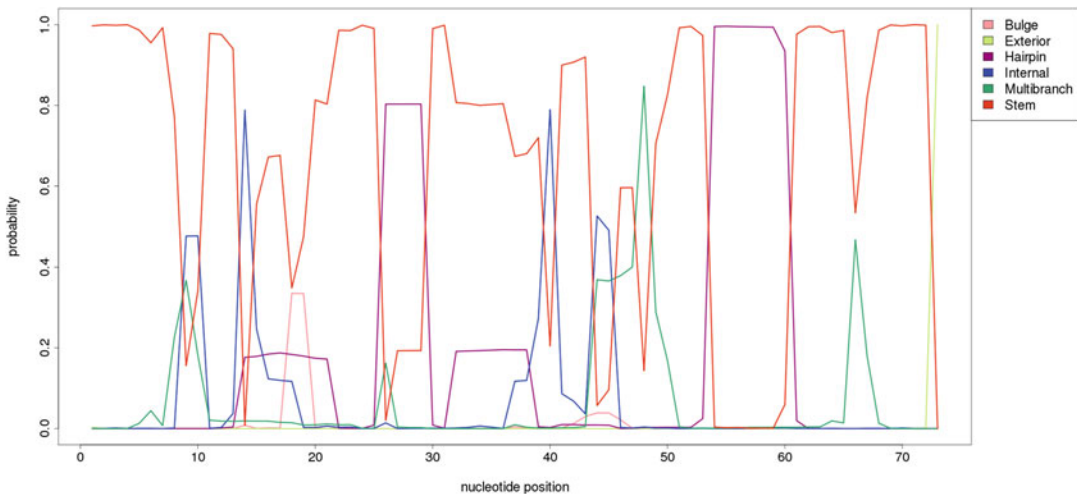


Fig. 8 The structural profile of each RNA base of tRNA as a set of six probabilities that the base belongs to each context computed by CapR

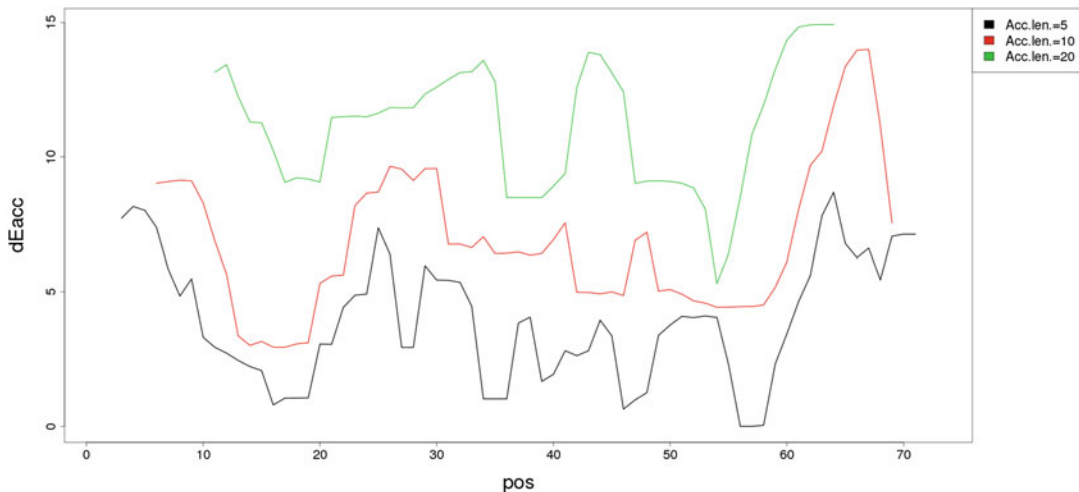


Fig. 9 The accessibility for each RNA base of tRNA computed by CapR

3.6 Raccess

Raccess [4] computes the accessibility of segment $[x, x + l - 1]$ in the transcript for all the positions x with access length $l = 5, 10, 20$. The thermodynamic energy that is required to keep range $[a, b]$ being accessible is given by:

$$\text{access_energy}([a, b]) = -RT \log(\text{prob}([a, b])),$$

where

$$\text{prob}([a, b]) = \frac{\sum_{s \in S_{[a, b]}} e^{-E(s)/RT}}{\sum_{s \in S_0} e^{-E(s)/RT}}.$$

where S_0 is all the possible secondary structures of the transcript and $S_{[a, b]}$ is all the secondary structures having range $[a, b]$ as a loop region. Raccess offers a figure (Fig. 9), in which each $\text{access_energy}([x, x + l - 1])$ is plotted at $x + 1/2$.

In Raccess, McCaskill model is utilized for a probability distribution of secondary structures. The accessibility for every nucleotide is computed by using probability of being unpaired, which is easily calculated by BPP matrix. The computational complexities of Raccess is also $O(NW^2)$ [4].

User can specify maximal span of base pairs in the same way as specified in Rchange.

3.7 RintD

RintD [2] computes the exact probability distributions of integer-valued features on the energy model of RNA secondary structures. The target secondary structures are predicted by CentroidFold (*structure 1*) and RNAfold [14] (*structure 2*) from the RNA sequence entered by the user, and RintD offers two figures as analysis results. The first figure (Fig. 10) shows the probability that RNA folds into a structure which has each Hamming distance from the reference structure (*structure 1*). $\alpha\%$ credibility limit

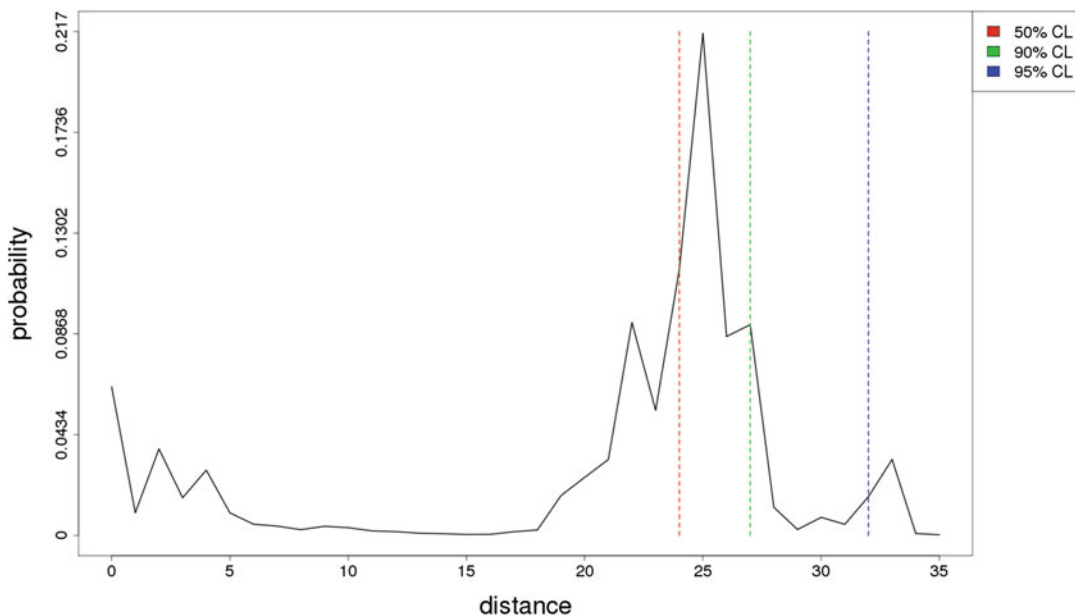


Fig. 10 The probability distribution of the secondary structure over the Hamming distance from the canonical structure is computed by RintD

(CL) is defined as the minimum Hamming distance radius that contains $\alpha\%$ of the possible RNA secondary structures. Note that the smaller CL indicates that the reference structure is more stable (reliable) in the secondary structures. The second figure (Fig. 11) shows the distribution of Hamming distance from two reference structures (*structure 1 and 2*) in a 2D landscape.

User can enter RNA secondary structures as the target. For the prediction of RNA secondary structure by CentroidFold, user can specify weight of base pairs. However, the energy model (McCaskill model) cannot be changed, because RintD utilizes McCaskill model to compute the distribution of RNA secondary structures [2].

3.8 RintW

This analysis detects essential alternative secondary structures from an RNA sequence by decomposing BPP matrix. The decomposition is calculated by RintW [3], which efficiently computes the BPP distributions on the Hamming distance from a canonical secondary structure.

First, this analysis computes a canonical secondary structure by CentroidFold from the RNA sequence. Second, the probability distribution of the secondary structure over the Hamming distance from the canonical structure is computed by RintD (Fig. 12). If there are multiple peaks in the distribution, which does not guarantee but implies that there are multiple clusters of secondary structures, the ranges of Hamming distance of these clusters are

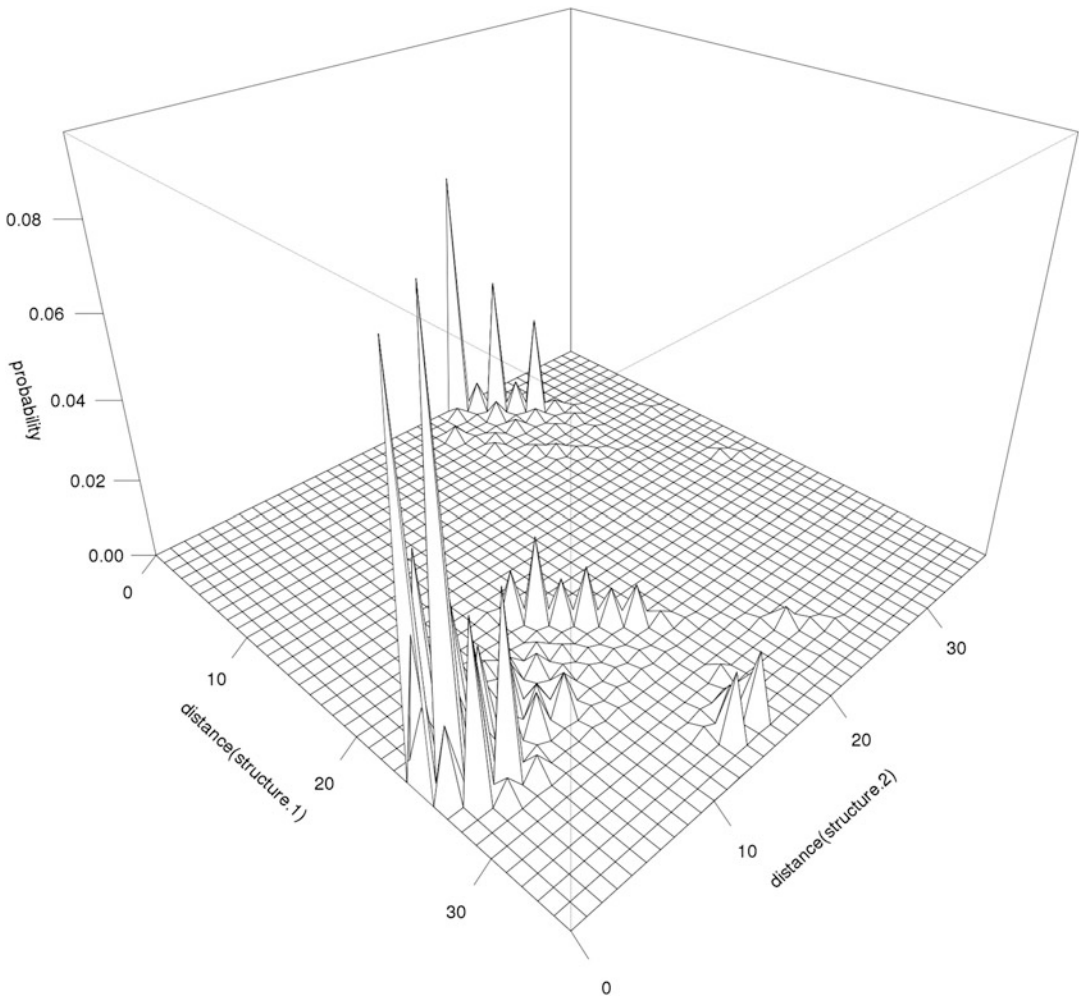


Fig. 11 The distribution of Hamming distance from two reference structures in a 2D landscape

detected. In Fig. 12, there are three clusters. Third, the BPP matrix of each range of Hamming distance is computed by RintW. The representative secondary structure for each cluster is estimated by posterior decoding of the corresponding base-pairing matrix (Fig. 13). Finally, for each pair of these secondary structures, 2D distribution of Hamming distance from the two structures is computed by RintD.

User can specify CentroidFold options to compute a canonical secondary structure from the RNA sequence.

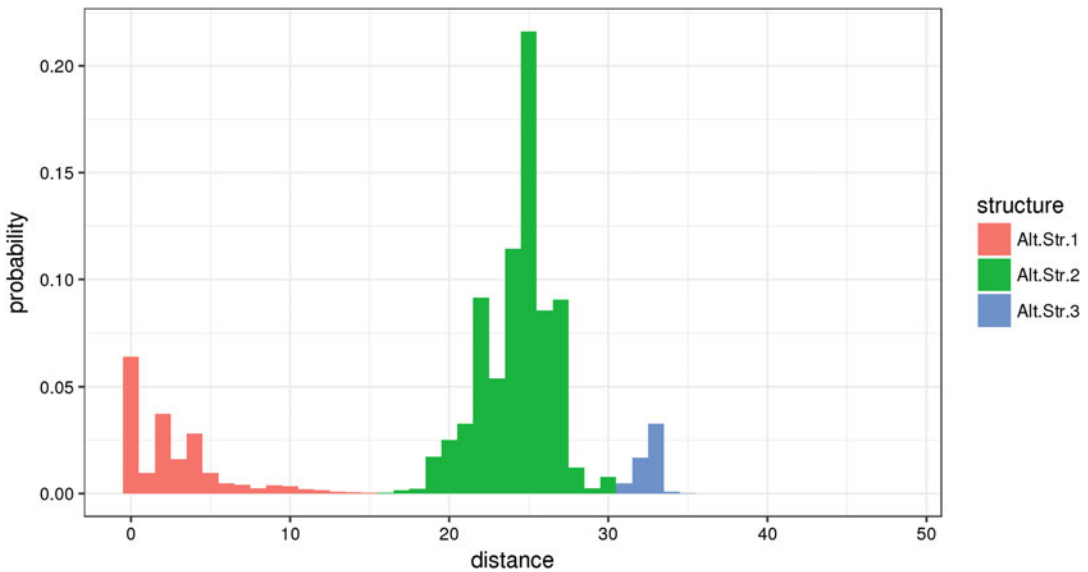


Fig. 12 The probability distribution of the secondary structure over the Hamming distance from the canonical structure is computed by RintD. This distribution has three peaks and is divided into three clusters

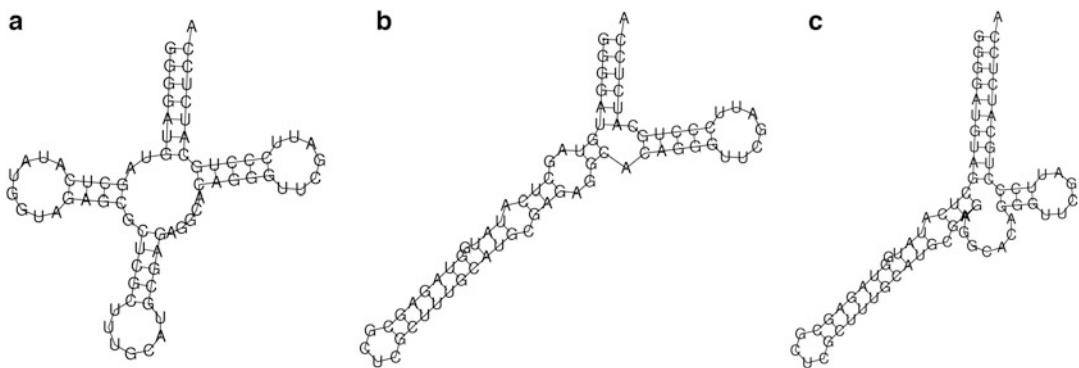


Fig. 13 The representative secondary structures computed for each cluster by RintW

References

1. Fukunaga T, Ozaki H, Terai G, Asai K, Iwasaki W, Kiryu H (2014) CapR: revealing structural specificities of RNA-binding protein target recognition using CLIP-seq data. *Genome Biol* 15(1):R16
2. Mori R, Hamada M, Asai K (2014) Efficient calculation of exact probability distributions of integer features on RNA secondary structures. *BMC Genomics* 15(S10):S6
3. Hagio T, Sakuraba S, Iwakiri J, Mori R, Asai K (2018) Capturing alternative secondary structures of RNA by decomposition of base-pairing probabilities. *BMC Bioinf* 19(1):85–95
4. Kiryu H, Terai G, Imamura O, Yoneyama H, Suzuki K, Asai K (2011) A detailed investigation of accessibilities around target sites of siRNAs and miRNAs. *Bioinformatics* 27(13):1788–1797
5. Kiryu H, Asai K (2012) Rchange: algorithms for computing energy changes of RNA secondary structures in response to base mutations. *Bioinformatics* 28(8):1093–1101
6. Hamada M, Ono Y, Kiryu H, Sato K, Kato Y, Fukunaga T et al (2016) Rtools: a web server for various secondary structural analyses on single RNA sequences. *Nucleic Acids Res* 44(W1):W302–W307

7. Hamada M, Kiryu H, Sato K, Mituyama T, Asai K (2009) Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics* 25(4):465–473
8. Hamada M, Kiryu H, Iwasaki W, Asai K (2011) Generalized centroid estimators in bioinformatics. *PLoS One* 6(2):e16450
9. Puton T, Kozłowski LP, Rother KM, Bujnicki JM (2013) CompaRNA: a server for continuous benchmarking of automated methods for RNA secondary structure prediction. *Nucleic Acids Res* 41(7):4307–4323
10. McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers Orig Res Biomol* 29(6–7):1105–1119
11. Turner DH, Mathews DH (2010) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res* 38(suppl_1):D280–D282
12. Andronescu MS, Pop C, Condon AE (2010) Improved free energy parameters for RNA pseudoknotted secondary structure prediction. *RNA* 16(1):26–42
13. Do CB, Woods DA, Batzoglou S (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* 22(14):e90–e98
14. Lorenz R, Bernhart SH, Zu Siederdisen CH, Tafer H, Flamm C, Stadler PF, Hofacker IL (2011) ViennaRNA package 2.0. *Algorithms Mol Biol* 6(1):26
15. Hamada M, Sato K, Kiryu H, Mituyama T, Asai K (2009) Predictions of RNA secondary structure by combining homologous sequence information. *Bioinformatics* 25(12):i330–i338
16. Kalvari I, Argasinska J, Quinones-Olvera N, Nawrocki EP, Rivas E, Eddy SR et al (2018) Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. *Nucleic Acids Res* 46(D1):D335–D342
17. Kielbasa SM, Wan R, Sato K, Horton P, Frith MC (2011) Adaptive seeds tame genomic sequence comparison. *Genome Res* 21(3):487–493
18. Do CB, Gross SS, Batzoglou S (2006) CONTRAlign: discriminative training for protein sequence alignment. In: Annual international conference on research in computational molecular biology. Springer, Berlin/Heidelberg, pp 160–174
19. Do CB, Mahabhashyam MS, Brudno M, Batzoglou S (2005) ProbCons: probabilistic consistency-based multiple sequence alignment. *Genome Res* 15(2):330–340
20. Sato K, Kato Y, Hamada M, Akutsu T, Asai K (2011) IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming. *Bioinformatics* 27(13):i85–i93
21. Darty K, Denise A, Ponty Y (2009) VARNA: interactive drawing and editing of the RNA secondary structure. *Bioinformatics* 25(15):1974
22. Zadeh JN, Steenberg CD, Bois JS, Wolfe BR, Pierce MB, Khan AR et al (2011) NUPACK: analysis and design of nucleic acid systems. *J Comput Chem* 32(1):170–173



Linear-Time Algorithms for RNA Structure Prediction

He Zhang, Liang Zhang, Kaibo Liu, Sizhen Li, David H. Mathews,
and Liang Huang

Abstract

RNA secondary structure prediction is widely used to understand RNA function. Existing dynamic programming-based algorithms, both the classical minimum free energy (MFE) methods and partition function methods, suffer from a major limitation: their runtimes scale cubically with the RNA length, and this slowness limits their use in genome-wide applications. Inspired by incremental parsing for context-free grammars in computational linguistics, we designed linear-time heuristic algorithms, LinearFold and LinearPartition, to approximate the MFE structure, partition function and base pairing probabilities. These programs are orders of magnitude faster than Vienna RNAfold and CONTRAfold on long sequences. More interestingly, LinearFold and LinearPartition lead to more accurate predictions on the longest sequence families for which the structures are well established (16S and 23S Ribosomal RNAs), as well as improved accuracies for long-range base pairs (500+ nucleotides apart). This chapter provides protocols for using LinearFold and LinearPartition for secondary structure prediction.

Key words RNA secondary structure prediction, Linear-time heuristic, Beam search approximation, Minimum free energy structure, Maximum expected accuracy structure, Partition function

1 Introduction

RNAs are involved in multiple processes [2, 8, 9], and their functionalities are highly related to structures. However, determining the structure using experimental methods [23, 38, 40] is costly and time-consuming. Therefore, fast and accurate computational prediction of RNA structure is desired, among which predicting secondary structure, defined as the set of all canonical base pairs (A–U, G–C, G–U), is an important and challenging problem [10, 32]. Free energy minimization-based methods [29, 43] were first proposed to predict the structure with minimum free energy (MFE) in 1980s. There was a subsequent shift to partition function-based methods [28] that account for folding ensembles and can, therefore, estimate structure and base pair probabilities. All these methods are based on bottom-up dynamic programming

and run in cubic time, thus are slow on long sequences, i.e., sequences of thousands of nucleotides.

LinearFold and LinearPartition aim to overcome the slowness bottleneck. LinearFold is an linear-time, linear-space, approximate algorithm for predicting the MFE structure, using incremental dynamic programming plus beam search, and providing options of both machine-learned and thermodynamic models [12]. LinearFold software is a Linux-oriented, command line interface software, providing the services of computing the MFE structure of a given RNA sequence, computing the constrained structure if constraints are specified, and evaluating the free energy change of a given RNA sequence and its structure. LinearPartition extends LinearFold to linear-time partition function and base pair probability calculation, and is also implemented as a command line interface software [39]. LinearPartition is able to output the free energy of the thermodynamic ensemble, the base pair probability matrix and the maximum expected accuracy (MEA) structure [7, 15].

1.1 Free Energy Minimization

Commonly, the minimum free energy (MFE) structure is predicted [29, 43]. Widely used systems such as RNAstructure [30], Vienna RNAfold [21], and CONTRAfold [7] all use similar dynamic programming algorithms [36, 43] to find the optimal secondary structure with minimum free energy. Though the implementations are optimized to make them faster, this set of algorithms have a running time of $O(n^3)$ that scales *cubically* with the sequence length n , which is too slow for long RNA sequences [17].

To help accelerate prediction speed, some systems provide an option to do local folding, for example, `RNAfold --maxBPspan` allows predicting base pairs shorter than a given threshold. Local folding algorithms are still based on bottom-up dynamic programming (DP), but impose a window size constraint and fill separate DP matrices of each window. Obviously, such approximations ignore all long-distance base pairs beyond the window size threshold. However, studies have confirmed that such long base pairs are in fact common in natural RNA structures, e.g., the length of the longest base pair grows nearly linearly with sequence length [19], and the physical distance between the 5' and 3' ends in folded structures is short and nearly constant [5, 16, 18, 37].

Unlike bottom-up global and local folding based algorithms, LinearFold accelerates the parsing process by scanning the RNA sequence from 5'-to-3', using a left-to-right dynamic program that runs in $O(n^3)$ time, and applying the beam pruning heuristic [11], which keeps only the top b highest-scoring (i.e., lowest energy) states for each prefix of the input sequence, to achieve linear runtime in practice. Figure 1a and b give the runtime comparisons of LinearFold and two baseline systems, Vienna RNAfold and CONTRAfold, on the relatively short ArchiveII set [33] and the longer RNAcentral set [1]. We confirm that LinearFold runs in linear time

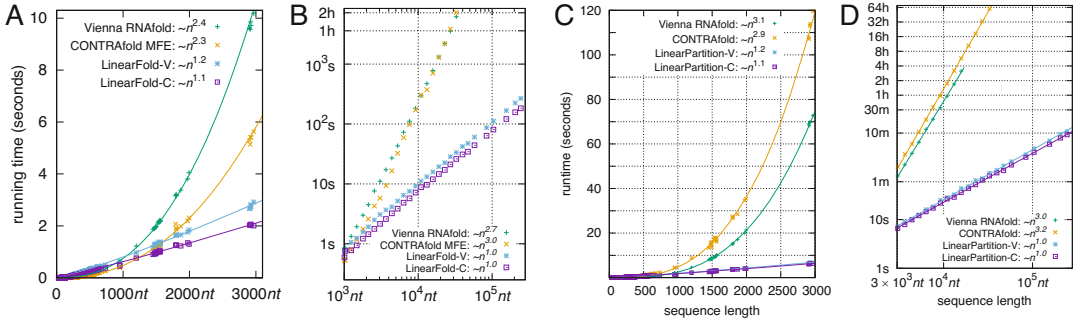


Fig. 1 Efficiency and scalability of LinearFold and LinearPartition. **(a)** and **(b)**: Runtime comparisons on the Archivell dataset **(a)** and the RNACentral dataset **(b, log-log)** between the two baselines, CONTRAfold MFE & Vienna RNAfold and LinearFold **(c)** and **(d)**: Total runtime comparisons of computing both the partition function and base pairing probabilities on the Archivell dataset **(c)** and **(d, log-log)** between the two baselines (both in partition function mode) and LinearPartition

while the other two run in cubic time, and LinearFold is much faster than the baselines on long sequences (all systems are run on Linux with 3.40 GHz Intel Xeon E3-1231 CPU and 32G memory). It is important to note that, LinearFold achieves linear runtime (as well as linear space) without imposing constraints on the base pair distance, i.e., it is a *global* folding algorithm. Interestingly, LinearFold even has higher accuracies for long-range base pairs (500 + *nt* apart).

In addition to the MFE structure, it is important to report suboptimal structures, i.e., structures with low folding free energy change. These structures are important alternative hypotheses for the structure. LinearFold implements Zuker’s suboptimal algorithm [42], and thus can provide diverse suboptimal structures.

1.2 Partition Function and Base Pair Probabilities

At equilibrium, the MFE structure is the most populated structure, but it is a simplification because multiple conformations can exist as an equilibrium ensemble for RNA sequences [16, 24]. Instead of predicting the MFE structure, we can compute the partition function, which is the sum of the equilibrium constants for all possible secondary structures, and is the normalization term for calculating the probability of a secondary structure in the Boltzmann ensemble. The partition function $Q(\mathbf{x})$ is defined as:

$$Q(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} e^{-\frac{\Delta G^\circ(\mathbf{x}, \mathbf{y})}{RT}} \quad (1)$$

where $\Delta G^\circ(\mathbf{x}, \mathbf{y})$ is the conformational Gibbs free energy change of RNA sequence \mathbf{x} and one of its structures \mathbf{y} , among all possible structures $\mathcal{Y}(\mathbf{x})$ [35]. R is the universal gas constant, and T is the absolute temperature.

The partition function calculation can also be used to calculate base pairing probabilities of each nucleotide i paired with each of possible nucleotides j by summing the probabilities of each structure that contains the base pair of i and j [24, 28]:

$$p_{i,j} = \sum_{(i,j) \in \text{pairs}(\mathbf{y})} p(\mathbf{y}) \quad (2)$$

Base pairing probabilities can be used for downstream prediction methods, such as maximum expected accuracy (MEA) [7, 15], an ensemble structure with improved accuracy compared with the MFE structure [22]. Other downstream prediction methods, such as ProbKnot [3], ThreshKnot [41], DotKnot [34] and IPknot [31], use base pairing probabilities to predict pseudoknotted structures with heuristics. Additionally, the partition function is the basis of stochastic sampling, in which structures are sampled with their probability of occurring in the Boltzmann ensemble [6, 25].

Therefore, there has been a shift from the classical MFE-based methods to partition function-based ones. Most of the prediction engines, such as RNAstructure [27], Vienna RNAfold [21] and CONTRAfold [7], provide partition function and base pair probability computation based on the cubic algorithms that McCaskill (1990) pioneered [28]. This employs dynamic programming to capture all possible (exponentially many) nested structures, but the $O(n^3)$ runtime scales poorly for longer sequences. This slowness is even more severe than the $O(n^3)$ -time MFE-based algorithms due to a much larger constant factor. For instance, for *H. pylori* 23S rRNA (sequence length 2968 nt), Vienna RNAfold’s computation of the partition function and base pairing probabilities is 9× slower than MFE (71 vs. 8 s), and CONTRAfold is even 20× slower (120 vs. 6 s).

Inspired by the efficient linearization by LinearFold, LinearPartition alleviates this cubic-factor slowness by providing a linear-time approximate calculation of partition function and base pair probabilities [39]. Although it is approximate, the well-designed heuristic ensures the surviving structures capture the bulk of the free energy of the ensemble. Figure 1c and d show the runtime comparison of LinearPartition with two baselines. Similar to the comparison between LinearFold and its competing systems, LinearPartition runs in linear time and is much faster than Vienna RNAfold and CONTRAfold (partition function mode). LinearPartition is 2771× faster than CONTRAfold for the longest sequence (32,753 nt) that CONTRAfold can run in the dataset (2.5 days vs. 1.3 min.). Interestingly, LinearPartition is orders of magnitude faster *without* sacrificing accuracy. In fact, the resulting base pairing probabilities are even better correlated with ground truth structures, and when applied to downstream structure prediction tasks, they lead to small accuracy improvements on longer families (small and large subunit

rRNAs), as well as a substantial accuracy improvement on long-distance base pairs.

It is worth noticing that local partition function computations, such as RNAplfold [4] and Rfold [14], also achieve linear runtime but can only consider pairs up to a fixed window size, and cannot output the partition function or Boltzmann probabilities. More importantly, RNAplfold outputs averaged local base pairing probabilities, leading to a problem that the base pairing probabilities do not normalize, i.e., for some nucleotides the probability of being base paired is greater than 1. LinearPartition is the first to achieve linear runtime without limitations on pair distance, and the base pairing probabilities are well-defined in range [0,1]. In addition, LinearPartition can output an approximation of partition function and Boltzmann probabilities, making it possible to do stochastic sampling.

2 Web Server

We provide LinearFold and LinearPartition web server for users who prefer a graphical interface. The web server is available at <http://linearfold.org/>.

2.1 Web Server of LinearFold

Figure 2 shows the screenshot of the interactive input window of LinearFold web server. The RNA sequence length limit on the web server is 100,000, which is 10× larger than the Vienna RNAfold web server. To start prediction, users can paste or type the RNA sequence in the input box, or upload a local file in FASTA format by clicking the “Choose File” button. The green “Sample” menu button is for choosing pre-loaded RNA sequences, including example sequences of tRNA, SRP RNA, 16S and 23S rRNAs. Users can set the beam size by entering a number in “Beam size” box. Here we provide the beam size ranging from 1 to 200. LinearFold-V is the default folding engine, which uses the thermodynamic model, while LinearFold-C, based on the CONTRAfold machine-learned model, is also available by clicking “LinearFold-C” in the “available model(s)” box.

When the settings are complete, the user clicks the “Run LinearFold” button, the computation starts and the page is refreshed to render the prediction results.

The predicted structures are visualized with circular plots. For example, in Fig. 3, the predicted structures of the *O. sativa* SRP RNA (311 nt) are shown. The base pairs predicted by both LinearFold-C and LinearFold-V are in blue, while base pairs predicted by LinearFold-C only are in red and by LinearFold-V only are in green. We can see that LinearFold-C and LinearFold-V agree

Interactive demo

Add a sequence

Paste or type your sequence here (length < 100,000):

```
>name
GCUCUGUUGGUGUAGUCCGGCCAAUCAUAUACCCUCU
```

Samples ▼

Or upload a file in FASTA format:

Choose File No file chosen...

Arguments

Beam size (1-200):

Constraints (one more line as structure constraints)

Zuker suboptimal structures

Available model(s)

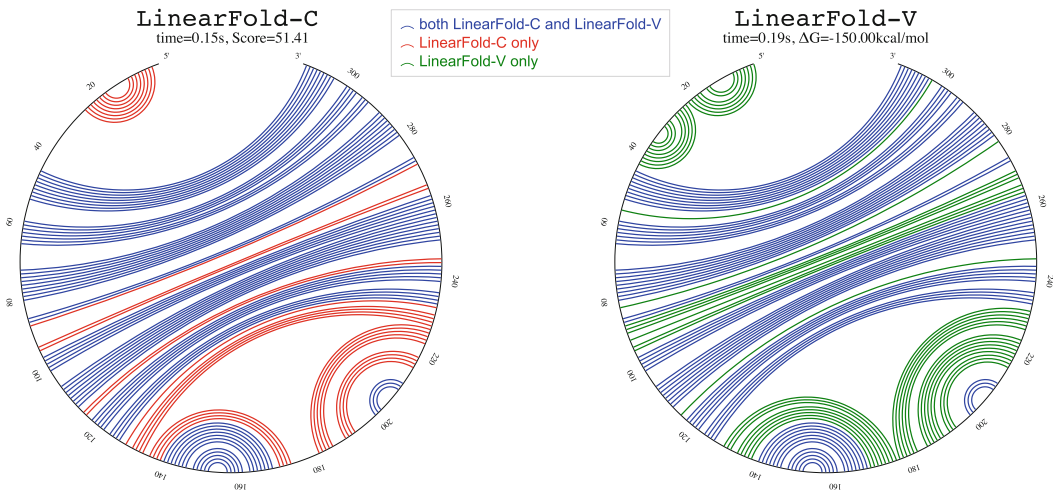
LinearFold-C (using [CONTRFold v2.0](#) machine-learned model, Do et al 2006)

LinearFold-V (using [Vienna RNAfold](#) thermodynamic model, Lorenz et al 2011, with parameters from Mathews et al 2004)

Run LinearFold >>

Fig. 2 The input page of LinearFold

Sequence name: **SRP_Oryz.sati_AC079888** (len:311)
Beam Size: 100 LinearFold-C: 0.15s LinearFold-V: 0.19s



[forna](#) (Kerpedjiev et al 2015) display:

Fig. 3 The screenshot of the LinearFold result page, showing the base pairing circular plots of *O. sativa* SRP RNA. In these plots, the backbone is arranged on a circle and base pairs are lines connecting nucleotides inside the circle

on the majority of base pairs, and LinearFold-V tends to predict more base pairs than LinearFold-C. Additional information, such as beam size, runtime and free energy change of the structure (model score for LinearFold-C), is also available on the page. For users who prefer “forna [13]” display, the page also provides a button to render them. Exact predicted results (in dot format) are also available on this page.

The web server also allows calculating diverse suboptimal structures. To enable this feature, check `zucker suboptimal structures` in the input box, and specify the maximum energy gap from the MFE structure (the value is 5 kcal/mol for LinearFold-V and 5 model scores for LinearFold-C by default). We also implement the Window feature [26], to ensure the suboptimal structures are more dissimilar. Users can download the suboptimal structures from the results page in dot notation as plain text.

If there are specified constraints, users can check the “constraints” box and add one more line for the customized constraints after the RNA sequence in the input box. It allows four types of constraints, “?”, “.”, “(” and “)”, each indicates a position for which the proper matching is unknown, unpaired, left or right parenthesis, respectively. The left parentheses is the 5' nucleotide in a base pair and the right parentheses is the 3' nucleotide in a base pair. The left and right parentheses need to be balanced so that both nucleotides in base pairs are specified. Additionally, no pseudoknotted pairs can be specified [20]. On the results part, the constrained base pairs are shown with dash arcs in the circular plots.

2.2 Web Server of LinearPartition

The input page of LinearPartition is similar as LinearFold. Figure 4 shows the screenshot of LinearPartition results page, using *O. sativa* SRP RNA as an example. In the two upper circular plots, the base pairs are represented by blue arcs, and the darkness of the arcs stands for magnitude of their corresponding base pairing probabilities. When the mouse floats on an arc, the information of this arc, the end indices of the arc and its corresponding base pairing probability, is shown. For example, if the information “[67, 291], p=0.04” is shown when the mouse floats on an arc, it means that the base pair between nucleotides 67 and 291 has a small pairing probability of 0.04.

In the two lower circular plots, the base pairs in the predicted MEA structures are shown. As in the upper circular plots, the probability of each base pair is represented by its darkness. The set of arcs in the MEA plots is a subset of probability plots, and there are no crossing arcs (i.e., pseudoknotted pairs) in the MEA plots.

Sequence name: SRP_Oryz.sati_AC079888 (len:311)

Beam Size: 100

LinearPartition-C: Partition Function: 0.19s Base Pairing Probabilities: 0.16s

LinearPartition-V: Partition Function: 0.21s Base Pairing Probabilities: 0.16s

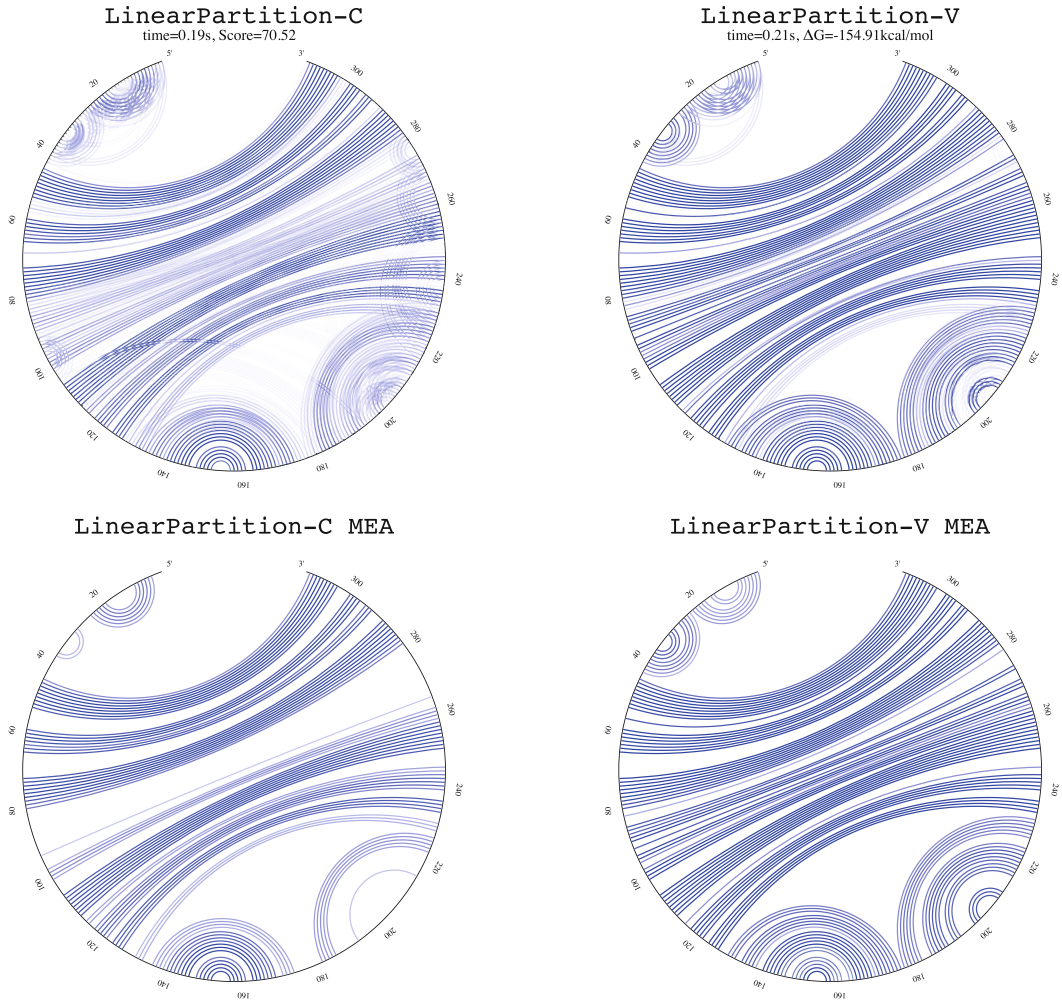


Fig. 4 The screenshot of the LinearPartition result page, showing the circular plots of base pairing probabilities and the MEA structures of *O. sativa* SRP RNA using LinearPartition-C and LinearPartition-V. The grayscale weight of the base pair lines indicates the pairing probability. The pairing partners and the pairing probability can be obtained by putting the mouse pointer over a base pair line

3 Command Line Protocols

3.1 Installation

The source code of LinearFold and LinearPartition is available online. A license is provided, allowing free use for academic purpose. The software are oriented for Unix/Linux and Mac platforms, but can also run on Microsoft Windows using the Linux bash shell. To use LinearFold, make sure that the dependencies,

GCC 4.8.5 or above (or clang on Mac), and Python 3, are correctly installed on the machine. If needed, GCC can be downloaded from <https://gcc.gnu.org/> and Python 3 can be downloaded from <https://www.python.org/downloads/>.

When all dependencies are ready, download the latest version of LinearFold from the GitHub repository <https://github.com/LinearFold/LinearFold>, and unzip the downloaded file. If a zip file was downloaded, in the command line interface use the command:

```
tar -xzvf LinearFold-master.zip
```

The source file has a Makefile for easy compilation, so just go to the LinearFold folder and type the command:

```
make
```

While compiling, it outputs information as below, indicating the compilation is successful:

```
chmod+linearfold
mkdir-pbin
g++src/LinearFold.cpp-std=c++11-O3-Dlv
-Dis_cube_pruning-Dis_candidate_list-o bin/linearfold_v
g++src/LinearFold.cpp-std=c++11-O3-Dis_cube_pruning
-Dis_candidate_list-o bin/linearfold_c
```

LinearPartition is available on the GitHub repository <https://github.com/LinearFold/LinearPartition>. The dependencies, installation and compilation of LinearPartition are the same as LinearFold, so the instructions above are identical for LinearPartition.

3.2 Run LinearFold for Minimum Free Energy Structure

As Vienna RNAfold, LinearFold accepts both single sequence and multiple sequences as input. To predict a single sequence, e.g., CCCCAGGGGG, users can input the sequence directly in the terminal and run LinearFold. The input command and the corresponding outputs are as follows:

```
$echoCCCCCAGGGGG|./linearfold
CCCCCAGGGGG
((((...))) (0.74)
```

The outputs take two lines, where the first line is the given RNA sequence and the second line is the predicted structure and its corresponding score (LinearFold-C) or free energy change (LinearFold-V).

Users can also save the sequence into a file and feed the file to LinearFold later on:

```
$catSEQUENCE|./linearfold
CCCCCAGGGGG
((((...))) (0.74)
```

For multiple sequence prediction, users can save all sequences into a file, either in FASTA format or pure-sequence format (where each sequence takes one line), and LinearFold predicts them sequentially when the file is fed in:

```
cattestseq|./linearfold
UGAGUUCUCGAUCUCUAAAAUCG
..... (-0.22)
AAAACGGUCCUUAUCAGGACCAAACA
..... ((((((.....)))))) (4.91)
AUUCUUGCUUCAACAGUGUUUGAACGGAAU
..... (-0.29)
UCGGCCACAAACACACAAUCUACUGUUGGUCGA
(((((((.....)))))) (0.99)
GUUUUUAUCUACACACGCUUGUGUAAGAUAGUUA
..... ((((((((((.....)))))))))) (6.66)
```

LinearFold integrates the machine-learned model borrowed from CONTRAfold (called LinearFold-C), and thermodynamic model from Vienna RNAfold (called LinearFold-V). By default it runs in LinearFold-C mode. To switch to LinearFold-V, add "-v" option:

```
$echoGGGCUCGUAGAUACAGCGGUAGAUUCGCUUCCUUCGCAAGGAAGCCUGGG
UUCAAAUCCAGCGAGUCCACCA|./linearfold-v
GGGCUCGUAGAUACAGCGGUAGAUUCGCUUCCUUCGCAAGGAAGCCUGGGUUCAA
AUCCAGCGAGUCCACCA
(((((((.....)))) (((((((.....))))))))).
(((((((.....)))))))) (.....) (-31.50)
```

LinearFold provides an option, “-b BEAM_SIZE”, to set beam size with an integer BEAM_SIZE for beam search approximation. Normally, a larger beam size deviates less from the exact MFE structure but takes more time. The default beam size is 100. To run without beam search approximation (no search error), use “-b 0”:

```
$echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCUGGG
UUCAAAUCCAGCGAGUCCACCA|./linearfold-V-b3
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCUGGGUCAA
AUCCAGCGAGUCCACCA
(((((((..(((.....))))(((((((...)))))))).
(((((((...)))..))))))....(-26.30)
```

To enable hairpin sharpturn in prediction, i.e., a hairpin loop with fewer than 3 unpaired nucleotides, use "--sharpturn". Note that this only works for LinearFold-C mode, since the thermodynamic model does not support hairpin sharpturn.

```
$echoCCCCCAGGGGG|./linearfold--sharpturn
CCCCCAGGGGG
((((((...)))))(0.81)
```

In addition, LinearFold is able to visualize the structure using a circular plot. For example, given the sequence and structure of *E. coli* tRNA^{Gly} in a fasta file named *ecoli_tRNA*, draw its circular plot with the Python script `draw_circular_plot`:

```
$catecoli_tRNA|./draw_circular_plot
```

Then a file named `circular_plot.pdf` is generated, showing the plot as in Fig. 5a. Another example in Fig. 5b is a longer sequence (2904 nt) of *E. coli* 23S rRNA. Note that to draw a circular plot, the sequence length should be longer than 50 and shorter than 5650.

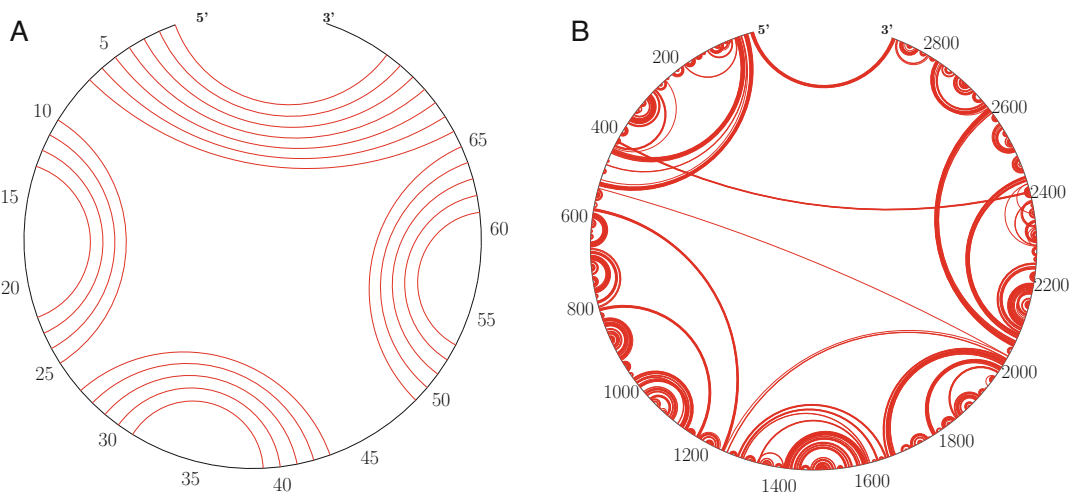


Fig. 5 The circular plots of *E. coli* tRNA^{Gly} (a) and *E. coli* 23S rRNA (b)

3.3 Run LinearFold for Suboptimal Structures

Besides outputting one single predicted structure, LinearFold can also compute suboptimal structure [42] with score/energy (5 kcal/mol for LinearFold-V and 5 model scores for LinearFold-C by default) in a certain range of the optimum for both the LinearFold-C mode and the LinearFold-V mode (for example, "--zucker --delta 4.0").

```
$echoGAACCCCGUCAGGUCCGGAAGGAAGCAGCGGUAAGU|
./linearfold-V--zucker--delta4.0
GAACCCCGUCAGGUCCGGAAGGAAGCAGCGGUAAGU
.....((((.....(((.....))).....))).....(-8.70)
Zukersuboptimalstructures...
.....((((.....(((.....))).....))).....(-8.70)
..((.....(((.....(((.....))).....))).....)(-7.40)
.....((((.....(((.....))).....))).....(-6.70)
..(((.....(((.....))).....).)).....(-5.30)
..(((.....(((.....))).....).)).....(-4.70)
```

3.4 Constrained Folding

LinearFold provides an option, "--constraints", to enable specific constraints in prediction. As on the web server, it allows four types of constraints, "?", ".", "(" and ")", standing for no constraint, unpaired, paired with a 3' nucleotide and paired with a 5' nucleotide one, respectively. The input should contain both the RNA sequence and corresponding constraint for each position:

```
$echo-e"GAACCCCGUCAGGUCCGGAAGGAAGCAGCGGUAAGU\n???????.
????????(??????.???????)"|./linearfold--constraints-V
GAACCCCGUCAGGUCCGGAAGGAAGCAGCGGUAAGU
???????.????????(??????.???????)
(.(((.....)))..(.....)).....(5.20)
```

Note that the constrained parentheses must be well-balanced and non-crossing, i.e., non-pseudoknotted.

3.5 Free Energy Evaluation

In some cases, users want to know the free energy for a specified structure of a sequence. LinearFold-V provides an option, "--eval", to help evaluate structure's free energy. Note that since LinearFold-C is not free energy-based, it does not provide the

evaluation option. As prediction mode, users can evaluate one single sequence and its structure by directly feeding them in terminal:

```
$echo-e"GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGGUCAAUAUCCAGCGAGUCCACCA\n((((((..(((.....))))
(((((((..)))))).((((.....)))))))))...."|
./linearfold--eval
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCUGGGUCAA
AUCCAGCGAGUCCACCA
((((((..(((.....))))(((((((..))))))).
((((.....))))))))).... (-31.50)
```

LinearFold supports reading from a file containing multiple pairs of sequence and structure, and evaluate each pair sequentially. The file should be in sequence-structure format, in which sequence and its corresponding structure each takes a line in order.

```
$cattesteval|head-4
UGAGUUCUCGAUCUCUAAAAUCG
.((((.....))).....
AAAACGGUCCUUAUCAGGACCAAACA
....(((((((..))))))....
$cattesteval|./linearfold--eval
UGAGUUCUCGAUCUCUAAAAUCG
.((((.....)))..... (-1.80)
AAAACGGUCCUUAUCAGGACCAAACA
....(((((((..)))))).... (-9.30)
AUUCUUGCUUCAACAGUGUUUGAACGGAU
((((((..(((.....)))))).)))) (-6.80)
UCGGCCACAACACACAUCUACUGUUGUCGA
(((((((.....)))))).... (-7.80)
GUUUUAUCUUAACACACGCUUGUGUAAGAUAGUUA
....(((((((.....)))))).... (-13.00)
```

To get a detailed free energy change for each loop, run evaluation mode with “--verbose”:

```

$echo-e"GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAG
CCCUGGGUCAAUCCAGCGAGUCCACCA\n(((((((..(((.....))))))
(((((((.....)))))))).((((.....)))))))))...." |
./linearfold--eval--verbose
Hairpinloop(13,21)CG:4.50
Interiorloop(12,22)UA;(13,21)CG:-2.40
Interiorloop(11,23)AU;(12,22)UA:-1.10
Interiorloop(10,24)GC;(11,23)AU:-2.40
Hairpinloop(32,36)UA:5.90
Interiorloop(31,37)UA;(32,36)UA:-0.90
Interiorloop(30,38)CG;(31,37)UA:-2.10
Interiorloop(29,39)CG;(30,38)CG:-3.30
Interiorloop(28,40)UA;(29,39)CG:-2.40
Interiorloop(27,41)UA;(28,40)UA:-0.90
Interiorloop(26,42)CG;(27,41)UA:-2.10
Interiorloop(25,43)GC;(26,42)CG:-3.40
Hairpinloop(49,57)GC:4.40
Interiorloop(48,58)GC;(49,57)GC:-3.30
Interiorloop(47,59)GC;(48,58)GC:-3.30
Interiorloop(46,60)UA;(47,59)GC:-2.10
Interiorloop(45,61)CG;(46,60)UA:-2.10
Multiloop(7,62)GC:1.40
Interiorloop(6,63)CG;(7,62)GC:-2.40
Interiorloop(5,64)UA;(6,63)CG:-2.40
Interiorloop(4,65)CG;(5,64)UA:-2.10
Interiorloop(3,66)GU;(4,65)CG:-2.50
Interiorloop(2,67)GC;(3,66)GU:-1.50
Interiorloop(1,68)GC;(2,67)GC:-3.30
Externalloop:-1.70
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCUGGGUUC
AAUCCAGCGAGUCCACCA
(((((((..(((.....))))))(((((((.....)))))))).
((((.....)))))))))....(-31.50)

```

3.6 Partition Function and Free Energy of Ensemble

The input requirement of running LinearPartition is the same as LinearFold, and some of the options, “-v” for switching to LinearPartition-V, “-b BEAM_SIZE” for beam size adjustment, and “--sharpturn” for enabling hairpin sharpturn, are also inherited from LinearFold and keep unchanged. To run LinearPartition for partition function (free energy of ensemble) calculation only (without base pair probabilities and MEA structure), use “-p” option:

```

echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCUGG
GUUCAAUCCAGCGAGUCCACCA|./linearpartition-p
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCUGGGUUC
AAUCCAGCGAGUCCACCA
LogPartitionCoefficient:15.88268

```

or

```
echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGUCAAUCCAGCGAGUCCACCA|./linearpartition-V-p
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
AAUCCAGCGAGUCCACCA
FreeEnergyofEnsemble:-32.14kcal/mol
```

The output takes two lines, in which the first line is the RNA sequence, and the second line is the natural logarithm partition coefficient (in LinearPartition-C) or free energy of ensemble (in LinearPartition-V).

“--verbose” provides the option to output beam size and runtime information.

```
$echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGGUCAAUCCAGCGAGUCCACCA|./linearpartition-V-p--verbose
beamsize:100
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
CAAUCCAGCGAGUCCACCA
FreeEnergyofEnsemble:-32.14kcal/mol
PartitionFunctionCalculationTime:0.01seconds.
```

3.7 Base Pairing Probabilities and MEA Structure Prediction

LinearPartition can output base pair probabilities to a single file with “-o FILE_NAME”, where “FILE_NAME” is user specified name:

```
$echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGGUCAAUCCAGCGAGUCCACCA|./linearpartition-oooutput_file
GGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
CAAUCCAGCGAGUCCACCA
FreeEnergyofEnsemble:-32.14kcal/mol
Outputtingbasepairingprobabilitymatrixtooutput_file...
Done!
```

Note that if the output file already exists in the folder, LinearPartition will stop with a warning to avoid unexpected overwriting. If users want to overwrite anyway, use the option “-r FILE_NAME” instead of “-o FILE_NAME”.

```
$echoGGGCUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGGUCAAUCCAGCGAGUCCACCA|./linearpartition-oooutput_file
WARNING:thisfilenamehasalreadybetaken.
Chooseanothernameoruse-rmode.
Exit!
```

The output file has three columns. The first column and the second column are the index (starting from 1) of paired bases, and the third column is the corresponding base pairing probabilities.

```
$catoutput_file|head-5
1689.9754e-01
1701.3048e-04
1712.4480e-04
2679.9903e-01
2685.5159e-04
```

If base pairing probabilities of multiple sequences needed to be stored in a single file, user can still use "-o FILE_NAME" option. The base pairing probabilities of each sequence will be output into the file in turns, with a blank line to separate each input sequence's probabilities.

LinearPartition also provides an option, "-c THRESHOLD", to only output base pair probabilities larger than a user specified threshold (between 0 and 1).

```
$echoGGGCUCGUAGAUCAAGCGGUAGAUCCUCCUCCGCAAGGAAGCCCU
GGGUUCAAAUCCAGCGAGUCCACCA|./linearpartition-V-routput
_file-c0.5
GGGCUCGUAGAUCAAGCGGUAGAUCCUCCUCCGCAAGGAAGCCCUGGGUU
CAAUCCAGCGAGUCCACCA
FreeEnergyofEnsemble:-32.14kcal/mol
Outputtingbasepairingprobabilitymatrixtooutput_file...
Done!
$catoutput_file|head-5
1689.9754e-01
2679.9903e-01
3669.9867e-01
4659.9991e-01
5649.9961e-01
```

As on the web server, the LinearPartition command line software also provides circular plot visualization, in which the darkness of each arc represents the probability each base pair. Figure 6 gives two plot instances of *E.coli* tRNA^{Gly} and *E.coli* 23S rRNA. The command to generate the circular plot of base pair probabilities is as follows, where `ecoli_tRNA` is a fasta file of *E.coli* tRNA^{Gly}, and `ecoli_tRNA_bpp` is the base pairing probability file generated by LinearPartition.

```
$catecoli_tRNA|./draw_bpp_plotecoli_tRNA_bpp
```

Heatmap is another commonly used plot to visualize the base pairing probabilities. LinearPartition also provides the Python script to draw a heatmap based on the probability file. For example, after generating probability file `ecoli_tRNA_bpp` of *E.coli* tRNA^{Gly} using LinearPartition, call Python script `draw_heatmap` to draw its lower triangle heatmap where the darkness represents pairing probability:

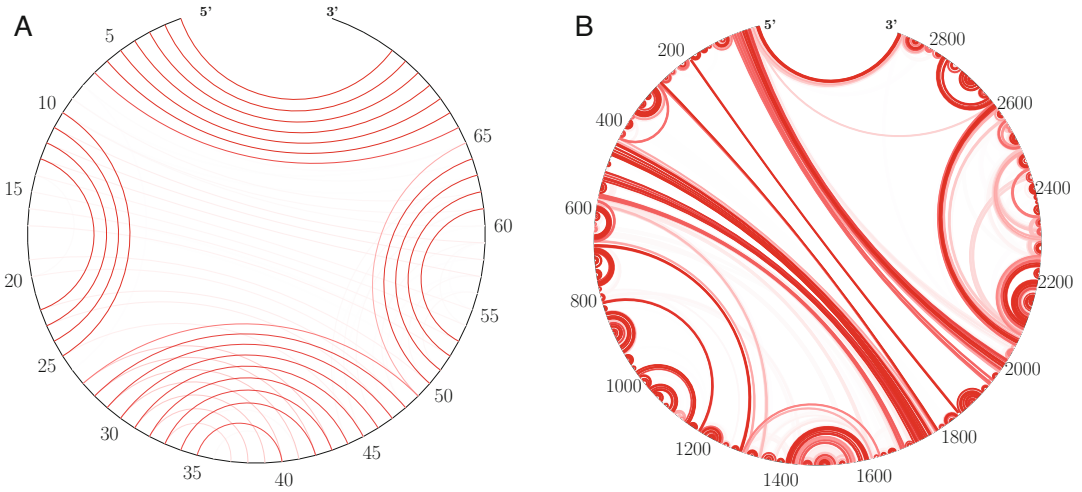


Fig. 6 The visualization of base pair probabilities using circular plots for two example sequences, *E.coli* tRNA^{Gly} (a) and *E.coli* 23S rRNA (b)

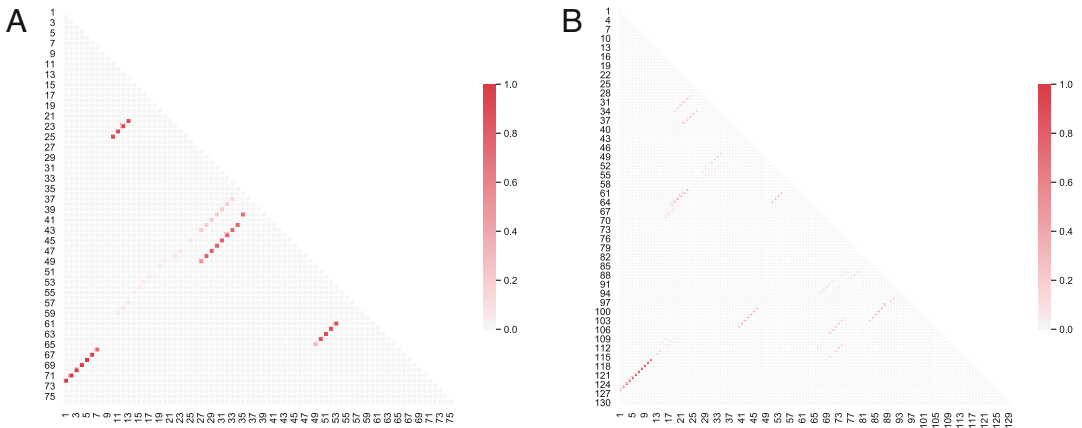


Fig. 7 The heatmaps of *E.coli* tRNA^{Gly} (a) and *M.acetivorans* 5S rRNA (b)

```
$catecoli_tRNA_bpp|. /draw_heatmap76
```

Since the probability file does not contain the information of sequence length, which is needed for drawing the heatmap, the user needs to specify it when calling the heatmap script. For instance, 76 is the length of *E.coli* tRNA^{Gly}. Then a file named `heat.pdf` is generated, showing the lower triangle heatmap as in Fig. 7a. Another example in Fig. 7b is a longer sequence (130 nt) of *M. acetivorans* 5S rRNA.

Maximum expected accuracy (MEA) [7, 15] is a partition function-based algorithm, which predicts the structure y that

maximizes the sum of the base pairing probabilities ($p_{i,j}$'s) and single-stranded probabilities (q_j 's):

$$2\gamma \sum_{(i,j) \in \text{pairs}(\mathbf{y})} p_{i,j} + \sum_{j \in \text{unpaired}(\mathbf{y})} q_j$$

The γ is a hyperparameter. It controls the balance between positive predictive value (PPV; a.k.a. precision) and sensitivity (a.k.a. recall) of the output structure.

To output the MEA structure, use "--MEA" or "-m" option. You can also set γ to a value (for example, 2.1) with "--gamma 2.1" or "-g 2.1" option. The γ is 1.0 by default.

```

$echoGGGUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCU
GGGUUCAAAUCCAGCGAGUCCACCA|./linearpartition-V-m-g2.1
GGGUCGUAGAUCAGCGGUAGAUCGCUUCCUUCGCAAGGAAGCCCUUGGGU
CAAUCCAGCGAGUCCACCA
FreeEnergyofEnsemble:-32.14kcal/mol
(((((((.....))))))(((((((.....)))))))).
(((((((.....))))))))).....

```

Acknowledgements

This work is supported in part by National Institutes of Health (R35GM145283 to D.H.M.)

References

- Petrov AI, Kay SJ, Kalvari I, Howe KL, Gray KA, Bruford EA, Kersey PJ, Cochrane G, Finn RD, Bateman A, Kozomara A, Griffiths-Jones S, Frankish A, Zwiab CW, Lau BY, Williams KP, Chan PP, Lowe TM, Cannone JJ, Gutell R, Machnicka MA, Bujnicki JM, Yoshihama M, Kenmochi N, Chai B, Cole JR, Szymanski M, Karlowski WM, Wood V, Huala E, Berardini TZ, Zhao Y, Chen R, Zhu W, Paraskevopoulou MD, Vlachos IS, Hatzigeorgiou AG, Ma L, Zhang Z, Puetz J, Stadler PF, McDonald D, Basu S, Fey P, Engel SR, Cherry JM, Volders PJ, Mestdagh P, Wower J, Clark MB, Quek XC, Dinger ME (2017) RNAcentral: a comprehensive database of non-coding RNA sequences. *Nucleic Acids Res* 45(D1):D128–D134
- Bachellerie JP, Cavaillé J, Hüttenhofer A (2002) The expanding snoRNA world. *Biochimie* 84(8):775–790
- Bellaousov S, Mathews DH (2010) ProbKnot: fast prediction of RNA secondary structure including pseudoknots. *RNA* 16(10):1870–1880
- Bernhart SH, Hofacker IL, Stadler PF (2006) Local RNA base pairing probabilities in large sequences. *Bioinformatics* 22(5):614–615
- Clote P, Ponty Y, Steyaert J (2012) Expected distance between terminal nucleotides of RNA secondary structures. *J Math Biol* 65(3):581–599
- Ding Y, Lawrence CE (2003) A statistical sampling algorithm for RNA secondary. *Nucleic Acids Res* 31(24):7280–7301
- Do C, Woods D, Batzoglou S (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* 22(14):e90–e98
- Doudna JA, Cech TR (2002) The chemical repertoire of natural ribozymes. *Nature* 418(6894):222–228
- Eddy SR (2001) Non-coding RNA genes and the modern RNA world. *Nat Rev Genet* 2(12):919–929
- Hofacker IL, Lorenz R (2014) Predicting RNA structure: advances and limitations. In: *RNA folding: methods and protocols*, pp 1–19

11. Huang L, Sagae K (2010) Dynamic programming for linear-time incremental parsing. In: Proceedings of ACL 2010. ACL, Uppsala, pp 1077–1086
12. Huang L, Zhang H, Deng D, Zhao K, Liu K, Hendrix DA, Mathews DH (2019) LinearFold: linear-time approximate RNA folding by 5'-to-3' dynamic programming and beam search. *Bioinformatics* 35(14):i295–i304
13. Kerpedjiev P, Hammer S, Hofacker IL (2015) Forna (force-directed RNA): simple and effective online RNA secondary structure diagrams. *Bioinformatics* 31(20):3377–3379
14. Kiryu H, Kin T, Asai K (2008) Rfold: an exact algorithm for computing local base pairing probabilities. *Bioinformatics* 24(3):367–373
15. Knudsen B, Hein J (2003) Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res* 31(13):3423–3428
16. Lai W-JC, Kayedkhordeh M, Cornell EV, Farah E, Bellaousov S, Rietmeijer R, Salsi E, Mathews DH, Ermolenko DN (2018) mRNAs and lncRNAs intrinsically form secondary structures with short end-to-end distances. *Nat Commun* 9(1):1–11
17. Lange SJ, Maticzka D, Mohl M, Gagnon JN, Brown CM, Backofen R (2012) Global or local? predicting secondary structure and accessibility in mRNAs. *Nucleic Acids Res* 40(12):5215–5226
18. Leija-Martínez N, Casas-Flores S, Cadena-Nava RD, Roca JA, Mendez-Cabañas JA, Gomez E, Ruiz-García J (2014) The separation between the 5'-3' ends in long RNA molecules is short and nearly constant. *Nucleic Acids Res* 42(22):13963–13968
19. Li TJ, Reidys CM (2018) The rainbow spectrum of RNA secondary structures. *Bull Math Biol* 80(6):1514–1538
20. Liu B, Mathews DH, Turner DH (2010) RNA pseudoknots: folding and finding. *F1000 Biology Reports* 2(8)
21. Lorenz R, Bernhart SH, Zu Siederdisen CH, Tafer H, Flamm C, Stadler PF, Hofacker IL (2011) ViennaRNA package 2.0. *Algorithms Mol Biol* 6(1):1
22. Lu ZJ, Gloor JW, Mathews DH (2009) Improved RNA secondary structure prediction by maximizing expected pair accuracy. *RNA* 15(10):1805–1813
23. Lyumkis D (2019) Challenges and opportunities in cryo-EM single-particle analysis. *J Biol Chem* 294(13):5181–5197
24. Mathews DH (2004) Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization. *RNA* 10(8):1178–1190
25. Mathews DH (2006) Revolutions in RNA secondary structure prediction. *J Mol Biol* 359(3):526–532
26. Mathews DH (2006) RNA secondary structure analysis using RNAstructure. *Curr Protoc Bioinform* 13(1):12–6
27. Mathews DH, Turner DH (2006) Prediction of RNA secondary structure by free energy minimization. *Curr Opin Struct Biol* 16(3):270–278
28. McCaskill JS (1990) The equilibrium partition function and base pair probabilities for RNA secondary structure. *Biopolymers* 29:11105–1119
29. Nussinov R, Jacobson AB (1980) Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc Nat Acad Sci USA* 77(11):6309–6313
30. Reuter JS, Mathews DH (2010) RNAstructure: software for RNA secondary structure prediction and analysis. *BMC Bioinform* 11(1):1–9
31. Sato K, Kato Y, Hamada M, Akutsu T, Asai K (2011) IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming. *Bioinformatics* 27(13):i85–i93
32. Sektin MG, Mathews DH (2012) RNA structure prediction: an overview of methods. In: *Bacterial regulatory RNA: methods and protocols*, pp 99–122
33. Sloma M, Mathews D (2016) Exact calculation of loop formation probability identifies folding motifs in RNA secondary structures. *RNA* 22:1808–1818
34. Sperschneider J, Datta A (2010) DotKnot: pseudoknot prediction using the probability dot plot under a refined energy model. *Nucleic Acids Res* 38(7):e103–e114
35. Turner DH, Mathews DH (2009) NNDB: The nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res* 38:D280–D282
36. Waterman MS, Smith TF (1986) Rapid dynamic programming algorithms for RNA secondary structure. *Adv Appl Math* 7(4):455–464
37. Yoffe AM, Prinsen P, Gelbart WM, Ben-Shaul A (2011) The ends of a large RNA molecule are necessarily close. *Nucleic Acids Res* 39(1):292–299
38. Zhang H, Keane S (2019) Advances that facilitate the study of large RNA structure and dynamics by nuclear magnetic resonance spectroscopy. *Wiley Interdiscip Rev RNA* 10:e1541

39. Zhang H, Zhang L, Mathews DH, Huang L (2020) LinearPartition: linear-time approximation of RNA folding partition function and base-pairing probabilities. *Bioinformatics* 36: i258–i267
40. Zhang J, Ferré-D’Amaré AR (2014) New molecular engineering approaches for crystallographic studies of large RNAs. *Curr Opin Struct Biol* 26:9–15
41. Zhang L, Zhang H, Mathews DH, Huang L (2019) ThreshKnot: thresholded probknot for improved RNA secondary structure prediction. bioRxiv
42. Zuker M (1989) On finding all suboptimal foldings of an RNA molecule. *Science* 244(4900):48–52
43. Zuker M, Stiegler P (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res* 9(1):133–148



Genome-Wide RNA Secondary Structure Prediction

Risa Karakida Kawaguchi and Hisanori Kiryu

Abstract

The information of RNA secondary structure has been widely applied to the inference of RNA function. However, a classical prediction method is not feasible to long RNAs such as mRNA due to the problems of computational time and numerical errors. To overcome those problems, sliding window methods have been applied while their results are not directly comparable to global RNA structure prediction. In this chapter, we introduce ParasoR, a method designed for parallel computation of genome-wide RNA secondary structures. To enable genome-wide prediction, ParasoR distributes dynamic programming (DP) matrices required for structure prediction to multiple computational nodes. Using the database of not the original DP variable but the ratio of variables, ParasoR can locally compute the structure scores such as stem probability or accessibility on demand. A comprehensive analysis of local secondary structures by ParasoR is expected to be a promising way to detect the statistical constraints on long RNAs.

Key words RNA secondary structure, Local structure, Genome-wide, Parallel computation, Maximal span constraint

1 Introduction

The prediction of RNA secondary structures has played an important role in the discovery of functional RNAs, particularly short noncoding RNAs (ncRNAs). Previous studies have revealed that RNA secondary structure is associated with the stability and interactions of short RNAs [1, 2]. In a recent application, researchers designed efficient guide RNAs for CRISPR systems according to the structural stability of individual and interacting nucleotide molecules [1]. Structural prediction for long RNAs, such as mRNAs or pre-mRNAs, also has great potential for inferring how coding RNAs are properly regulated; thus, it has relevance to understanding mechanisms such as protein expression and splicing efficiency [3, 4].

Regardless of its great utility, genome-wide prediction of RNA secondary structures is computationally demanding, requiring substantial computing time and memory. For example, the computational time for algorithms such as Mfold [5] or McCaskill [6],

which utilize a thermodynamic energy mode based on dynamic programming (DP), is $O(N^3)$ to predict RNA secondary structures and base pairing probabilities for a sequence of length N . This is the case where all pseudoknot-free base pairs are considered without approximation. A general approach to genome-wide RNA structure analysis that can overcome these computational resource limitations involves performing local RNA structure predictions for an entire sequence. In support of this approach, Mahen et al. [7] reported that mRNAs tend to form proximal base pairs more frequently than distant base pairs in vivo [7], which suggests that the efficiency and precision of RNA structure prediction may be less affected by ignoring distant base pairs. Moreover, a base pair distance of around 150 nucleotides was found to be most suitable in a previous analysis of structural regions [8]. Thus, it can be concluded that setting a constraint on the maximal span is a reasonable approach to reducing computational complexity from $O(N^3)$ to $O(NW^2)$, where W is the maximal span. Moreover, the accuracy of the energy parameters used for prediction is potentially reduced as the distance of distant base pairs increases. For example, the most widely used energy models for RNA structure prediction are constructed according to the melting temperature measured for short oligonucleotide RNAs [9]. Other variants of energy models constructed via machine learning methods, such as the Boltzmann Likelihood model [10] and ContraFold [11], have been trained using data from existing structure databases, which also largely consist of short RNAs. Although distant base pairs are equally or more important than proximal base pairs when determining whole structures, the strategy of predicting the local density of RNA structures, i.e., local accessibility, has been widely studied for the reasons outlined above.

In local RNA structure analysis, three major strategies are available for practical prediction of structures: (1) computing structures for whole long RNAs while applying a constraint for possible secondary structures, e.g., the maximal span of base pairs; (2) predicting structure features based on the results of each independent window analysis; and (3) estimating regional structure characteristics from sequential features. The first strategy advantageously produces results for local RNA structures that can be directly compared with global RNA structures; however, the second and third strategies can be applied to a variety of targets without the huge computational time constraints by reducing the demand of computational resources more efficiently.

In this chapter, we introduce Parasor, a parallel computation platform for local RNA secondary structure that utilizes the first strategy mentioned above to analyze a genome-wide structural tendency. Parasor can compute a variety of structural features, such as base pairing probability or accessibility over the globally consistent structures. With the maximal span constraint W to limit

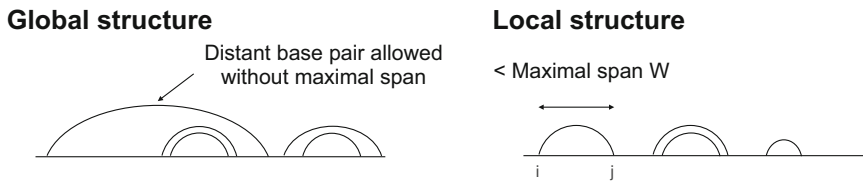


Fig. 1 Example of local and global RNA secondary structures without pseudoknots

the base pair distance, ParasoR predicts a local structure rather than global structure (*see* Fig. 1) at a genome-wide level with the computational complexity $O(NW^2)$. This computation is performed by multiple computational nodes in parallel with utilizing the ratios of DP variables to avoid over- and underflow problems so that the results of ParasoR can be obtained in a consistent manner for classical secondary structure prediction for short RNAs regardless of the sequence length.

2 Methods

ParasoR is a method by which to conduct parallel computation for genome-scale prediction of RNA secondary structure and other structural features. The predictions of ParasoR are based on the Boltzmann ensemble for globally consistent secondary structures. It can be applied as an integrative platform for computation of different local RNA secondary structure predictions, integrating algorithms such as Rfold [12], Raccess [13], and CapR [14]. To achieve genome-scale RNA secondary structure prediction, ParasoR has two different modes: single- and multi-node computation. In single-node mode, the functions of ParasoR include the prediction of representative RNA secondary structures such as minimum free-energy (MFE) structure [5], maximal expected accuracy (MEA) structure [11], and centroid structure [15]. On the other hand, multi-node mode is optimized to handle RNA secondary structure predictions of longer RNA sequences, such as mRNA and pre-mRNA; thus, only part of the MEA structure can be predicted for each region.

The key features underlying ParasoR in the multi-node mode included (1) distributing the computation of DP matrices and (2) computing the ratio of DP matrices rather than original DP variables. This parallel option enables computation of the same structure scores as those computed in single-node mode but also incorporates multiple computer clusters. Switching between the two modes is achieved using the option “previous method” (Table 1). The settings for RNA secondary structure prediction, such as maximal span, energy model, or output file prefix, can also be edited via the parameters shown in Table 1. The descriptions of

Table 1
Primary parameters used in ParasoR to control the basic settings

Option name	Command	Argument	Description
Previous method	--pre	NA	Enable a single-core computation
Maximal constraint	--constraint	int	Set the limit for a base pair distance
Output file name	--name	string	Prefix for output files
Start and end	-s and -e	long int	Region to be analyzed
Energy model	--energy	string	Choice of energy models

other available parameters can be found by running ParasoR with a help option.

In the following subsections, we present a variety of examples of local RNA structure analyses. First, we detail how to use ParasoR in single-node mode and then provide a simple case for its use in multi-node computation mode.

2.1 RNA Secondary Structure Analysis in Single-Node Mode

ParasoR can handle several types of structure prediction, whereas only MEA structures are available in parallel computation mode. An MFE structure is that for which the free-energy change is estimated as the minimum; therefore, the probability of an MFE structure is highest among all possible structures. However, in reality, the probability of an MFE structure is known to be extremely low because RNA can form many possible structures via many combinations of base pairs [16]. To reflect the stability distribution of the whole structure space instead of the best structure with a low possibility, MEA and γ -centroid structures were developed to evaluate not only the MFE structure but also other structures at the level of each base pair. An MEA structure is that which maximizes the MEA estimator to increase the structure prediction accuracy. The γ -centroid estimator used for centroid structure prediction is another estimator that further improves prediction accuracy based on the principles of the MEA estimator. Both of these estimators achieve higher prediction accuracies than MFE structures in available benchmark datasets [17]. These various structure predictions are implemented in ParasoR because comparisons of MFE and other structure predictions are informative when estimating the robustness of RNA structure prediction for a given sequence. Examples that demonstrate MFE and centroid structure prediction via ParasoR are provided with explanations below.

Example 1: ParasoR --pre -f [sequence] --mfe --struct

Example 2: ParasoR --pre -f [sequence] --struct=[gamma]

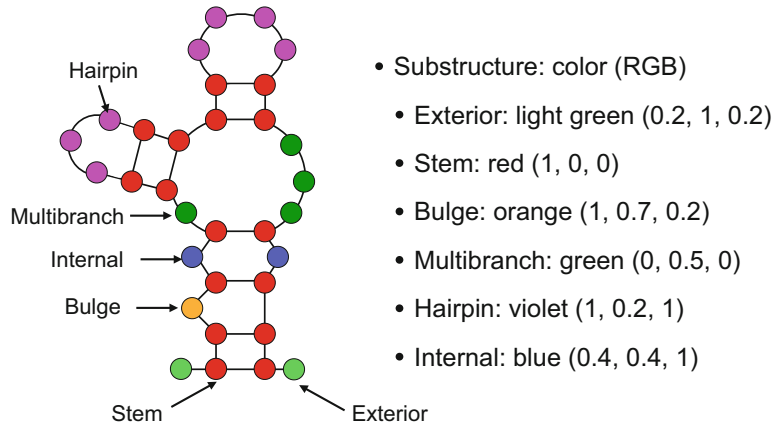


Fig. 2 Example of RNA secondary structure visualization and six substructure classes

Example 1: MFE structure prediction. An MFE structure is the most stable structure for which the estimated free energy change is the minimum among all possible structures. This prediction is only available in single-node mode because of its computational time ($O(NW^2)$).

Example 2: A γ -centroid structure. A γ -centroid structure prediction is also only available in single-node mode because its DP computation procedure has the same computational complexity as in the MFE structure prediction. The option for centroid structure prediction allows a floating point number to modulate the threshold of probability for permissible base pairs.

Example 2': MEA structure. In multi-node mode for genome-wide structure prediction, only a partial MEA structure can be predicted using base pairing probabilities of the target region using a “struct” option. The predicted structures do not contain any base pairs from outside of the region. Additionally, a full MEA structure is obtained by running ParasoR in multi-node mode using a single node although this computation would be impractical for long RNAs.

Rather than predicting single secondary structures, local structure features can also be widely applied to analyze long RNAs. RNA secondary structures consist of six possible substructures: bulge, exterior hairpin, multibranch, stem, and internal loops (Fig. 2). The basic idea of local structure evaluation is computing the probability of forming a specific substructure being at each base or contiguous region level. In ParasoR, five structure scores are computed in single- and multicore mode. Examples that demonstrate local structure analyses are provided below.

Example 1: `ParasoR --pre -f [sequence]`

Example 1': `ParasoR --pre --input [fasta file]`

Example 2: `ParasoR --pre -f [sequence] --stem`

Example 2': `ParasoR --pre -f [sequence] --struct --image`

Example 3: `ParasoR --pre -f [sequence] --acc --window [int]`

Example 4: `ParasoR --pre -f [sequence] --prof`

Example 4': `ParasoR --pre -f [sequence] -prof --struct --image`

Example 5: `ParasoR --pre -f [sequence] --motif`

Example 1: Base pairing probability. $p(i, j)$ is the summation of the probabilities of the structures in which the i th and j th bases are paired each other, where $1 \leq i < j \leq N$. In ParasoR, the maximal constraint, which is represented by W , is set to satisfy the condition $j - i \leq W$. The default output of ParasoR is set to base pairing probability.

Example 2: Stem probability. $p(i)$ is the summation of the probabilities of the structures for which the i th base is paired with any other base. Using “image” and “struct” options, ParasoR can output an image of the predicted structure whose color of each base corresponds to the stem probability.

Example 3: Accessibility. $p_{acc}(i, j)$ is the summation of the probabilities for the structure in which the entire region from the i th to j th base is unpaired and belongs to any of substructures except for stem. The parameter w controls the distance between i and j . Further detail on accessibility is provided by Kiryu et al. [13].

Example 4: Structure profiles. $p_s(i)$ is the summation of the probabilities for the structure in which the i th base involves substructure s from six possible substructures. Further detail on structure profiles is provided by Fukunaga et al. [14]. ParasoR can additionally produce an image of the pie charts of structure profiles for each base on the predicted structure.

Example 5: Structure motifs. This option changes the output of structure profiles into a motif sequence of length N . For each base position, one of the substructures for which the probability, or structure profile $p_s(i)$, is highest is selected. The first letters of the selected substructure for each base are concatenated as a structure motif sequence, which is applicable to a simple enrichment analysis of structure motifs using a general motif search application, e.g., HOMER [18]. This option could feasibly be used for the analysis of datasets from high-throughput motif enrichment experiments, such as the methods to detect the binding sites of RBPs and ncRNAs.

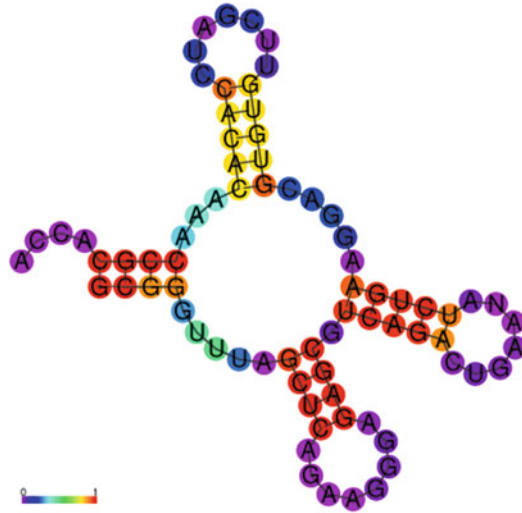


Fig. 3 Predicted γ -centroid structure of phe-tRNA. Each circle in the predicted structure shows the stem probability of each base

2.2 Transfer RNA Secondary Structure Prediction by ParasoR: An Example

Here, an example output of a single-node ParasoR computation for the phe-tRNA sequence from *Neurospora crassa* [19] is provided. Figures 3 and 4 represent a predicted centroid structure with different structural profile annotations. In Fig. 3, the color of each base represents a stem probability; it can be applied to check the consistency between the predicted structure and stem probability. On the other hand, in Fig. 4, each base includes a pie chart in which the probability of each substructure (structure profiles) is shown by slices of different colors; it may be useful for checking the consistency of a predicted structure and its reliability in a single-base resolution.

Example:

```
ParasoR --pre -f GCGGGUUUAGCUCAGAAGGGAGAGCGUCAGACUGAAYAUCUGAAG
GACGUGUGTUCGAUCCACACAAACCGCACCA --image --struct
```

Example outputs:

```
#-Check working directory: /Users/cawa/Software/ParasoR/
#-Temperature 37 616.321
#-Read Energy -> /Users/cawa/Software/ParasoR/energy_param/
rna_turner2004.par
# Not parallel, Rfold computing
# GCGGGUUUAGCUCAGAAGGGAGAGCGUCAGACUGAANAUCUGAAGGACGU...
#--log(Z): 33.02887936
#--hard-const-disabled
#-Centroid structure (gamma = 1)
#structure ((((((.....((((.....))))).((((.....)))))).....
((((.....))))))....
```

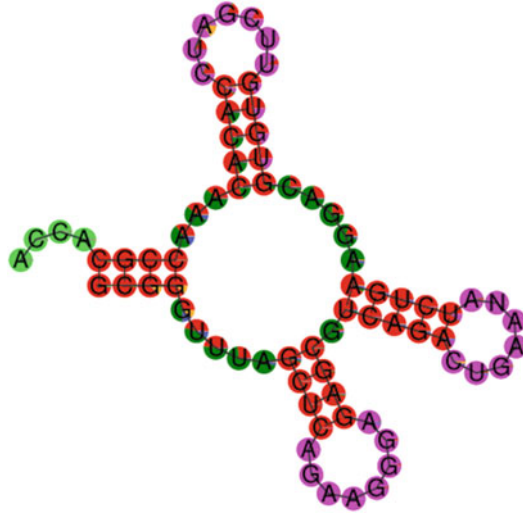


Fig. 4 Predicted centroid structure of phe-tRNA. Each circle in the predicted structure corresponds to a pie chart in which each slice represents the probability of each substructure, i.e., the structure profile. Specifically, the probabilities of exterior, stem, bulge, multibranch, hairpin, and internal loop are shown in light green, red, orange, green, violet, and blue, respectively

```
#-Drawing to /Users/cawa/Software/ParasoR/prob/_stem_0_-
se=0_75_gamma=1.ps
# local seq: GCGGUUUAGCUCAGAAGGGAGCGUCAGACUGAANAUCUGAAGGAC
GUGUGUUCGAUCCACACAAACCGCACCA
# local str: ((((((.....((((.....)))))).((((.....)))))).....
((((.....)))))).....
```

2.3 Rfold Algorithm

The Rfold model and its derivative models, such as Raccess and ParasoR, are algorithms for analyses of local RNA secondary structures. These models are the variants of stochastic context-free grammars and applicable to existing thermodynamic folding models. An important feature of Rfold involves setting a maximal span of base pairs to implement feasible structural analysis of long RNAs.

In its grammar, Rfold includes seven nonterminal symbols: Outer, Stem, StemEnd, Multi, MultiBif, Multi1, and Multi2. Each state emits a corresponding structure, which has been detailed by Kiryu, Kin, and Asai [12] and by Kawaguchi and Kiryu [20]. To calculate a structure probability such as the base pairing probability, Rfold calculates the sum of free energy changes for target structures based on a thermodynamic folding model. This process is recursively computed based on an inside outside algorithm for each state and subsequence. For state s , inside variables $\alpha_s(k, i)$ correspond to all structures being s at the edge of a partial sequence from the k th to i th bases. Outside variables $\beta_s(k, i)$ represent all structures of the

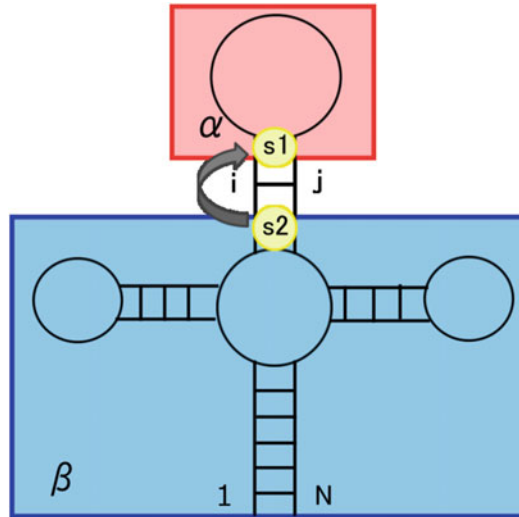


Fig. 5 Rfold algorithm. Inside and outside variables and corresponding regions used to calculate base pairing probability. The inside Stem variable stores the sum of free-energy changes of structures with a particular base pair for a given region (indicated in red). The outside variables store the same for a region outside of the specified indices (represented by a blue rectangle). By multiplying the inside, outside, and transition energy variables for the combination of substructure states that can produce a base pair between them, Rfold computes a base pairing probability following a specific thermodynamic folding model for an estimation of free energy change

sequence from the first to k th bases and from the i th to last bases that involve s at the internal edge of the two sequences (as shown in Fig. 5).

As an example, the calculation of inside Outer variables (represented by α_{Outer}), or the sum of the free-energy changes of all structures for a partial sequence, is provided. $\alpha_{Outer}(i)$ is obtained from inside variables with an index $\leq i$ and transition energy as follows (also see Fig. 6):

$$\alpha_{Outer}(i) = \alpha_{Outer}(i-1) + \sum_{k=i-W-1}^{i-1} \alpha_{Outer}(k) \tilde{n} \alpha_{Stem}(k, i) \tilde{n} t - (Outer \rightarrow Outer \tilde{n} Stem),$$

where W represents the maximal span and $t(Outer \rightarrow Outer \cdot Stem)$ represents the transition energy between *Outer* and *Outer · Stem*. $\alpha_{Stem}(k, i)$ corresponds to the sum of the free energy changes for the structures of the partial sequence that have a base pair between the k th and i th base, which can be computed independently from α_{Outer} . This inside Outer variable is equal to the partition function, which is required for probability computing, when the i th base is set to the end of the sequence. As such, Rfold can efficiently compute a probability considering all possible substructures following such recursive equations.

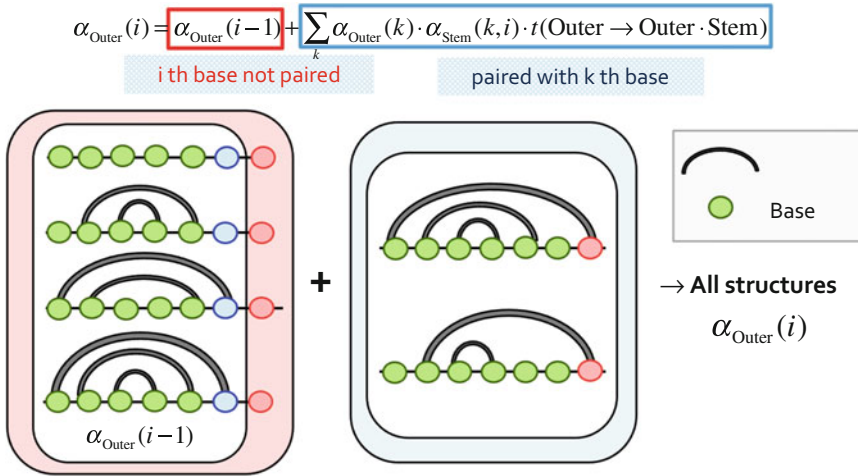


Fig. 6 A schematic diagram showing the recursive computation of α_{Outer}

2.4 ParasoR Algorithm

While the Rfold model enables the efficient computation of structure characteristics with the constraint of maximal span, it cannot be applied to genome-wide RNA secondary structure prediction because of increased computational time and over- and underflow problems. To overcome these problems, ParasoR was developed as an extension of Rfold with optimization for multi-node computation of long RNA sequences (Fig. 7). In the ParasoR algorithm, an RNA sequence is first fragmented into chunks, and then each chunk is distributed to multiple nodes for parallel computation. Next, ParasoR connects the temporary databases constructed by each node and predicts local structures and probabilities over globally consistent secondary structures. Examples of parallel computation by ParasoR where two computational nodes are available are provided below.

Examples:

```
ParasoR -i 0 -k 2 # run using the first node.
```

```
ParasoR -i 1 -k 2 # run using the second node.
```

```
ParasoR -i 2 -k 2 # connects the ratios of inside and outside variables.
```

After the connection step, two files are stored in an “outer” directory, which includes the variables associated with inside- and outside-outer variables. Using this database, ParasoR computes an MEA structure and structural features efficiently. Examples of local structure analyses are provided below.

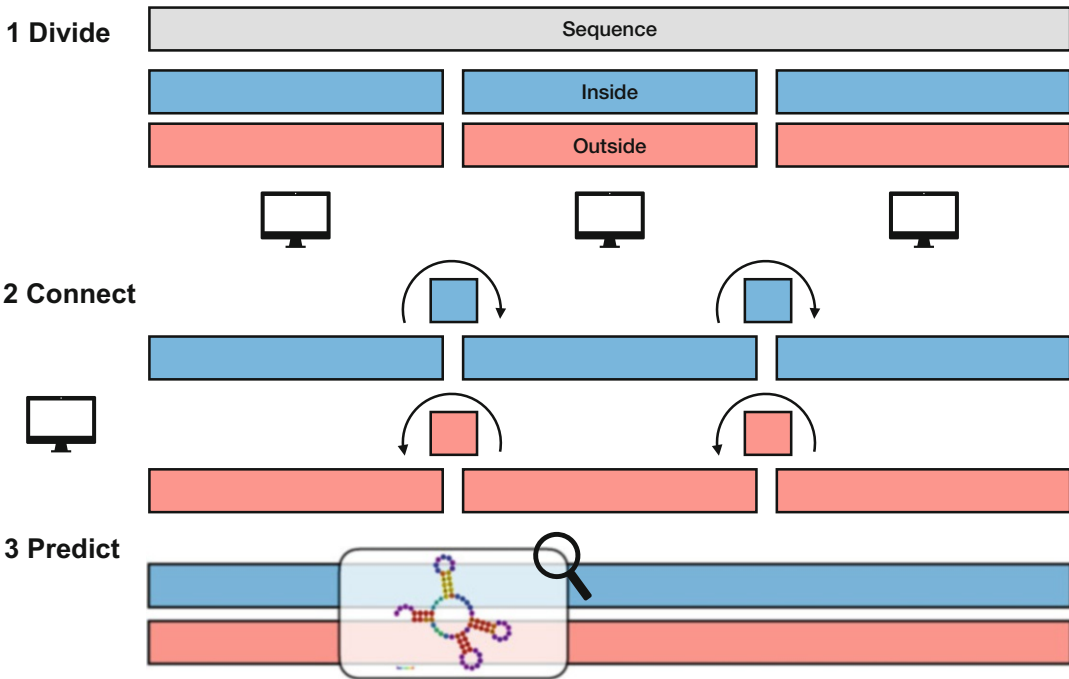


Fig. 7 ParasoR algorithm. In the multi-node mode of ParasoR, a genome-wide local structure prediction is achieved by following three procedures: *divide*, *connect*, and *predict*. After computation of the temporary databases for each region, all databases are concatenated using a single node to construct the final database of globally consistent inside and outside variables. Using the resultant database, ParasoR computes a local RNA secondary structure from any point in the sequence

Examples:

```
ParasoR -s [start] -e [end] --stem.
```

```
ParasoR -s [start] -e [end] --struct # MEA structure prediction.
```

```
ParasoR -i 0 -k 2 --stemdb --stdout # computes the stem probability for the first half of a sequence.
```

3 Notes

For the analysis of local secondary structures, the most critical difference between ParasoR and other models, such as RNAplfold [21] and Localfold [8], is that ParasoR considers a globally consistent structure for structure evaluation. While other methods integrate predictions of secondary structures for partial sequences toward a local structure analysis, ParasoR predictions can be used like classical structure predictions without the constraint of a maximal span, such as in Mfold or RNAfold [22]. While local structure prediction over globally consistent structures is currently available

in RNAfold with the “maxBPspan” option, the further advantage of ParasoR is its applicability to genome-scale long sequences via the application of DP ratio computation by multiple computational nodes.

While ParasoR can predict a secondary structure even for genome-scale long RNAs, it was originally designed for mRNAs and pre-mRNAs, assuming that one sequence is transcribed into a single molecule and forms a single RNA secondary structure. Therefore, other strategies may be better options when the aim is discovery of functional regions containing small ncRNAs transcribed individually from genome sequence through a local structure analysis. A combination of RNA folding and sequence alignments is an approach applied in previous studies to identify functional ncRNAs, e.g., with Evofold [23], LocARNA-P [24], and RNAscClust [25]. Other alignment-free approaches, such as RNAz [26] or ScanFold [27], utilize a control distribution of free energy changes obtained by machine learning and shuffled sequences, respectively. As an alternative option, resources and databases of conserved structural elements and predicted RNA secondary structures, which have been created via genome-wide scanning methods employing abundant computational resources, are also available [28–30]. A database containing the results of high-throughput structure probing experiments may also help elucidate the structural elements of the genome under *in vitro* and *in vivo* conditions [31–34]. Recent studies have revealed that *in vivo* RNA modification can change structures substantially from those predicted *in silico* [35, 36]. Therefore, integrative use of *ab initio* local structure analysis and other such resources could potentially reveal new functional aspects of long RNAs.

References

1. Kocak DD, Josephs EA, Bhandarkar V, Adkar SS, Kwon JB, Gersbach CA (2019) Increasing the specificity of CRISPR systems with engineered RNA secondary structures. *Nat Biotechnol* 37(6):657–666. <https://doi.org/10.1038/s41587-019-0095-1>
2. Mann M, Patrick RW, Backofen R (2017) IntaRNA 2.0: enhanced and customizable prediction of RNA-RNA interactions. *Nucleic Acids Res* 45(W1):W435–W439. <https://doi.org/10.1093/nar/gkx279>
3. Mauger DM, Cabral BJ, Presnyak V, Su SV, Reid DW, Goodman B, Link K, Khatwani N, Reynders J, Moore MJ, McFadyen IJ (2019) mRNA structure regulates protein expression through changes in functional half-life. *Proc Natl Acad Sci U S A* 116(48):24075–24083. <https://doi.org/10.1073/pnas.1908052116>
4. Ding Y, Tang Y, Kwok CK, Zhang Y, Bevilacqua PC, Assmann SM (2014) *In vivo* genome-wide profiling of RNA secondary structure reveals novel regulatory features. *Nature* 505(7485):696–700. <https://doi.org/10.1038/nature12756>
5. Michael Z, Stiegler P (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res.* <https://doi.org/10.1093/nar/9.1.133>
6. McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers* 29(6-7):1105–1119. <https://doi.org/10.1002/bip.360290621>
7. Elisabeth MM, Watson PY, Cottrell JW, Fedor MJ (2010) mRNA secondary structures fold sequentially but exchange rapidly *in vivo*.

- PLoS Biol 8(2):e1000307. <https://doi.org/10.1371/journal.pbio.1000307>
8. Lange SJ, Maticzka D, Möhl M, Gagnon JN, Brown CM, Backofen R (2012) Global or local? Predicting secondary structure and accessibility in mRNAs. *Nucleic Acids Res* 40(12):5215–5226. <https://doi.org/10.1093/nar/gks181>
 9. Turner DH, Mathews DH (2010) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res* 38(Database issue):D280–D282. <https://doi.org/10.1093/nar/gkp892>
 10. Andronescu M, Condon A, Hoos HH, Mathews DH, Murphy KP (2010) Computational approaches for RNA energy parameter estimation. *RNA* 16(12):2304–2318. <https://doi.org/10.1261/rna.1950510>
 11. Do CB, Woods DA, Batzoglou S (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* 22(14):e90–e98. <https://doi.org/10.1093/bioinformatics/btl246>
 12. Kiryu H, Kin T, Asai K (2008) Rfold: an exact algorithm for computing local base pairing probabilities. *Bioinformatics* 24(3):367–373. <https://doi.org/10.1093/bioinformatics/btm591>
 13. Kiryu H, Terai G, Imamura O, Yoneyama H, Suzuki K, Asai K (2011) A detailed investigation of accessibilities around target sites of siRNAs and miRNAs. *Bioinformatics* 27(13):1788–1797. <https://doi.org/10.1093/bioinformatics/btr276>
 14. Fukunaga T, Ozaki H, Terai G, Asai K, Iwasaki W, Kiryu H (2014) CapR: revealing structural specificities of RNA-binding protein target recognition using CLIP-seq data. *Genome Biol* 15(1):R16. <https://doi.org/10.1186/gb-2014-15-1-r16>
 15. Hamada M, Kiryu H, Sato K, Mituyama T, Asai K (2009) Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics* 25(4):465–473. <https://doi.org/10.1093/bioinformatics/btn601>
 16. Schroeder SJ (2009) Advances in RNA structure prediction from sequence: new tools for generating hypotheses about viral RNA structure-function relationships. *J Virol* 83(13):6326–6334. <https://doi.org/10.1128/jvi.00251-09>
 17. Puton T, Kozłowski LP, Rother KM (2014) CompaRNA: a server for continuous benchmarking of automated methods for RNA secondary structure prediction. *Nucleic Acids Res* 42(8):5403–5406. <https://doi.org/10.1093/nar/gkt101>
 18. Heinz S, Benner C, Spann N, Bertolino E, Lin YC, Laslo P, Cheng JX, Murre C, Singh H, Glass CK (2010) Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Mol Cell*. <https://doi.org/10.1016/j.molcel.2010.05.004>
 19. Alzner-DeWeerd B, Hecker LI, Barnett WE (1980) The nucleotide sequence of phenylalanine tRNA from the cytoplasm of *Neurospora Crassa*. *Nucleic Acids Res*. <https://doi.org/10.1093/nar/8.5.1023>
 20. Kawaguchi R, Kiryu H (2016) Parallel computation of genome-scale RNA secondary structure to detect structural constraints on human genome. *BMC Bioinformatics* 17(1):203. <https://doi.org/10.1186/s12859-016-1067-9>
 21. Bernhart SH, Hofacker IL, Stadler PF (2006) Local RNA base pairing probabilities in large sequences. *Bioinformatics* 22(5):614–615. <https://doi.org/10.1093/bioinformatics/btk014>
 22. Lorenz R, Bernhart SH, Höner Zu Siederdisen C, Tafer H, Flamm C, Stadler PF, Hofacker IL (2011) ViennaRNA package 2.0. *Algorithms Mol Biol* 6:26. <https://doi.org/10.1186/1748-7188-6-26>
 23. Pedersen JS, Bejerano G, Siepel A, Rosenbloom K, Lindblad-Toh K, Lander ES, Kent J, Miller W, Haussler D (2005) Identification and classification of conserved RNA secondary structures in the human genome. *PLoS Comput Biol*. <https://doi.org/10.1371/journal.pcbi.0020033>
 24. Will S, Joshi T, Hofacker IL, Stadler PF, Backofen R (2012) LocARNA-P: accurate boundary prediction and improved detection of structural RNAs. *RNA* 18(5):900–914. <https://doi.org/10.1261/rna.029041.111>
 25. Miladi M, Junge A, Costa F, Seemann SE, Havgaard JH, Gorodkin J, Backofen R (2017) RNAscClust: clustering RNA sequences using structure conservation and graph based Motifs. *Bioinformatics* 33(14):2089–2096. <https://doi.org/10.1093/bioinformatics/btx114>
 26. Gruber AR, Findeiß S, Washietl S, Hofacker IL, Stadler PF (2009) RNAz 2.0: improved noncoding RNA detection. *Biocomputing 2010*. https://doi.org/10.1142/9789814295291_0009
 27. Andrews RJ, Roche J, Moss WN (2018) ScanFold: an approach for genome-wide discovery of local RNA structural elements-applications

- to Zika virus and HIV. *PeerJ* 6:e6136. <https://doi.org/10.7717/peerj.6136>
28. Andrews RJ, Baber L, Moss WN (2017) RNAstruromeDB: a genome-wide database for RNA structural inference. *Sci Rep* 7(1):17269. <https://doi.org/10.1038/s41598-017-17510-y>
 29. Thiel BC, Ochsenreiter R, Gadekar VP, Tanzer A, Hofacker IL (2018) RNA structure elements conserved between mouse and 59 other vertebrates. *Genes* 9(8):392. <https://doi.org/10.3390/genes9080392>
 30. Danaee P, Rouches M, Wiley M, Dang D, Huang L, Hendrix D (2018) bpRNA: large-scale automated annotation and analysis of RNA secondary structure. *Nucleic Acids Res* 46(11):5381–5394. <https://doi.org/10.1093/nar/gky285>
 31. Berkowitz ND, Silverman IM, Childress DM, Kazan H, Wang L, Gregory BD (2016) A comprehensive database of high-throughput sequencing-based RNA secondary structure probing data (Structure Surfer). *BMC Bioinformatics* 17(1):215. <https://doi.org/10.1186/s12859-016-1071-0>
 32. Yesselman JD, Tian S, Liu X, Shi L, Li JB, Das R (2018) Updates to the RNA mapping Database (RMDB), version 2. *Nucleic Acids Res* 46(D1):D375–D379. <https://doi.org/10.1093/nar/gkx873>
 33. Wirecki TK, Merdas K, Bernat A, Boniecki MJ, Bujnicki JM, Stefaniak F (2020) RNAProbe: a web server for normalization and analysis of RNA structure probing data. *Nucleic Acids Res* 48(W1):W292–W299. <https://doi.org/10.1093/nar/gkaa396>
 34. Norris M, Kwok CK, Cheema J, Hartley M, Morris RJ, Aviran S, Ding Y (2017) FoldAtlas: a repository for genome-wide RNA structure probing data. *Bioinformatics* 33(2):306–308. <https://doi.org/10.1093/bioinformatics/btw611>
 35. Zubradt M, Gupta P, Persad S, Lambowitz AM, Weissman JS, Rouskin S (2017) DMS-MaPseq for genome-wide or targeted RNA structure probing in vivo. *Nat Methods* 14(1):75–82. <https://doi.org/10.1038/nmeth.4057>
 36. Liu B, Merriman DK, Choi SH, Schmacher MA, Plangger R, Kreutz C, Horner SM, Meyer KD, Al-Hashimi HM (2018) A potentially abundant junctional RNA motif stabilized by m⁶A and Mg²⁺. *Nat Commun* 9(1):2761. <https://doi.org/10.1038/s41467-018-05243-z>



Nucleic Acid Structure Prediction Including Pseudoknots Through Direct Enumeration of States: A User's Guide to the LandscapeFold Algorithm

Ofer Kimchi, Michael P. Brenner, and Lucy J. Colwell

Abstract

Here we detail the LandscapeFold secondary structure prediction algorithm and how it is used. The algorithm was previously described and tested in (Kimchi O et al., *Biophys J* 117(3):520–532, 2019), though it was not named there. The algorithm directly enumerates all possible secondary structures into which up to two RNA or single-stranded DNA sequences can fold. It uses a polymer physics model to estimate the configurational entropy of structures including complex pseudoknots. We detail each of these steps and ways in which the user can adjust the algorithm as desired. The code is available on the GitHub repository <https://github.com/ofcer-kimchi/LandscapeFold>.

Key words Pseudoknot, Structure enumeration, Minimum free energy structure, Free energy landscape, Polymer physics theory

1 Introduction

Short RNA molecules are ubiquitous in modern biology. In vivo, small non-coding RNA molecules are present at high copy numbers in a wide variety of both eukaryotic and prokaryotic cells [1, 2], have been implicated in nearly all aspects of biological regulation [3], and have been found to interact with DNA, mRNA, other non-coding RNA, and proteins [4, 5]. In vitro, the laboratory evolution of RNA, especially through SELEX [6–8], has led to an explosion of applications for short RNA and single-stranded DNA molecules, due to their ability to tightly and specifically bind to a remarkable range of target ligands [9].

Where they are known, the functions and interaction partners of many RNA molecules are determined by their minimum free energy structures and by their structure landscapes [10–15]. RNA structures, while fully three-dimensional in nature, can in many cases be productively defined by a list of the base pairs in the

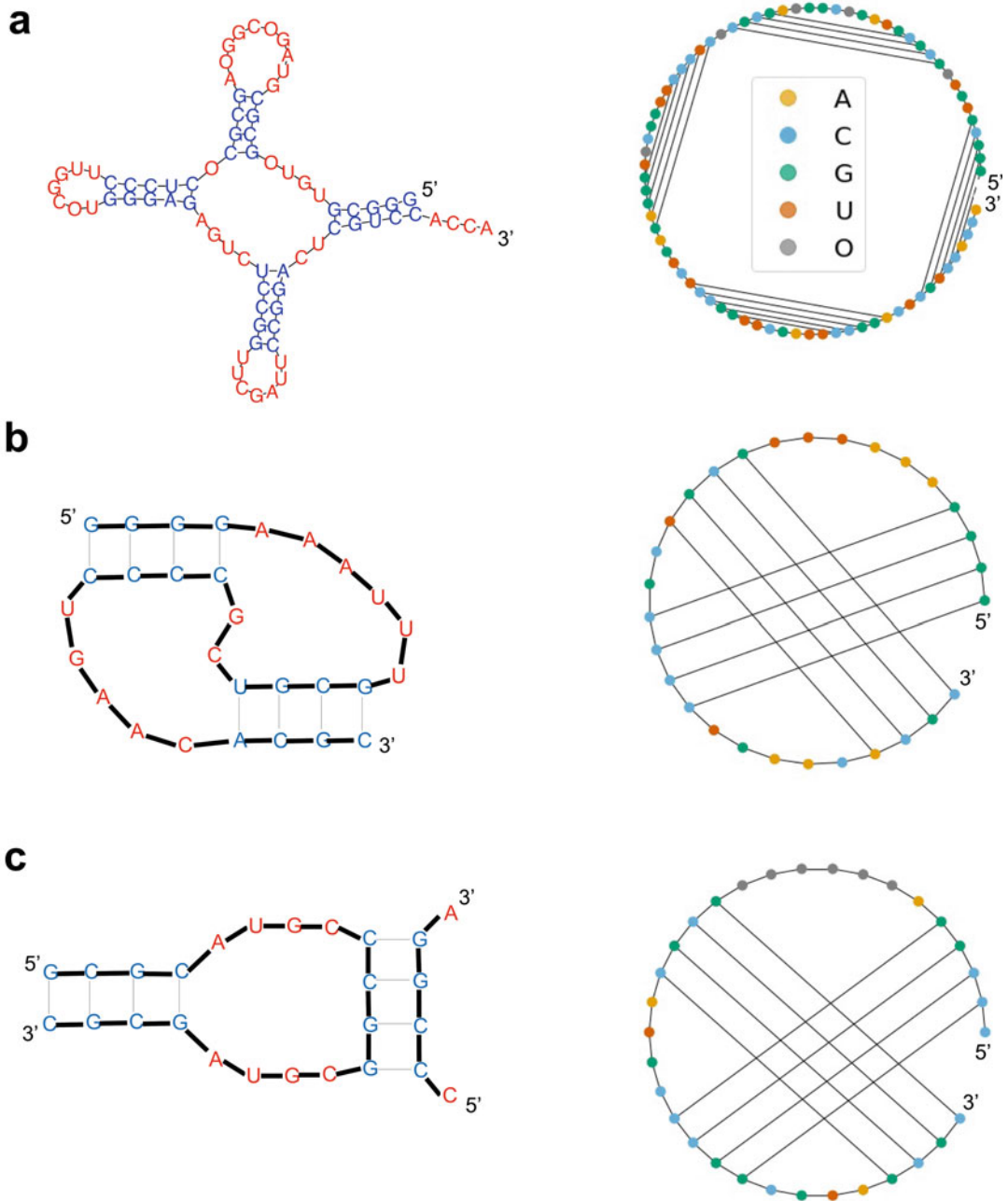


Fig. 1 RNA structures and pseudoknots. Three RNA secondary structures are depicted, each in two forms: as a planar graph (left) where paired nucleotides are nearby, and as a circular diagram (right) where paired nucleotides are connected by arcs. The planar graphs are color-coded by whether or not the nucleotide is paired; the circular diagrams by nucleotide sequence. **(a)** Non-pseudoknotted structure. An example of a non-pseudoknotted structure. O's represent unknown nucleotides and are unpaired. The specific structure shown is motivated by Ref. [17]. The circular diagrams of structures without pseudoknots do not contain any intersections in the arcs connecting paired nucleotides. **(b)** A simple pseudoknot. A simple intramolecular pseudoknot is depicted. Pseudoknots are defined as non-nested loops, and are easy to visualize in circular diagrams as intersections in the arcs connecting paired nucleotides. **(c)** An intermolecular pseudoknot. A

structure, termed the secondary structure (Fig. 1a). The minimum free energy structures of short RNA molecules (without non-nested loops) can be predicted with accuracies of $\sim 80\%$ (depending on the accuracy measure) [16].

1.1 Pseudoknots Are Not Well-Modeled by Most Current Tools

Structures including non-nested loops, termed pseudoknots (Fig. 1b), have remained a longstanding challenge for secondary structure prediction tools. Pseudoknots make up roughly 1.4% of base pairs [18] and are overrepresented in functionally important regions of RNA [19]. For example, pseudoknots make up the catalytic cores of many ribozymes, and they play a significant role in programmed ribosomal frameshifting in viruses [20–22]. In addition to intramolecular pseudoknots, binding between two complementary strands can often result in pseudoknot-like structures (Fig. 1c) which also play an essential role in a diverse array of biological processes [23–28].

Two major challenges arise when predicting RNA structures including pseudoknots, and many leading secondary structure prediction algorithms (e.g., Refs. [29, 30]) exclude pseudoknots from their analysis. The first is the challenge of enumerating pseudoknotted structures: the enumeration of all pseudoknotted structures into which an arbitrary sequence can fold is NP-complete [31]. The second is the challenge of computing the free energy of pseudoknotted structures, particularly their configurational entropy. Significant work over the past two decades have led to major developments on both these fronts. To address the enumeration challenge, dynamic programming approaches have been constructed that enable the polynomial-time enumeration of certain classes of pseudoknotted structures [32–39], and heuristic methods have been developed to find low (but not necessarily optimal) free energy structures [40–47]. For the second challenge, physical models have been developed for the entropies of the simplest pseudoknots [38–40, 48–50].

1.2 LandscapeFold Can Predict the Complete Secondary Structure Landscape Including Pseudoknots

LandscapeFold was developed to further address these two challenges and to enable future research into how properties of nucleic acids are influenced by their full free energy landscape.

LandscapeFold directly enumerates all possible structures into which a given sequence can fold (Subheading 3). This approach was proposed in the early days of RNA structure prediction but quickly



Fig. 1 (continued) simple intermolecular pseudoknot is depicted. Secondary structure prediction algorithms such as LandscapeFold predict hybridization by concatenating sequences separated by a linker of inert “O”s. Intramolecular base pairing can easily result in a pseudoknot, as exemplified here. The configurational entropies of such structures are difficult to predict by traditional means but are readily computable by the graphical model described in Subheading 4.3

abandoned in favor of dynamic programming methodologies [17]. While this complete enumeration is far slower than dynamic programming approaches for finding the lowest free energy non-pseudoknotted structures into which a sequence can fold, it has two particular benefits. First, it is the only way to enumerate all pseudoknotted structures. Second, it will enable further study of those RNA properties hypothesized to depend on the complete landscape rather than only the lowest free energy structures [12, 13, 15]. In particular, folding and hybridization kinetics are expected to be highly dependent on properties of the complete landscape [51].

The other major difference between LandscapeFold and other structure prediction algorithms is its pseudoknot entropy model. LandscapeFold uses a graphical formalism based on polymer physics theory which can calculate the entropy of arbitrarily complex pseudoknots (Subheading 4.3). Importantly for hybridization prediction, LandscapeFold is able to address pseudoknot-like structures that emerge in many instances of intermolecular binding, a simple example of which is shown in Fig. 1c.

In this chapter, we will give a “user’s guide” to the LandscapeFold algorithm. Throughout, we will explain the algorithm while making reference to functions found in its Python implementation. A MatLab implementation is also available.

All code is available on the GitHub repository <https://github.com/ofek-kimchi/LandscapeFold>.

2 Overall Use of the Code

2.1 Simple Example Usage

For most applications, the LandscapeFold algorithm can be run using only one line of code. For example, to calculate the free energy landscape of the short hairpin GCGCAAU⁺GCGC and save it to the variable `sol`, a user can run

```
sol = LandscapeFold(['GCGCAAU+GCGC']).mainLandscapeCalculation()
```

The code will automatically plot a diagram of the minimum free energy (MFE) structure, as well as print the top five lowest free energy structures, their free energies, and their probabilities.

The variable `sol` returned by the code above is an object of class `LandscapeFold` that defines the free energy landscape of the inputted RNA sequence. The object is initialized by calling `LandscapeFold()` with desired inputs. The function `sol.mainLandscapeCalculation()` then calculates the free energy landscape given those inputs.

The rest of this chapter will describe many of the sub-functions that go into the code above, as well as how user inputs can allow for

greater control over the results. Inputs are put into the argument of `LandscapeFold` following the list of sequences.

2.2 Python Jargon

In this chapter, we will describe several methods of the `LandscapeFold` class by referencing the class instance `sol` defined above (e.g., `sol.foo()`), and to reduce jargon, we will refer to these as “functions.” Similarly, we will refer to data attributes (e.g., `sol.bar`) by their type (e.g., `list`). We will refer to numpy arrays simply as arrays. Finally, we will refer to functions outside of the `LandscapeFold` class as, e.g., `baz()`.

2.3 The Sequences Input

The main input to `LandscapeFold`, `sequences`, is a list of up to two sequences. Each sequence is a string comprised of G’s, C’s, A’s, and either T’s or U’s depending on if the string is RNA or single-stranded DNA. Other characters are treated as unknown nucleotides, “O”s, and are not allowed to base pair.

Whether each sequence should be treated as RNA or DNA is specified by the input `DNA`. `DNA` is a list of at least the same length as `sequences`, and for each sequence is `True` if the sequence is DNA, and `False` if RNA. `LandscapeFold` attempts to correct errors in this specification: if the string contains U’s but no T’s, `LandscapeFold` assumes the sequence is RNA; if it contains T’s but no U’s, it assumes DNA. Within the `LandscapeFold` algorithm, DNA and RNA sequences differ in two major ways. First, while RNA/RNA G-U pairs are allowed, DNA/RNA G-U pairs, RNA/DNA G-T pairs, and DNA/DNA G-T pairs are all disallowed. Second, the free energies of RNA and DNA are parameterized differently (*see* Subheading 4.2). Aside from these differences, sequences are treated equivalently by the algorithm regardless of whether they represent RNA or DNA. In this chapter, we will refer to an arbitrary sequence as “RNA” since single-stranded RNA structure prediction is more common than DNA; however, everything we discuss here will be equally applicable for RNA and DNA.

3 Enumerating the Complete Free Energy Landscape

For short enough RNA molecules, the complete enumeration of all possible secondary structures is possible. The `LandscapeFold` algorithm uses a secondary structure enumeration technique developed by Pipas and McMahon in the 1970s to completely enumerate all secondary structures into which a primary sequence can fold [17]. The enumeration technique is broken up into two sub-functions, which Pipas and McMahon called `START` and `PERMU`. The `START` function enumerates all possible stems the sequence can form, where a stem is defined as a sequence of consecutive base pairs. `PERMU` seeks all realizable combinations of these stems that can coexist in the same structure.

Table 1
The main input parameters to the LandscapeFold algorithm affecting the enumeration procedure

Parameter	Type	Description	Default
sequences	List of strings	A list of sequences (up to two)	N/A
DNA	List of Booleans	For each sequence, whether it is a sequence of DNA (True) or RNA (False)	[False, False]
minBPInStem	Positive integer	Minimum length of a stem	3
allowIntramolecularPseudoknots	Boolean	Whether to enumerate structures with intramolecular pseudoknots	True
allowIntermolecularPseudoknots	Boolean	Whether to enumerate structures with intermolecular pseudoknots	True
substems	Non-negative integer or “all”	Determines the length of substems to consider	“all”
frozenBPs	$n \times 2$ nested list of integers	List of base pairs that should be present in all structures returned	empty list
minNtsInHairpin	Positive integer	Minimum number of nucleotides in a hairpin	3
onlyAllowSubsetsOfLongestStems	Boolean	Whether to only consider the longest possible stem and its subsets	False
onlyConsiderSubstemsFromEdges	Boolean	Whether to disallow subsets of stems which do not include either end of the full stem	False
onlyConsiderBondedStrands	Boolean	Whether to only include structures with at least one intermolecular base pair	False

The main input parameters to the algorithm affecting the enumeration procedure are given in Table 1.

3.1 The START Function

In order to enumerate all secondary structures, we first enumerate all possible stems that can be formed by the sequence. A stem is a set of consecutive base pairs $\{(i, j), (i+1, j-1), \dots, (i+n, j-n)\}$.

3.1.1 Determining Nucleotide Complementarity

Within LandscapeFold, the nucleotide sequence is numbered from 0 to $N-1$ from the 5' end, where N is the sequence length. We define an $N \times N$ symmetric matrix B which describes which nucleotides can bind to each other: $B_{i,j} = 1$ if nucleotides i and j can bind to make base pair $i \cdot j$ and 0 otherwise. Defining rA as an RNA adenine, and dA as a DNA adenine, etc. binding is allowed for pairs in the set

$$\{(rA, rU), (rA, dT), (rC, rG), (rC, dG), (rG, rU), (rG, dC), (rU, dA), (dA, dT), (dC, dG)\}.$$

The user can also directly input B using the `allowedBPs` input.

3.1.2 Enumerating All Possible Stems

For each nucleotide i , we search for a complementary nucleotide by traversing the sequence backwards. We check each nucleotide for complementarity until we reach $i+h$ where h is the minimum hairpin length. h can be set by the user with the `minNtsInHairpin` input. If a complementary nucleotide j is found, the stem is extended one nucleotide at a time as long as complementarity is maintained. Once complementarity is broken or the resulting hairpin length of the stem becomes too short, the stem is added to the list of stems, and we continue searching for the next nucleotide complementary to i . Following Pipas and McMahon, we call the list of stems the S-Table.

Stems are only added if they are longer than the minimum stem length, which is set by the user with the `minBPInStem` input (this parameter was termed m in Ref. [52]). Furthermore, stems are only considered valid if they do not create a hairpin that is too short; i.e. they are valid only if $j-n > i+n+h$ where (i, j) is the first base pair in the stem and n is the length of the stem.

The user can choose to, at this point, remove all stems shorter than the longest stem found, by setting the input `onlyAllowSubsetsOfLongestStems` to `True`. This is useful in some engineered systems where only one very long stem is expected to be relevant.

Next, we add all possible truncations of these enumerated stems (we call these “sub-stems”). A stem of length s has $s-n+1$ possible sub-stems of length n . By setting the input `substems` to a non-negative integer, the user can specify that only sub-stems of length at least $s-\text{substems}$ should be considered. For example, if `substems` is zero, no sub-stems will be considered. Setting the input `substems` to `all` is equivalent to setting it to an arbitrarily large number.

If the user sets the input `onlyConsiderSubstemsFromEdges` to `True`, only sub-stems that include one of the two edges of the stem will be considered, leading to two sub-stems of each length.

3.1.3 S-Table Storage and Computation Time

Stems are stored in `LandscapeFold` in two ways. One, following Pipas and McMahon, is as a list of length $2s$ (where s is the length of the stem) giving the nucleotide indices of the 5' strand, followed by their complement (e.g., `[1, 2, 3, 31, 30, 29]`). Stems are also stored in `LandscapeFold` directly as an $s \times 2$ nested list of base pairs (e.g., `[[1, 31], [2, 30], [3, 29]]`). Both of these indicate the same stem, comprised of three base pairs (where the first base pair is comprised of nucleotide 1 bound to nucleotide 31, etc.). There are two versions of the S-Table for the two storage methods: `sol.STableStructure` and `sol.STableBPs`, respectively.

The creation of the S-Table is implemented in the `LandscapeFold` algorithm by the `sol.createSTable()` function. As a practical matter, while this function is extremely fast relative to the rest of the code, the computation time the rest of the code will take can

be very roughly estimated from the number of stems enumerated, N_{stems} . If fewer than 100 stems are enumerated, the code should take less than a minute to run; if between 100–150, less than an hour; up to 200, several hours. These times were computed using a 2017 Macbook Pro with 3.1 GHz processor and 16 GB RAM.

3.1.4 Determining the Compatibility of Stems

Having created the S-Table, we will next enumerate all possible structures by finding all viable combinations of stems. In order to determine if two stems can coexist in the same structure, we define the $N_{\text{stems}} \times N_{\text{stems}}$ symmetric compatibility matrix C , where $C_{p,q} = 1$ if a structure could be made with both stems p and q , and 0 otherwise.

There are three reasons $C_{p,q}$ may be zero. (1) We impose the constraint that each nucleotide may be paired with, at most, one other nucleotide by setting $C_{p,q} = 0$ if stems p and q share at least one nucleotide. (2) We also set $C_{p,q} = 0$ if the user inputted `False` for the `allowIntramolecularPseudoknots` or `allowIntermolecularPseudoknots` arguments, and stems p and q form an intramolecular (e.g., Fig. 1b) or intermolecular (e.g., Fig. 1c) pseudoknot, respectively. (3) If stems p and q directly follow one another and are together equivalent to a single longer stem under consideration, we set $C_{p,q} = 0$. We set $C_{q,q} = 1$ for all q .

The user can input a list of base pairs that must be present in each structure considered by the algorithm using the `frozenBPs` argument. For each “frozen” base pair inputted by the user, we make a list of all stems containing that base pair (these lists are stored as a nested list in the `sol.frozenStems` property). Thus, each possible structure must include one stem from each of these lists. For each stem, we ensure it is compatible with one element from each list (i.e., it can coexist along with each of the “frozen” base pairs); if it is not, we remove the stem from the S-Table.

After making the compatibility matrix C , we have found it useful to further define three- and four-way compatibility tensors `C3` and `C4`. These allow us to ignore structures that include higher-order pseudoknots whose “minimal graphs” (*see* Subheading 4.3.2) consist of three or four stems. While our theory for pseudoknot entropy is valid for these higher-order pseudoknots, the algorithm does not currently support their entropy calculation. The user can choose to allow higher-order pseudoknots (though their free energy calculation will be inaccurate) by setting the `considerC3andC4` argument to `False`.

3.2 The PERMU Function

We are now in a position to enumerate all possible secondary structures into which the sequence can fold, by identifying all mutually compatible combinations of stems. Starting from a single stem s_1 , we consider subsequent stems s_2 and add the first stem for which $C_{s_1,s_2} = 1$. Then, we repeat the process, adding the first stem $s_3 > s_2$ compatible with both s_1 and s_2 (and, using the `C3` tensor,

compatible with s_1 and s_2 simultaneously). We continue this process until we can add no more stems.

At this point, we check if the resulting structure, composed of M stems, contains all “frozen” base pairs (if any were inputted). If so, we add it to the list of possible structures. The user can also specify that structures comprised of fewer than a given number of stems will not be added with the `minNumStemsInStructure` input, which is by default set to zero (to include also the completely unfolded structure).

If two sequences were input, the user can also choose to only add structures that include at least one intermolecular stem by setting the input `onlyConsiderBondedStrands` to `True`.

After adding (or not) the resulting structure, we then remove the last stem added, to obtain the structure composed of stems s_1, s_2, \dots, s_{M-1} , and continue the process. This algorithm returns all possible secondary structures resulting from the primary sequence.

The possible structures are stored in the list `sol.structures`, which has length $N_{\text{structures}}$. Each element of `sol.structures` is a list of stem indices (between 0 and $N_{\text{stems}} - 1$, inclusive) specifying the stems that comprise that particular structure. Thus, `sol.structures` is used in conjunction with the S-Table to determine the particular base pairs comprising each structure.

4 Performing the Free Energy Calculation

In the terminology of Pipas and McMahon, the process of calculating the free energy of each structure is termed the CHECK function. This process is completely parallelizable, though this parallelizability has not been implemented yet in the Python version of LandscapeFold (it has in the MATLAB version). Unparallelized, it is generally significantly slower than the enumeration procedure, and the loop entropy calculation in particular (Subheading 4.3) is typically the rate-limiting process.

Each structure into which an RNA sequence can fold has a corresponding enthalpy ΔH and entropy ΔS . These combine to give the free energy ΔG :

$$\Delta G = \Delta H - T\Delta S \quad (1)$$

where T is the temperature in Kelvin. T can be input to LandscapeFold using the `T` argument. By default, LandscapeFold predicts the structure landscape at 37°C . In Eq. 1 the Δ 's signify that these terms are measured with respect to the free chain. In other words, the empty structure with no base pairs will have all three of these terms equal to zero.

In equilibrium, the probability of an RNA sequence folding into a given structure σ with free energy ΔG_σ is given by the Boltzmann factor

$$p(\sigma) = \frac{\exp(-\beta\Delta G_\sigma)}{\sum_{\sigma'} \exp(-\beta\Delta G_{\sigma'})} \quad (2)$$

where $\beta = 1/k_B T$ (k_B is Boltzmann's constant). The denominator ensures that the probability distribution is normalized ($\sum_{\sigma} p(\sigma) = 1$).

There are three steps to performing the calculation in Eq. 1. First, the free energies of bonds, ΔH_{stems} and ΔS_{stems} , are calculated using the nearest-neighbor model. Second, the configurational entropy of the structure, ΔS_{loops} , is calculated. Finally, for structures that include intermolecular base pairs, penalties ΔH_{duplex} and ΔS_{duplex} are added. In other words, we assume that:

$$\begin{aligned} \Delta H &= \Delta H_{\text{stems}} + \Delta H_{\text{duplex}} \\ \Delta S &= \Delta S_{\text{stems}} + \Delta S_{\text{loops}} + \Delta S_{\text{duplex}} \end{aligned} \quad (3)$$

In Table 2 we enumerate the input parameters that affect the free energy calculation.

4.1 The Cost of Intermolecular Pairing

4.1.1 Origins of This Penalty

The penalties ΔH_{duplex} and ΔS_{duplex} are the simplest to implement in LandscapeFold. They are motivated physically by the enthalpic and entropic costs of two molecules binding (e.g., ion effects, and the translational and orientational entropies lost upon bimolecular association). The effective entropy cost is higher for more dilute solutions (i.e., larger volumes per particle). The free energy cost is expected to scale logarithmically with the particle masses as well, though experiments measuring or parameterizing this scaling are lacking [53]. The dependence of ΔH_{duplex} on the concentration of sodium in solution has been measured, finding that for lower sodium concentrations, electrostatic repulsion between the two strands leads to a higher cost of duplex formation [54]; the effects of other cations have been similarly studied [55]. The penalties also have some sequence dependence and likely differ for DNA-DNA, RNA-RNA, and DNA-RNA duplexes [54, 56–59]. While each of these effects has been studied in isolation, a comprehensive formalism combining all, or even most, of these effects remains lacking.

4.1.2 Estimates for the Penalty

The free energy cost of association has been estimated in the literature for DNA-DNA interactions to be 1.90 kcal/mol $+k_B T \ln(u_0/u)$, where $u_0 = 1\text{M}$ is a reference concentration and u is the actual concentration [60]. However, for some models (i.e., those that account for concentration elsewhere), including LandscapeFold, this penalty should be considered as independent of concentration. For such models, a value of 4.09 kcal/mol is used for the free energy cost of RNA-RNA association [61–63]; 1.96 for the free energy cost of DNA-DNA association [64]; and 3.1 for the free energy cost of RNA-DNA association [56, 65]. LandscapeFold allows the penalties to be user-defined: the input

Table 2

The main input parameters to the LandscapeFold algorithm affecting the free energy calculation. *In the current version, only one value for v_s can be inputted, even if one sequence is RNA and the other DNA

Parameters	Type	Description	Default
T	float	Temperature of the system (in Kelvin)	310.15
duplexPenalties	list of two floats	Enthalpy and entropy penalties to forming at least one intermolecular base pair (in units of kcal/mol and kcal/mol K, respectively)	[3.61, -0.0015]
concentrations	list of two floats	Concentrations of each strand (in units of M)	[1,1]
includeTerminal Mismatches	Boolean	Whether to include terminal mismatches	True
includeTerminal AUATPenalties	Boolean	Whether to include penalty for A-U, G-U, or A-T base pair ending a stem	True
includeDanglingEnds	Boolean	Whether to include dangling ends	True
includeFlush CoaxialStacks	Boolean	Whether to include flush coaxial stacks	True
considerAllAs TerminalMismatches	Boolean	Whether to treat all nucleotide pairs following a stem as a terminal mismatch	False
unmatchedBPPenalty	Boolean	Whether to substitute A for purine and C for pyrimidine for unpaired complementary bases	True
unboundButCould BindPenalties	list of two floats	Enthalpy and entropy penalties for unpaired complementary bases	[0,0]
corruptFESeed	float	Set to zero to use tabulated nearest-neighbor model parameters; non-zero to randomly modify those parameters	0
b	float	The persistence length of single-stranded RNA (or DNA) in units of nts	0.8/0.33
v_s	float*	Volume within which two nucleotides can bind in units of nts ³	0.02

duplexPenalties tells the algorithm what values to use, in units of kcal/mol and kcal/mol K (respectively), for ΔH_{duplex} and ΔS_{duplex} . The defaults correspond to RNA-RNA association penalties [61]. These terms together define a free energy cost to bimolecular association, given by $\Delta G_{\text{duplex}} = \Delta H_{\text{duplex}} - T\Delta S_{\text{duplex}}$.

4.1.3 Details of LandscapeFold Implementation

Following Ref. [66], LandscapeFold implements a correction to the user-input values by subtracting $k_B T \log(\rho_{H_2O}/(1 \text{ mol/L})) \approx 2.5$ kcal/mol from ΔG_{duplex} . This correction leads to ratios of free energies being treated as ratios of mole fractions as opposed to

molarities (*see* footnote 13 of Ref. [66]). The correction can be ignored by setting the input variable `includeRhoH2OCorrection` to `False`. This correction affects the free energies of the structures, but not the predicted equilibrium concentrations of monomers and dimers, since this factor of ρ_{H_2O} exactly cancels out with a similar factor included in the concentration calculation if `includeRhoH2OCorrection` is `True`. *See* Subheading 5.2.4 for further discussion.

Practically, these penalties are implemented by keeping track of intermolecular stems in the list `sol.linkedStems`. `sol.linkedStems` is a Boolean array of length N_{stems} which is `True` for stems that define base pairs across strands, and `False` for the rest. We also keep track of which structures include at least one stem from this list in `sol.linkedStructures`, a similar Boolean array of length $N_{\text{structures}}$. (For simplicity, `sol.linkedStructures` is an empty list if only one sequence is inputted, rather than an array in which every element is `False`). A penalty of ΔG_{duplex} is introduced for those structures that have at least one intermolecular base pair, and no penalty is introduced for structures that contain only intramolecular base pairs.

4.1.4 Symmetry Penalties

If the two sequences input are identical, then structures with a 2-fold symmetry have an extra free energy penalty of $k_B T \ln 2$ [66]. This penalty is effectively taken into account through our complete enumeration approach: asymmetric structures will be considered twice, while symmetric structures are considered only once. Thus, no further penalty need to be applied at this stage.

To illustrate, consider two of the structures that the self-complementary sequence “GCAGC” can form: one in which the 5′ end of the first strand is bound to the 5′ end of the second; the other in which the 5′ end of the first strand is bound to the second’s 3′ end. The former structure is enumerated only once. The latter, however, is enumerated twice: the same structure is considered again as the structure where the 3′ end of the first strand is bound to the 5′ end of the second. This differential in the structure enumeration is a direct result of the symmetry of the former structure and the asymmetry of the latter, and effectively adds a $k_B T \ln 2$ penalty to the former structure compared to the latter.

4.2 The Stem Free Energy Model

4.2.1 The Basic Nearest-Neighbor Free Energy Model

The nearest-neighbor free energy model has shown decades of success in accurately estimating the free energies of both intra- and intermolecular bonds of RNA and DNA molecules. The details of the model are best described elsewhere [63, 67]. Here we will give only a brief overview of the model and a guide to how to modify it within the `LandscapeFold` algorithm as desired.

The backbone of the nearest-neighbor model is that the enthalpy and entropy of a stem can be well approximated by

considering each neighboring base pair independently. For example, consider the bottom stem in Fig. 1a:

5'	C	U	C	C	G	G	U	3'
3'	C	A	G	G	C	C	U	5'

where we have included the terminal mismatches as well. Terminal mismatches are the two (unpaired) nucleotides following the last base pair in the stem or preceding the first base pair. Within the nearest-neighbor approximation, the enthalpy and entropy of this stem can be calculated by summing up the enthalpies and entropies of each of the following neighboring base pairs:

5' C U 3'	5' UC 3'	5' CC 3'	5' CG 3'	5' GG 3'	5' G U 3'
3' C A 5'	3' AG 5'	3' GG 5'	3' GC 5'	3' CC 5'	3' C U 5'

The enthalpies and entropies of every possible set of neighboring base pairs (including terminal mismatches) have been tabulated for both RNA and DNA [63, 64]. In the LandscapeFold algorithm, the tables for RNA/RNA, DNA/DNA, and RNA/DNA bonds are given by `bondFreeEnergies()` based on data from Refs. [53, 56, 61, 63, 64, 67–74]. For RNA/DNA hybrids, however, parameters for some terminal mismatches have not been tabulated. For these, LandscapeFold assumes that their enthalpies and entropies are given by the means of the RNA/RNA and DNA/DNA parameters. Terminal mismatches can be ignored in the free energy calculation by setting the input `includeTerminalMismatches` to `False`.

4.2.2 Terminal A-U, G-U, and A-T Penalties

When a stem starts or ends with an A-U, G-U, or A-T base pair, an enthalpy and entropy penalty are introduced by the nearest-neighbor model. These penalties are given by the `terminalAUATPenalties()` function in LandscapeFold. The parameters for the A-T penalties are given in Ref. [64]; for A-U and G-U pairs (which are treated equivalently), the penalties comes from Ref. [53]. For RNA/DNA hybrids, A-T pairs are given the DNA penalties and A-U pairs the RNA penalties. These penalties can be ignored in the free energy calculation by setting `includeTerminalAUATPenalties` to `False`.

The A-T penalties are: $\Delta H_{\text{penalty}} = 2.2$ kcal/mol; $\Delta S_{\text{penalty}} = 6.9 \times 10^{-3}$ kcal/mol *K*. The A-U penalties are: $\Delta H_{\text{penalty}} = 3.72$ kcal/mol; $\Delta S_{\text{penalty}} = 1.05 \times 10^{-2}$ kcal/mol *K*.

4.2.3 Dangling Ends

LandscapeFold also accounts for dangling ends, base pairs adjacent to a single nucleotide (for example, the rightmost stem in Fig. 1a). These parameters are tabulated in the `danglingEndMatrices()` function for RNA/RNA bonds [61] and for DNA/DNA bonds

[75]. For RNA/DNA bonds, LandscapeFold uses the RNA/RNA parameters if the dangling nucleotide belongs to an RNA molecule, and the DNA/DNA parameters if it belongs to a DNA molecule. Dangling ends can be ignored in the free energy calculation by setting the input `includeDanglingEnds` to `False`.

4.2.4 Flush Coaxial Stacks

If two stems are separated by a bulge loop (i.e., two adjacent nucleotides are bound to two non-adjacent nucleotides) we have a “flush coaxial stack.” The nearest-neighbor model calculates the free energy as if the bulge was not present and the two stems were continuations of one another. LandscapeFold differs from the standard nearest-neighbor model [76] in that it considers flush coaxial stacks for any bulge loop and not just those of length one, since these stacks compensate for LandscapeFold’s higher configurational entropy cost of forming bulge loops. LandscapeFold also differs from the standard models in that in the presence of a three-way junction where each stem is flush with the next, LandscapeFold considers two flush coaxial stacks, while previous methodologies argue for considering only the most energetically favorable stack, and treating the other as a dangling end [63]. Flush coaxial stacks can be ignored by setting the input `includeFlushCoaxialStacks` to `False`.

The user can also use the `considerAllAsTerminalMismatches` input to ignore both dangling ends and flush coaxial stacks. If this is set to `True`, all ends of stems are treated as terminal mismatches (even if the next nucleotides over are both bound as part of different stems).

4.2.5 Terminal Mismatches Which Could Bind

Another element of the nearest-neighbor model is a free energy penalty for terminal mismatches which could have been paired in a different structure. If two nucleotides are complementary but unpaired in a given structure, the purine is replaced by an A and the pyrimidine by a C for the purposes of the free energy calculation [76]. This modification is made in LandscapeFold if the `unmatchedBPPenalty` input is set to `True`, keeping RNA nucleotides as RNA and DNA as DNA.

Whether or not this modification is made, the user can choose to introduce an alternative penalty with the `unboundButCouldBindPenalties` input. This input is a list of two floats, where the first gives an enthalpic cost to each set of complementary unpaired nucleotides, and the second is an entropic cost.

Other minor differences between LandscapeFold’s implementation of the nearest-neighbor model and others’ are described in Ref. [52].

4.2.6 *Modifying the Nearest-Neighbor Model Parameters*

The nearest-neighbor model parameters are imperfect due to errors in their measurement, approximations made by the model, and different experimental conditions [18, 76, 77]. In order to determine if a prediction is stable to variations in the model parameters, we introduce a function `corruptBondFreeEnergies()`. This function returns parameters in the same form as the `bondFreeEnergies()` function, but modifies the parameters by multiplying them by a random multivariate Gaussian to introduce errors of 6.5%, 7.3%, and 2.4% in ΔH_{stems} , ΔS_{stems} , and ΔG_{stems} (at 37 °C), respectively [61]. These percentages can be changed by the user and are given as inputs to the `corruptBondFreeEnergies()` function. The error in ΔG_{stems} is lower than the other two because of extremely high correlations (~ 1) between measurement errors in ΔH_{stems} and ΔS_{stems} [61]. These corrupted parameters are used in place of those from the `bondFreeEnergies()` function if the input `corruptFESeed` is set to a non-zero value. If it is, it serves as the seed for the random number generator in order to ensure reproducibility.

4.3 *The Configurational Loop Entropy Model*

The full derivation of the configurational loop entropy model can be found in Ref. [52]. Here, we will provide a guide to implementing the model. The process has seven steps:

1. Convert the RNA structure to a graph, where each node is the base pair at the edge of a stem (each stem thus yields two nodes). Nodes are connected by two types of edges, representing single- and double-stranded RNA.
2. Count the number of double-stranded edges present. This will determine the number of factors of v , in the final equation. If any intermolecular stems are present in the structure, subtract one from that number.
3. Remove “bridges,” which are edges whose removal disconnects the graph.
4. Remove nodes disconnected from other nodes. Any nodes that are connected only to two single-stranded edges can similarly be removed, and the two edges concatenated.
5. For each resulting disconnected graph, convert the graph to an integral. The positions of each node but one are integrated over three-dimensional space, and the integrands are given by the bonds: double-stranded bonds are converted to delta functions (Eq. 4), while single-stranded bonds are converted to Gaussians (Eq. 5).
6. Perform the integrals. These can either be done by hand or numerically, as described in detail in Ref. [52]. All integrals that involve up to two double-stranded edges can be performed by hand, and the LandscapeFold algorithm has those results hard-coded in.

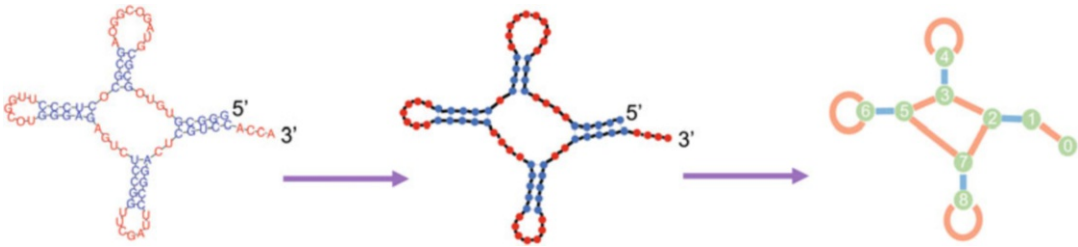


Fig. 2 Graph construction. The process of converting from a structure to a graph (steps 1–2). Graphs are sequence independent (middle). Nodes correspond to base pairs at the ends of each stem. Blue edges represent double-stranded RNA connecting the nodes; red edges represent single-stranded RNA. For clarity, we added a node corresponding to the final RNA nucleotide, as LandscapeFold does. Such nodes can be added or not; they are removed as part of the graph decomposition process (Fig. 3). For clarity we number the nodes 0–8

7. Multiply the integrals by one another and by v_s raised to the appropriate power (determined by Step 2). Finally, take the natural logarithm and multiply by Boltzmann’s constant k_B to get the configurational loop entropy of the structure.

4.3.1 Converting from a Structure to a Graph

The process of converting a structure to a graph (steps 1–2) is depicted in Fig. 2. The structure under consideration is shown on the left. The graph is sequence independent (middle) and is constructed by placing nodes at the two edges of each stem. For clarity, it is useful to make the first and last nucleotides into their own nodes, though these will be removed as part of the graph decomposition process.

Nodes constructed from the same stem are connected by one type of edge corresponding to double-stranded RNA (blue). Another type of edge represents single-stranded RNA connecting the nodes (red).

Nodes that do not correspond to the first or last nucleotide of an RNA molecule are always connected to one double-stranded and two single-stranded edges. A node connected to itself by a single-stranded edge has no other single-stranded edge connections.

The graph construction process is implemented in LandscapeFold by the `createGraphFromStructure()` function.

4.3.2 Decomposing the Graph into Minimal Graphs

The most time-consuming step of the LandscapeFold algorithm as a whole is the graph decomposition process (steps 3–4). This process is depicted in Fig. 3, and is implemented in LandscapeFold by the `graphDecomposition()` function.

We start with the graph previously constructed. It now becomes important to note that at the time of graph construction, each edge is given a length associated with it. The length of double-stranded edges l_i is one fewer than the number of base pairs in the corresponding stem (e.g., in the figure, $l_1 = 3$; $l_2 = l_3 = l_4 = 4$). The

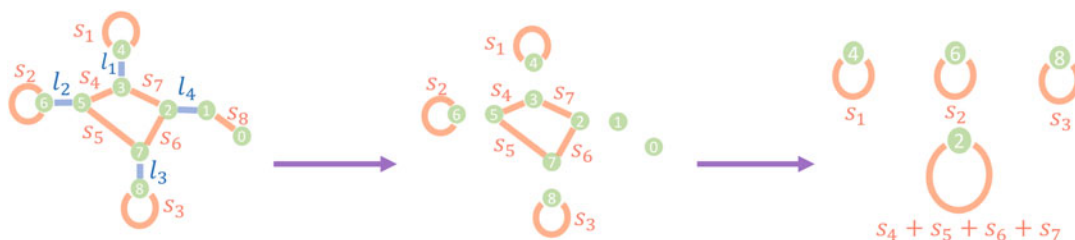


Fig. 3 Graph decomposition. The graph decomposition process (steps 3–4) is depicted. We keep track of the length of RNA corresponding to each edge (s_i and l_i in the figure). After being created (left) the graph is decomposed into its minimal graphs, where each minimal graph cannot be disconnected by the removal of any edge (middle). Nodes disconnected from any edge are then removed. Nodes connected only to two single-stranded edges are removed one by one, and the two edges merged (right). For this structure, $l_1 = 3$; $l_2 = l_3 = l_4 = 4$; $s_1 = 11$; $s_2 = 8$; $s_3 = 8$; $s_4 = 2$; $s_5 = 5$; $s_6 = 3$; $s_7 = 5$; $s_8 = 5$

length of single-stranded edges s_i is one more than the number of nucleotides in between the stems (in the figure, $s_1 = 11$; $s_2 = 8$; $s_3 = 8$; $s_4 = 2$; $s_5 = 5$; $s_6 = 3$; $s_7 = 5$; $s_8 = 5$).

The graph decomposition process consists of two steps. The first is edge removal: if the removal of an edge (single- or double-stranded) disconnects the graph, that edge is removed and the graph is disconnected. This process is depicted by the middle panel of Fig. 3.

The second step of graph decomposition is node removal: disconnected nodes (e.g., nodes 0 and 1 in the figure) are removed. Then, any node that is connected only to two single-stranded edges can similarly be removed, and the two edges concatenated. Thus in the figure, the cycle consisting of nodes 2, 3, 5, and 7 is substituted for a single node connected to itself by a single-stranded edge of length $s_4 + s_5 + s_6 + s_7$. Each of the minimal graphs resulting from the graph decomposition process can now be treated independently.

LandscapeFold currently has hard-coded the entropies of all structures whose minimal graphs consist of no more than two stems.

4.3.3 Converting Each Graph to an Integral

The graph represents the entropy of the RNA in integral form. The conversion of each graph to the configurational entropy of the RNA (steps 5–7) is implemented by the `calculateEntropyFromGraph()` function. In order to explain how LandscapeFold calculates the entropy of an RNA structure from the graphs found in the previous section, we show here how to perform the same calculation by hand.

For each graph, the positions of each node but one are integrated over three-dimensional space. These positions are measured with respect to the fixed node. In other words, the fixed node is placed at the origin. The integrands are determined by the edges of the graph.

Double-stranded edges correspond to rigid stems, and, therefore, to a delta function in the integrand keeping the distance between the nodes fixed. For example, a double-stranded edge of length l_{12} connecting nodes 1 and 2 corresponds to a term

$$\frac{\delta(|\vec{r}_1 - \vec{r}_2| - l_{12})}{4\pi l_{12}^2}$$

in the integrand, where \vec{r}_i is the position of node i in three-dimensional space, and the absolute value signs represent the magnitude of the vector $\vec{r}_1 - \vec{r}_2$. The delta function is defined such that for any function $f(r)$ (where $r = |\vec{r}|$),

$$\int \delta(r - l)f(r)dr = f(l) \quad (4)$$

as long as l is within the limits of integration (the integral yields zero otherwise).

Single-stranded edges correspond to flexible unpaired RNA. The persistence length of single-stranded RNA is denoted b and is approximately equal to 0.8 nm [78]. The persistence length of single-stranded DNA is similar [79]. The persistence length in units of nucleotides (nts, approximately 1/3 nm) can be input to LandscapeFold through the input b . For concision, much of LandscapeFold is written using a parameter $\gamma = 3/2b$ instead of b directly.

A single-stranded edge of length s_{12} connecting nodes 1 and 2 corresponds to a Gaussian term $P_{s_{12}}(\vec{r}_1 - \vec{r}_2)$ in the integrand:

$$\begin{aligned} P_{s_{12}}(\vec{r}_1 - \vec{r}_2) &= \left(\frac{3}{2\pi s_{12}b}\right)^{3/2} \exp\left(-\frac{3(\vec{r}_1 - \vec{r}_2)^2}{2s_{12}b}\right) \\ &= \left(\frac{\gamma}{\pi s_{12}}\right)^{3/2} \exp\left(-\frac{\gamma(\vec{r}_1 - \vec{r}_2)^2}{s_{12}}\right) \end{aligned} \quad (5)$$

4.3.4 Graph Decomposition Revisited

It is worth mentioning that the graph decomposition process is merely a visual way of performing the simplest of these resulting integrals. Edges that disconnect the graph can be removed because the resulting disconnected graphs correspond to separable integrals, and because $\int d\vec{r} P_s(\vec{r}) = \int d\vec{r} \delta(|\vec{r}| - l)/4\pi l^2 = 1$. Similarly, nodes connected only to two single-stranded edges can be removed and the edges concatenated because $\int P_x(\vec{r}_1)P_y(\vec{r}_2 - \vec{r}_1) d\vec{r}_1 = P_{x+y}(\vec{r}_2)$.

4.3.5 Using the Integrals to Calculate the Configurational Entropy

After writing down the relevant integrals, we multiply together the results for each minimal graph into which the structure was decomposed. Since we ultimately take the logarithm of these results to get

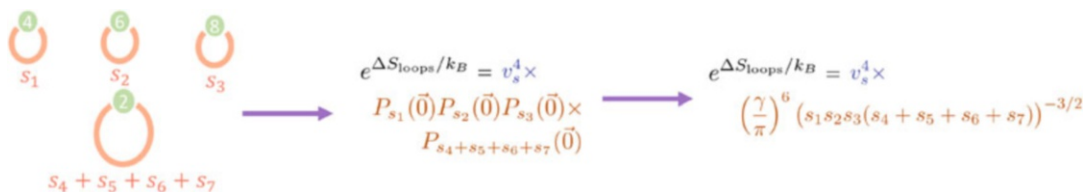


Fig. 4 Entropy calculation. The minimal graphs (left) are directly converted to integral form (middle). For non-pseudoknotted structures, each minimal graph corresponds to a factor of $P_s(\vec{0})$ (Eq. 5). The results from each minimal graph are multiplied together and by four factors of v_s from the four stems in the original graph (Fig. 2)

the loop entropy, multiplication here is equivalent to summing the entropies of each minimal graph to get the total entropy.

We also multiply by a factor v_s^r , where v_s is the volume within which two nucleotides can bind, and r is given by the number of stems present in the original structure (e.g., four for the structure in Fig. 2 whose minimal graphs are shown in Fig. 4). However, if the structure under consideration includes any intermolecular stems, r is subtracted by one, since the first intermolecular stem is considered separately by the ΔS_{duplex} term.

The user can specify a value for v_s to use in LandscapeFold, in units of nts^3 , using the v_s input. Currently, only a single value of v_s can be specified, even if one sequence is RNA and the other is DNA. We found previously that $v_s = 0.020 \pm 0.004 \text{ nts}^3$ for RNA by comparing Eq. 5 to previously determined entropy costs of forming hairpins of different lengths (i.e., different values of s_{12} , with $|\vec{r}_1 - \vec{r}_2| = 0$) [52]. A similar analysis on DNA using data on hairpins of lengths 3–8 from Ref. [64] finds a significantly different best-fit value of $v_s = 0.38 \pm 0.06 \text{ nts}^3$ for single-stranded DNA. Due to lack of similar data for RNA-DNA bonds, it is unclear what an appropriate value for v_s for RNA-DNA bonds should be.

The result of these integrations, after multiplying by the appropriate factors of v_s , is the exponential of the entropy, normalized by Boltzmann's constant: $e^{\Delta S_{\text{loops}}/k_B}$. Thus, we take the natural logarithm of the result (which is unitless) and multiply by k_B to get the configurational loop entropy of the structure.

4.3.6 The Entropy of Non-pseudoknotted Structures

In Fig. 4 we show the resulting (disconnected) minimal graphs from Figs. 2 and 3. The graphs are all identical in form: each is a node connected to itself by a single-stranded edge of a certain length. As previously mentioned, in order to convert from a graph to an integral we integrate over the positions of all nodes but one; therefore, since only one node is present in each graph, we do not need to compute any integrals. We instead multiply the appropriate factors of $P_s(\vec{0})$ by one another (one for each minimal graph), and multiply the result by v_s^4 . The result is shown in the rightmost panel.

In fact, any structure that contains no pseudoknots will ultimately contain no integrals after the graph decomposition process, and will only contain factors of $P_s(\mathbf{0})$. A non-pseudoknotted structure will always be converted to a set of single nodes connected to themselves by a single-stranded edge. These represent internal loops, bulge loops, hairpin loops, or multiloops; all take the same form for their configurational entropy within our polymer physics model.

4.3.7 The Entropy of Pseudoknotted Structures

In Figs. 5 and 6 we show two further examples of converting from a structure to the integral representing its configurational entropy.

In Fig. 5 we consider a simple pseudoknot, termed the H-type pseudoknot. We convert from the structure to its respective graph, which contains two double-bonded edges and three single-bonded edges. The resulting graph is not disconnected by the removal of any edge and is, therefore, minimal. It corresponds to the integrals shown in the rightmost panel of the figure. In that equation, the positions of three of the four nodes are integrated over all of three-dimensional space. Two delta functions (corresponding to the two stems) and three Gaussians (corresponding to the three single-stranded edges) are present in the integrand, as are the two factors of v_s . The result of this integration is not shown, but a step-by-step demonstration of how to perform this and similar Gaussian integrals with delta functions is given in Ref. [52].

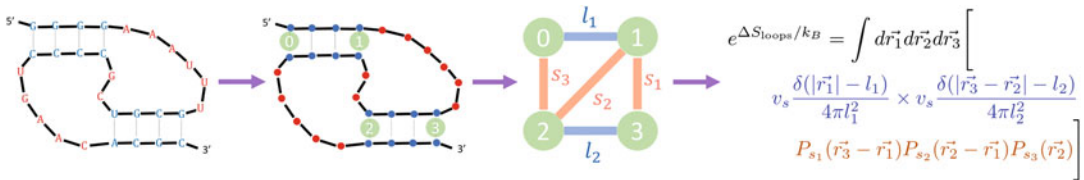


Fig. 5 Intramolecular pseudoknot entropy example. A simple pseudoknot is converted to a graph, and from there to integral form. The positions of all nodes but one are integrated over three-dimensional space. Each double-stranded edge corresponds to a delta function in the integrand; each single-stranded edge corresponds to a Gaussian P_s . Two factors of v_s are included for the two stems. For this structure, $l_1 = l_2 = 3$; $s_1 = 7$; $s_2 = 6$; $s_3 = 3$. Figure is adapted from Ref. [52]

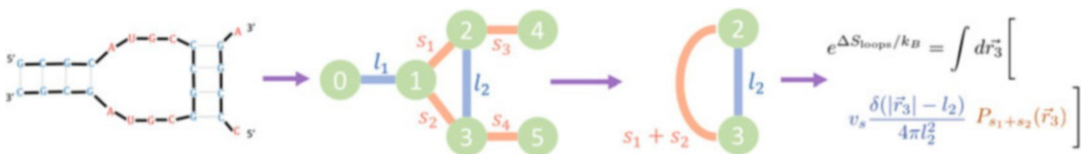


Fig. 6 Intermolecular pseudoknot entropy example. A simple intermolecular pseudoknot is converted to a graph, decomposed into its minimal graphs, and converted to integral form. Since there are intermolecular stems, one fewer factor of v_s is included than the number of total stems. For this structure, $l_1 = l_2 = 3$; $s_1 = s_2 = 5$

In Fig. 6 we consider a simple intermolecular pseudoknot. In this example, once the structure is converted to a graph, the graph can be decomposed further into a simple minimal graph. The configurational entropy of the full structure can be found by converting the graph to its integral form. There are two stems in the original structure, which in a single-molecule structure would correspond to two factors of v_s in the final equation. However, since the structure includes at least one intermolecular stem, we have $2 - 1 = 1$ factor of v_s present in the equation for ΔS_{loops} .

5 The Results of the LandscapeFold Calculation

5.1 Accessing the Structures and Their Free Energies

The results of the landscape calculation are stored in the `LandscapeFold` object (`sol` in the example from Subheading 2). For example, `sol.STableBPs` and `sol.STableStructure` are the two versions of the S-Table discussed in Subheading 3.1.2. Similarly, `sol.structures` stores the list of stems present in each structure, where each stem is referred to by its index in the S-Table.

The ordering of `sol.structures` is determined by the enumeration procedure. However, `sol.indexSort` is an array that provides a more practical ordering for the structures: its first element is the index of the minimum free energy (MFE) structure, its second element is the index of the second-lowest free energy structure, etc. In order to examine the specific base pairs comprising low free energy structures, the function `sol.MFEStructures(n)` returns a list of the n lowest free energy structures where here the base pairs making up each structure are given.

Similarly, `sol.sortedFES` and `sol.sortedProbs` are sorted arrays providing, respectively, the free energies and equilibrium probabilities (Eq. 2) of each structure. To examine each component of the free energy in more detail, the arrays `sol.allBondEnergies`, `sol.allBondEntropies`, `sol.allLoopEntropies`, and `sol.allDuplexEntropies` yield ΔH_{stems} , ΔS_{stems} , ΔS_{loops} , and ΔS_{duplex} , respectively, in the same ordering as the `sol.structures` list.

5.2 Multiple Sequences

5.2.1 Implementation of Multiple Sequences in LandscapeFold

If two sequences are inputted, `LandscapeFold` will return all possible structure pairs into which they can fold, some of which include only intramolecular base pairs, and some of which include intermolecular base pairs. For example, consider two sequences s_1 and s_2 . Sequence s_1 can fold into structures s_1^1, s_1^2, s_1^3 , and sequence s_2 can fold into structures s_2^1 and s_2^2 . They can also bind to one another to form a structure s_{12}^1 . In this case, the elements of `sol.structures` will be the elements of the set: $\{(s_1^1, s_2^1), (s_1^1, s_2^2), (s_1^2, s_2^1), (s_1^2, s_2^2), (s_1^3, s_2^1), (s_1^3, s_2^2), s_{12}^1\}$, and the elements of, e.g., `sol.sortedFES` will be the (sorted) total free energies of

each structure pair. Each of these structure pairs is thus treated the same way an individual structure is treated for a unimolecular input.

Bimolecular structure landscapes are considered by concatenating the two sequences and separating them by a linker of “O”s which is disregarded in free energy calculations [29]. We use a linker of 6 nucleotides (or more precisely, twice the user-specified minimum number of nucleotides in a hairpin). The indices of nucleotides corresponding to the linker are stored in the `sol.linkerPos` array.

When using inputs such as `frozenBPs` on a bimolecular landscape, the nucleotides are numbered assuming that the linker is present. For example, in order to specify that G must bind to C for the sequences pair [GUU, AAC], the user should input `frozenBPs=[[0, 11]]`.

5.2.2 Potential Speed-ups for Multiple Sequences

In practice, treating multiple sequences by concatenation can also lead to wasted computation time (e.g., by recalculating the free energy of structure s_1^1 multiple times, for each structure s_2^i with which it is paired). The function `sol.twoStrandLandscapeCalculation()` cuts down on that wasted computation time by first treating each sequence separately, and next considering only those structures that include intermolecular base pairs, thus enumerating each structure only once. LandscapeFold does not currently consider homo-dimers in this calculation.

5.2.3 Prediction of Monomer and Dimer Concentrations: User Inputs and Outputs

If multiple sequences are input, LandscapeFold also calculates the equilibrium concentrations of the monomer and dimer species. The total concentration of each strand (in units of M) is input using the `concentrations` variable. LandscapeFold stores the predicted equilibrium concentrations of monomers and dimers in the variable `sol.equilibriumConcentrations`. It is generally a list of three values: the concentration of the first monomer, of the second monomer, and of the dimer.

In the case where the two sequences are identical, only the first element of the `concentrations` input is used. In this case, `sol.equilibriumConcentrations` is a list of only two values: the concentration of the monomers, and of the dimers.

5.2.4 Prediction of Monomer and Dimer Concentrations: Details of LandscapeFold's Process

Equilibrium concentrations are calculated by finding a simultaneous solution to a set of equations. Letting the total concentration of the first strand be c_1 and of the second strand be c_2 (as determined by the `concentrations` input), the equilibrium monomer concentration of the two strands be c_{m_1} and c_{m_2} , respectively, and the equilibrium dimer concentration be c_d , there are two conservation laws given by

$$\begin{aligned}c_{m_1} + c_d &= c_1 \\c_{m_2} + c_d &= c_2\end{aligned}\tag{6}$$

If the two sequences are identical, there is only one conservation law ($c_{m_1} + 2c_d = c_1$).

The relative ratios of the monomer and dimer concentrations are determined by the ratio of Boltzmann factors. We let Z_{m_1} be defined as $Z_{m_1} = \sum_{\sigma_1} \exp(-\beta G_{\sigma_1})$, where the sum is over all monomeric structures of the first strand σ_1 each of which has free energy G_{σ_1} , and we let Z_{m_2} be similarly defined. Letting Z_m^2 be equal to the product $Z_{m_1} Z_{m_2}$, it can be seen that Z_m^2 is defined as a similar sum over all monomeric structure pairs. Finally, Z_d is defined as a similar sum over all dimeric structures: $Z_d = \sum_{\sigma_d} \exp(-\beta G_{\sigma_d})$. With these definitions in hand, we can write the final equation constraining our system [66]

$$\frac{c_d \rho_{H_2O}}{c_{m_1} c_{m_2}} = \frac{Z_d}{Z_m^2}\tag{7}$$

where the factor of $\rho_{H_2O} \approx 55$ M exactly cancels out with the correction factor introduced in Subheading 4.1.3. The factor of ρ_{H_2O} is omitted from this formula if `includeRhoH2OCorrection` is set to `False`.

LandscapeFold automatically solves this simultaneous set of equations to find the equilibrium monomer and dimer concentrations. If the two sequences are identical, the product $c_{m_1} c_{m_2}$ is replaced by c_m^2 in Eq. 7.

5.3 Re-running the Code with Different Parameters

LandscapeFold stores results in a way that makes it easy to examine how changes to the nearest-neighbor parameters or the entropy model parameters affect the landscape. In essence, LandscapeFold stores for each secondary structure how each parameter will affect the free energy of that structure. Then, by running the function `sol.postCalculationFxn()`, LandscapeFold can quickly recalculate the free energy of each structure with given modified parameters. As a general estimate, `sol.postCalculationFxn()` takes about 10% of the total calculation time. In this section, we describe how it is implemented.

The property `sol.allComponentGraphs` is a list of length $N_{\text{structures}}$. For each structure, it keeps track of how many instances of each type of minimal graph are present in that structure, and the lengths of each edge in those graphs. In addition, `sol.allNumVs` is an array keeping track of how many factors of v_s are included in each structure integral. With these arrays, if the parameters b or v_s are modified, the configurational loop entropy of each structure can be quickly recalculated without needing to again convert each structure to a graph and perform graph decomposition.

The property `sol.bondFECOUNTS` is also a list of length $N_{\text{structures}}$. It holds for each structure a set of sparse arrays

describing how many instances of each set of possible neighboring base pairs are present in that structure. For example, for each structure, it holds a sparse array with $4 \times 4 \times 4 = 64$ elements yielding the number of instances of each of the 64 possible neighboring DNA/DNA base pairs present in that structure. That array can then be multiplied by arrays storing the energies and entropies of each of these neighboring pairs (according to the nearest-neighbor model) to yield the DNA/DNA contributions to ΔH_{stems} and ΔS_{stems} . `sol.bondFECounts` also stores similar lists of sparse arrays for RNA/RNA and RNA/DNA nearest neighbors, as well as for the number of terminal A-U, G-U, and A-T base pairs present in each structure (Subheading 4.2.2).

The matrices `sol.dangling5Count` and `sol.dangling3Count` store similar lists of sparse matrices giving for each structure how many of each possible 3' and 5' dangling ends are present in that structure (Subheading 4.2.3). `sol.unboundButCouldBindCounts` stores how many complementary terminal mismatches are present in each structure (Subheading 4.2.5) allowing the user to easily and quickly examine how changing the penalty for these affects the overall structure landscape.

Thus, only a few simple matrix multiplications need to be performed in order to examine how modifications to the nearest-neighbor parameters affect the complete free energy landscape of a set of sequences.

5.4 Returning Graph Topologies

Considering the graph associated with each structure (Subheading 4.3.1) is a useful way to coarse-grain over similar structures by their topologies. In order to instruct the algorithm to store information regarding graphs, the user can set the input `storeGraphs` to `True`. In that case, the unique list of graphs corresponding to structures enumerated by the algorithm is given by `sol.structureGraphList` (a list of length $N_{\text{graphs}} \leq N_{\text{structures}}$). The index of that list to which each structure corresponds is given by the length- $N_{\text{structures}}$ array `sol.allWhichStructureGraph`.

By summing over the equilibrium probabilities of each structure corresponding to a given graph, we can get the equilibrium probability of that topology forming. That information is stored in the sorted array `sol.sortedGraphProbs`, while the array `sol.indexSortedGraphProbs` provides the mapping between the sorted and unsorted graph orderings.

5.5 Visualizing Results

If the user sets the input `makeFigures` to `True`, `LandscapeFold` will automatically make three plots at the end of the calculation. The first two visualize the minimum free energy structure found in planar graph and circle diagram formats. These are implemented using the `drawRNAstructure()` and `drawRNAstructureSeqCircle()` functions, respectively, which take as inputs a structure to visualize and its corresponding sequence.

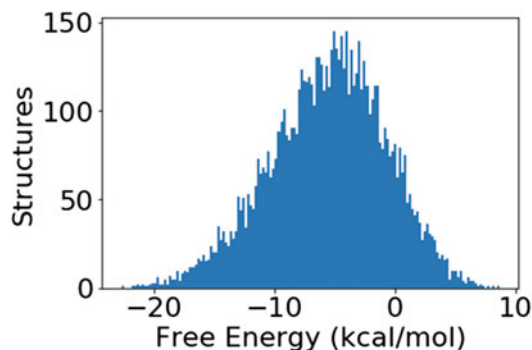


Fig. 7 Histogram of structure free energies. A histogram of the free energies of all structures with a minimum stem length of 4 nts into which the sequence shown in Fig. 1a can fold

The third plot made is a histogram of the free energies of all structures returned by the algorithm (Fig. 7) implemented with the `sol.histFEs()` function.

References

- Ahmed W, Zheng K, Liu ZF (2016) Small non-coding RNAs: new insights in modulation of host immune response by intracellular bacterial pathogens. *Front Immunol* 7:1–10. ISSN 16643224. <https://doi.org/10.3389/fimmu.2016.00431>
- Boyd SD (2008) Everything you wanted to know about small RNA but were afraid to ask. *Lab Invest* 88(6):569–578. ISSN 00236837. <https://doi.org/10.1038/labinvest.2008.32>
- Zhang DY, Winfree E (2009) Control of DNA strand displacement kinetics using toehold exchange. *J Am Chem Soc* 131(47):1–16. ISSN 00027863. <https://doi.org/10.1021/ja906987s>
- Melamed S, Peer A, Faigenbaum-Romm R, Gatt YE, Reiss N, Bar A, Altuvia Y, Argaman L, Margalit H (2016) Global mapping of small RNA-target interactions in bacteria. *Mol Cell* 63(5):884–897. ISSN 10974164. <https://doi.org/10.1016/j.molcel.2016.07.026>
- Ramanathan M, Porter DF, Khavari PA (2019) Methods to study RNA–protein interactions. *Nat Methods* 16(3):225–234. ISSN 15487105. <https://doi.org/10.1038/s41592-019-0330-1>
- Ellington AD, Szostak JW (1990) In vitro selection of RNA molecules that bind specific ligands. *Nature* 346:818–822. ISSN 0028-0836. <https://doi.org/10.1038/346818a0>
- Tuerk C, Gold L (1990) Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. *Science* 249(4968):505–510. ISSN 0036-8075. <https://doi.org/10.1126/science.2200121>
- Robertson DL, Joyce GF (1990) Selection in vitro of an RNA enzyme that specifically cleaves single-stranded DNA. *Nature* 344(6265):467–468. ISSN 0028-0836. <https://doi.org/10.1038/344467a0>
- Olea C, Joyce GF (2016) Real-time detection of a self-replicating RNA enzyme. *Molecules* 21(10):1–12. ISSN 14203049. <https://doi.org/10.3390/molecules21101310>
- Spitale RC, Flynn RA, Zhang QC, Crisalli P, Lee B, Jung JW, Kuchelmeister HY, Batista PJ, Torre EA, Kool ET, Chang HY (2015) Structural imprints in vivo decode RNA regulatory mechanisms. *Nature* 519(7544):486–490. ISSN 14764687. <https://doi.org/10.1038/nature14263>
- Doudna JA (2000) Structural genomics of RNA. *Nat Struct Biol* 7:954–956. ISSN 10728368. <https://doi.org/10.1038/80729>
- Ritchie DB, Foster DA, Woodside MT (2012) Programmed –1 frameshifting efficiency correlates with RNA pseudoknot conformational

- plasticity, not resistance to mechanical unfolding. *Proc Nat Acad Sci USA* 109(40): 16167–16172. ISSN 00278424. <https://doi.org/10.1073/pnas.1204114109>
13. Wan Y, Kertesz M, Spitale RC, Segal E, Chang HY (2011) Understanding the transcriptome through RNA structure. *Nat Rev Genetics* 12(9):641–655. ISSN 14710056. <https://doi.org/10.1038/nrg3049>
 14. Barrick JE, Breaker RR (2007) The distributions, mechanisms, and structures of metabolite-binding riboswitches. *Genome Biol* 8(11). ISSN 14747596. <https://doi.org/10.1186/gb-2007-8-11-r239>
 15. Mortimer SA, Kidwell MA, Doudna JA (2014) Insights into RNA structure and function from genome-wide studies. *Nat Rev Genet* 15(7): 469–479. ISSN 14710064. <https://doi.org/10.1038/nrg3681>
 16. Mathews DH (2019) How to benchmark RNA secondary structure prediction accuracy. *Methods* 162–163:60–67. ISSN 10959130. <https://doi.org/10.1016/j.ymeth.2019.04.003>
 17. Pipas JM, McMahon JE (1975) Method for predicting RNA secondary structure. *Proc Nat Acad Sci* 72(6):2017–2021. ISSN 0027-8424. <https://doi.org/10.1073/pnas.72.6.2017>
 18. Mathews DH, Turner DH (2006) Prediction of RNA secondary structure by free energy minimization. *Curr Opin Struct Biol* 16(3): 270–278. ISSN 0959440X. <https://doi.org/10.1016/j.sbi.2006.05.010>
 19. Hajdin CE, Bellaousov S, Huggins W, Leonard CW, Mathews DH, Weeks KM (2013) Accurate SHAPE-directed RNA secondary structure modeling, including pseudoknots. *Proc Nat Acad Sci* 110(14):5498–5503. ISSN 0027-8424. <https://doi.org/10.1073/pnas.1219988110>
 20. Staple DW, Butcher SE (2005) Pseudoknots: RNA structures with diverse functions. *PLoS Biol* 3(6):0956–0959. ISSN 15449173. <https://doi.org/10.1371/journal.pbio.0030213>
 21. De Messieres M, Chang JC, Belew AT, Meskauskas A, Dinman JD, La Porta A (2014) Single-molecule measurements of the CCR5 mRNA unfolding pathways. *Biophys J* 106(1):244–252. <https://doi.org/10.1016/j.bpj.2013.09.036>
 22. Kames J, Holcomb DD, Kimchi O, DiCuccio M, Hamasaki-Katagiri N, Wang T, Komar AA, Alexaki A, Kimchi-Sarfaty C (2020) Sequence analysis of SARS-CoV-2 genome reveals features important for vaccine design. *Sci Rep* 10(15643)
 23. Madhani HD, Guthrie C (1994) Dynamic RNA-RNA interactions in the spliceosome. *Ann Rev Genet* 28:1–26. ISSN 00664197. <https://doi.org/10.1146/annurev.ge.28.120194.000245>
 24. Paillart JC, Shehu-Xhilaga M, Marquet R, Mak J (2004) Dimerization of retroviral RNA genomes: an inseparable pair. *Nat Rev Microbiol* 2(6):461–472. ISSN 17401526. <https://doi.org/10.1038/nrmicro903>
 25. Paillart J-C, Skripkin E, Ehresmann B, Ehresmann C, Marquet R (1996) A loop-loop “kissing” complex is the essential part of the dimer linkage of genomic HIV-1 RNA. *Proc Nat Acad Sci USA* 93(May):5572–5577.
 26. Kolb FA, Engdahl HM, Slaughter-Jäger JG, Ehresmann B, Ehresmann C, Westhof E, Wagner EGH, Romby P (2000) Progression of a loop-loop complex to a four-way junction is crucial for the activity of a regulatory antisense RNA. *EMBO J* 19(21):5905–5915. ISSN 02614189. <https://doi.org/10.1093/emboj/19.21.5905>
 27. Kolb FA, Westhof E, Ehresmann B, Ehresmann C, Wagner EGH, Romby P (2001) Four-way junctions in antisense RNA-mRNA complexes involved in plasmid replication control: a common theme? *J Mol Biol* 309(3):605–614. ISSN 00222836. <https://doi.org/10.1006/jmbi.2001.4677>
 28. Bouchard P, Legault P (2014) A remarkably stable kissing-loop interaction defines substrate recognition by the Neurospora Varkud Satellite ribozyme. *RNA* 20(9):1451–1464. ISSN 14699001. <https://doi.org/10.1261/rna.046144.114>
 29. Zuker M (2003) Mfold web server for nucleic acid folding and hybridization prediction. *Nucl Acids Res* 31(13):3406–3415. ISSN 03051048. <https://doi.org/10.1093/nar/gkg595>
 30. Hofacker IL, Fontana W, Stadler PF, Bonhoeffer LS, Tacker M, Schuster P (1994) Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie* 125(2): 167–188. ISSN 00269247. <https://doi.org/10.1007/BF00818163>
 31. Lyngsø RB, Pedersen CNS (2000) RNA pseudoknot prediction in energy-based models. *J Comput Biol* 7(3–4):409–427. ISSN 1066-5277. <https://doi.org/10.1089/106652700750050862>
 32. Rivas E, Eddy SR (1999) A dynamic programming algorithm for RNA structure prediction

- including pseudoknots. *J Mol Biol* 285(5): 2053–2068. ISSN 0022-2836. <https://doi.org/10.1006/jmbi.1998.2436>
33. Uemura Y, Hasegawa A, Kobayashi S, Yokomori T (1999) Tree adjoining grammars for RNA structure prediction. *Theor Comput Sci* 210(2):277–303. ISSN 03043975. [https://doi.org/10.1016/S0304-3975\(98\)00090-5](https://doi.org/10.1016/S0304-3975(98)00090-5)
 34. Akutsu T (2000) Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Appl Math* 104(1–3):45–62. ISSN 0166218X. [https://doi.org/10.1016/S0166-218X\(00\)00186-4](https://doi.org/10.1016/S0166-218X(00)00186-4)
 35. Condon A, Davy B, Rastegari B, Zhao S, Tarrant F (2004) Classifying RNA pseudoknotted structures. *Theor Comput Sci* 320(1):35–50. ISSN 03043975. <https://doi.org/10.1016/j.tcs.2004.03.042>
 36. Dirks RM, Pierce NA (2003) A partition function algorithm for nucleic acid secondary structure including pseudoknots. *J Comput Chem* 24(13):1664–1677. ISSN 01928651. <https://doi.org/10.1002/jcc.10296>
 37. Reeder J, Giegerich R (2004) Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC Bioinform* 5:1–12. ISSN 14712105. <https://doi.org/10.1186/1471-2105-5-104>
 38. Cao S, Chen SJ (2006) Predicting RNA pseudoknot folding thermodynamics. *Nucl Acids Res* 34(9):2634–2652. ISSN 03051048. <https://doi.org/10.1093/nar/gkl346>
 39. Cao S, Chen S-J (2009) Predicting structures and stabilities for H-type pseudoknots with interhelix loops. *RNA* 15(4):696–706. ISSN 1355-8382. <https://doi.org/10.1261/rna.1429009>
 40. Isambert H, Siggia ED (2000) Modeling RNA folding paths with pseudoknots: application to hepatitis delta virus ribozyme. *Proc Nat Acad Sci* 97(12):6515–6520. ISSN 0027-8424. <https://doi.org/10.1073/pnas.110533697>
 41. Ruan J, Stormo GD, Zhang W (2004) An Iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics* 20(1):58–66. ISSN 13674803. <https://doi.org/10.1093/bioinformatics/btg373>
 42. Ren J, Rastegari B, Condon A, Hoos HH (2005) HotKnots : heuristic prediction of RNA secondary structures including pseudoknots HotKnots: heuristic prediction of RNA secondary structures including pseudoknots. *RNA* 11(1):1494–1504. <https://doi.org/10.1261/rna.7284905.knots>
 43. Bellaousov S, Mathews DH (2010) ProbKnot: fast prediction of RNA secondary structure including pseudoknots. *RNA* 16(10): 1870–1880. ISSN 1355-8382. <https://doi.org/10.1261/rna.2125310>
 44. Sato K, Kato Y, Hamada M, Akutsu T, Asai K (2011) IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming. *Bioinformatics* 27(13):85–93. ISSN 13674803. <https://doi.org/10.1093/bioinformatics/btr215>
 45. Jabbari H, Condon A, Zhao S (2008) Novel and efficient RNA secondary structure prediction using hierarchical folding. *J Comput Biol* 15(2):139–163. ISSN 1066-5277. <https://doi.org/10.1089/cmb.2007.0198>
 46. Sperschneider J, Datta A, Wise MJ (2011) Heuristic RNA pseudoknot prediction including intramolecular kissing hairpins. *RNA* 17(1):27–38. ISSN 13558382. <https://doi.org/10.1261/rna.2394511>
 47. Bon M, Micheletti C, Orland H (2013) McGenus: a Monte Carlo algorithm to predict RNA secondary structures with pseudoknots. *Nucl Acids Res* 41(3):1895–1900. <https://doi.org/10.1093/nar/gks1204>
 48. Aalberts DP, Hodas NO (2005) Asymmetry in RNA pseudoknots: observation and theory. *Nucl Acids Res* 33(7):2210–2214. ISSN 03051048. <https://doi.org/10.1093/nar/gki508>
 49. Lucas A, Dill KA (2003) Statistical mechanics of pseudoknot polymers. *J Chem Phys* 119(4): 2414–2421. ISSN 00219606. <https://doi.org/10.1063/1.1587129>
 50. Xayaphoummine A, Bucher T, Isambert H (2005) Kinofold web server for RNA/DNA folding path and structure prediction including pseudoknots and knots. *Nucl Acids Res* 33 (Suppl. 2):605–610. ISSN 03051048. <https://doi.org/10.1093/nar/gki447>
 51. Kucharik M, Hofacker IL, Stadler PF, Qin J (2015) Pseudoknots in RNA folding landscapes. *Bioinformatics* 32(2):187–194. ISSN 14602059. <https://doi.org/10.1093/bioinformatics/btv572>
 52. Kimchi O, Cragolini T, Brenner MP, Colwell LJ (2019) A polymer physics framework for the entropy of arbitrary pseudoknots. *Biophys J* 117(3):520–532. ISSN 15420086. <https://doi.org/10.1016/j.bpj.2019.06.037>
 53. Xia T, SantaLucia J, Burkard ME, Kierzek R, Schroeder SJ, Jiao X, Cox C, Turner DH (1998) Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson–Crick base pairs. *Biochemistry* 37(42):14719–14735.

- ISSN 00062960. <https://doi.org/10.1021/bi9809425>
54. Manyanga F, Horne MT, Brewood GP, Fish DJ, Dickman R, Benight AS (2009) Origins of the “Nucleation” free energy in the hybridization thermodynamics of short duplex DNA. *J Phys Chem B* 113(9):2556–2563. ISSN 15206106. <https://doi.org/10.1021/jp809541m>
 55. Nakano SI, Fujimoto M, Hara H, Sugimoto N (1999) Nucleic acid duplex stability: influence of base composition on cation effects. *Nucl Acids Res* 27(14):2957–2965. ISSN 03051048. <https://doi.org/10.1093/nar/27.14.2957>
 56. Sugimoto N, Nakano S-I, Katoh M, Matsumura A, Nakamuta H, Ohmichi T, Yoneyama M, Sasaki M (1995) Thermodynamic parameters to predict stability of RNA/DNA hybrid duplexes. *Biochemistry* 34(35):11211–11216. ISSN 15204995. <https://doi.org/10.1021/bi00035a029>
 57. Sugimoto N, Nakano SI, Yoneyama M, Honda KI (1996) Improved thermodynamic parameters and helix initiation factor to predict stability of DNA duplexes. *Nucl Acids Res* 24(22):4501–4505. ISSN 03051048. <https://doi.org/10.1093/nar/24.22.4501>
 58. SantaLucia J, Allawi HT, Seneviratne PA (1996) Improved nearest-neighbor parameters for predicting DNA duplex stability. *Biochemistry* 35(11):3555–3562. ISSN 00062960. <https://doi.org/10.1021/bi951907q>
 59. Aalberts DP, Parman JM, Goddard NL (2003) Single-strand stacking free energy from DNA beacon kinetics. *Biophys J* 84(5):3212–3217. ISSN 00063495. [https://doi.org/10.1016/S0006-3495\(03\)70045-9](https://doi.org/10.1016/S0006-3495(03)70045-9)
 60. Srinivas N, Ouldridge TE, Šulc P, Schaeffer JM, Yurke B, Louis AA, Doye JP, Winfree E (2013) On the biophysics and kinetics of toehold-mediated DNA strand displacement. *Nucl Acids Res* 41(22):10641–10658. ISSN 03051048. <https://doi.org/10.1093/nar/gkt801>
 61. Xia T, Mathews DH, Turner DH (2001) Thermodynamics of RNA secondary structure formation. In: Soll D, Nishimura S, Moore PB (eds) *RNA*, chapter 2. Pergamon, 1st edn. pp 21–48 [https://doi.org/10.1002/\(sici\)1097-0282\(1997\)44:3%3C309::aid-bip8%3E3.0.co;2-z](https://doi.org/10.1002/(sici)1097-0282(1997)44:3%3C309::aid-bip8%3E3.0.co;2-z)
 62. Andronescu M, Zhang ZC, Condon A (2005) Secondary structure prediction of interacting RNA molecules. *J Mol Biol* 345(5):987–1001. ISSN 00222836. <https://doi.org/10.1016/j.jmb.2004.10.082>
 63. Turner DH, Mathews DH (2009) NNDB: The nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucl Acids Res* 38(Suppl.1):2009–2011. ISSN 03051048. <https://doi.org/10.1093/nar/gkp892>
 64. SantaLucia J, Hicks D (2004) The thermodynamics of DNA structural motifs. *Ann Rev Biophys Biomol Struct* 33(1):415–440. ISSN 1056-8700. <https://doi.org/10.1146/annurev.biophys.32.110601.141800>
 65. Turner DH (2000) Conformational changes. In: Bloomfield VA, Crothers DM, Tinoco I (eds) *Nucleic acids: structures, properties, and functions*, chapter 8. University Science Books, Sausalito, pp 271–291. ISBN 0935702490
 66. Dirks RM, Bois JS, Schaeffer JM, Winfree E, Pierce NA (2007) Thermodynamic analysis of interacting nucleic acid strands. *SIAM Rev* 49(1):65–88. ISSN 00361445. <https://doi.org/10.1137/060651100>
 67. Serra MJ, Turner DH (1995) Predicting thermodynamic properties of RNA. *Methods Enzymol* 259:242–261.
 68. Mathews DH, Sabina J, Zuker M, Turner DH (1999) Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J Mol Biol* 288:911–940
 69. Allawi HT, SantaLucia J (1997) Thermodynamics and NMR of internal G·T mismatches in DNA. *Biochemistry* 36(34):10581–10594. ISSN 00062960. <https://doi.org/10.1021/bi962590c>
 70. Allawi HT, SantaLucia J (1998) Nearest-neighbor thermodynamics of internal A·C mismatches in DNA: sequence dependence and pH effects. *Biochemistry* 37(26):9435–9444. ISSN 00062960. <https://doi.org/10.1021/bi9803729>
 71. Allawi HT, SantaLucia J (1998) Nearest neighbor thermodynamic parameters for internal G·A mismatches in DNA. *Biochemistry* 37(8):2170–2179. ISSN 00062960. <https://doi.org/10.1021/bi9724873>
 72. Allawi HT, SantaLucia J (1998) Thermodynamics of internal C·T mismatches in DNA. *Nucl Acids Res* 26(11):2694–2701. ISSN 03051048. <https://doi.org/10.1093/nar/26.11.2694>
 73. Peyret N, Seneviratne PA, Allawi HT, SantaLucia J (1999) Nearest-neighbor thermodynamics and NMR of DNA sequences with internal A·A, C·C, G·G, and T·T mismatches. *Biochemistry* 38(12):3468–3477. ISSN

00062960. <https://doi.org/10.1021/bi9825091>
74. Watkins NE, Kennelly WJ, Tsay MJ, Tuin A, Swenson L, Lee HR, Morosyuk S, Hicks DA, SantaLucia J (2011) Thermodynamic contributions of single internal rA·dA, rC·dC, rG·dG and rU·dT mismatches in RNA/DNA duplexes. *Nucl Acids Res* 39(5):1894–1902. ISSN 03051048. <https://doi.org/10.1093/nar/gkq905>
75. Bommarito S, Peyret N, SantaLucia J (2000) Thermodynamic parameters for DNA sequences with dangling ends. *Nucl Acids Res* 28(9):1929–1934. ISSN 03051048. <https://doi.org/10.1093/nar/28.9.1929>
76. Lu ZJ, Turner DH, Mathews DH (2006) A set of nearest neighbor parameters for predicting the enthalpy change of RNA secondary structure formation. *Nucl Acids Res* 34(17):4912–4924. ISSN 03051048. <https://doi.org/10.1093/nar/gkl472>
77. Xu ZZ, Mathews DH (2016) Secondary structure prediction of single sequences using RNAstructure. In: Turner DH, Mathews DH (eds) *Methods in molecular biology. RNA structure determination*, vol 1490, chapter 2. Humana Press, New York, pp 15–35. ISBN 978-1-4939-6431-4. <https://doi.org/10.1007/978-1-4939-6433-8>
78. Jacobson DR, McIntosh DB, Saleh OA (2013) The snakelike chain character of unstructured RNA. *Biophys J* 105(11):2569–2576. ISSN 00063495. <https://doi.org/10.1016/j.bpj.2013.10.019>
79. Chen H, Meisburger SP, Pabit SA, Sutton JL, Webb WW, Pollack L (2012) Ionic strength-dependent persistence lengths of single-stranded RNA and DNA. *Proc Nat Acad Sci USA* 109(3):799–804. ISSN 00278424. <https://doi.org/10.1073/pnas.1119057109>



Metrics for RNA Secondary Structure Comparison

Feiqi Wang, Tatsuya Akutsu, and Tomoya Mori

Abstract

RNA secondary structure comparison is one of the important analyses for elucidating individual functions of RNAs since it is widely accepted that their functions and structures are strongly correlated. However, although the RNA secondary structures with pseudoknot play important roles *in vivo*, it is difficult to deal with such structures *in silico* due to their structural complexity, which is a major obstacle to the analysis of RNA functions.

Here, we introduce an algorithm and a metric for comparing pseudoknotted RNA secondary structures based on topological centroid identification and tree edit distance and describe the usage protocol of a software enabling us to run the comparison. This software is publicly available and works on both Microsoft Windows and Apple macOS.

Key words RNA secondary structure, Pseudoknot, Topological centroid, Tree edit distance

1 Introduction

RNAs are essential biomolecules that transmit genetic information from DNAs to proteins. It is well known that some transcripts, such as ribosomal RNAs and transfer RNAs, work without being translated into proteins, but in recent years it has been revealed that there are many functional noncoding RNAs that play important roles represented by regulation of gene expression, and many studies have been conducted to clarify their functions. Among them, in the field of bioinformatics, the approaches of predicting the functions by comparing RNA structures have been widely adopted based on an observation that there is a strong correlation between their structures and functions, i.e., the functions are similar if their structures are similar. However, it is already known that analysis of RNA three-dimensional structures is a very difficult problem in terms of both biological experiments and computational experiments. Therefore, RNA secondary structure is widely used as a mathematical representation that is suitable for computational processing and structural analysis. For example, Vienna RNA is one of

the most widely used packages for RNA secondary structure prediction and comparison [1].

However, most of the existing methods for RNA secondary structure analysis do not support RNAs having pseudoknot structures, which is a specific folding motif in RNA secondary structure. It has been suggested that the pseudoknotted RNAs play important roles in the living body, but unlike the usual base pairing pattern, pseudoknots are not nested, which makes it extremely difficult to handle the pseudoknotted RNAs *in silico*. In fact, it has already been shown that both prediction and alignment of pseudoknotted RNA secondary structures are nondeterministic polynomial-time hard (NP-hard) problems [2–4]. Therefore, the pseudoknot structures have often been ignored or converted to pseudoknot-free structures for efficient computation [5–7]. On the other hand, algorithms for predicting pseudoknotted RNA secondary structures based on integer linear programming have also been proposed, and it is reported that the algorithm works in a practical amount of time [8, 9].

Similarly, it is also difficult to deal with pseudoknotted RNAs even in secondary structure comparison. This is because a general approach is to transform them into tree structures when comparing, but the structural complexity of pseudoknot makes it hard. Actually, many RNA comparison algorithms are limited to structures without pseudoknots [10, 11] and there are few methods that can handle secondary structures with arbitrary pseudoknot structures [12]. In contrast, Möhl et al. developed a fixed parameter tractable algorithm using a general edit distance of arbitrary pseudoknotted RNA structures for reasonable scoring schemes, but its time complexity of the worst case is exponential [13]. PSMAlign was also developed for comparing pseudoknotted structures [12]. The algorithm was based on a local alignment approach and achieved efficient alignment of pseudoknotted structures by identifying similar stem structures, but its worst-case time complexity is also exponential. Thus, the structural complexity of pseudoknots has been a major obstacle to RNA secondary structure research.

In this article, we describe a newly developed practical algorithm for comparing RNA secondary structures with pseudoknots. The algorithm is based on topological centroid identification of input plane graphs representing RNA secondary structures by **PEELING** algorithm [14] and tree edit distance, where the latter is one of the most widely used metrics for comparing tree-structured data [15]. Since the details of the algorithm have already been published in [16], the explanation of the algorithm will be limited to the outline in this article, but instead here we describe a small extension of the proposed method and the usage of the tool implementing our algorithm published in GitHub (<https://github.com/feiqiwang/planeGraph2tree>).

2 Materials

We provide a tool named *planeGraph2tree*, which transforms an input plane graph into a topological centroid tree and compares them based on the tree edit distance. The tool can be downloaded from GitHub (<https://github.com/feiqiwang/planeGraph2tree>). The project files which are executable in Visual Studio (<https://visualstudio.microsoft.com>) and Apple Xcode (<https://apps.apple.com/us/app/xcode/id497799835?mt=12>) are contained in the repository, so it is necessary to download and install Microsoft Visual Studio or Xcode beforehand which are suitable for user's computing environment (see **Notes 1** and **2**).

In *planeGraph2tree*, an input RNA is transformed to multiple rooted ordered labeled trees formatted as the bracket notation, and they are saved as text files. The tree edit distance between the resulting trees is computed by RTED, which is a publicly available practical software for computing the ordered tree edit distance (<http://tree-edit-distance.dbresearch.uni-salzburg.at>) [17]. Although RTED is automatically downloaded from its official web page by a UNIX/Linux command *wget* (<http://www.gnu.org/software/wget/manual/wget.html>) in a script of *planeGraph2tree*, Java SE Development Kit (<https://www.oracle.com/technetwork/java/javase/overview/index.html>) is also required to run RTED.

When the tree edit distances between all input RNAs are obtained, a distance matrix is generated from it, and then hierarchical clustering of the RNAs is performed, where the hierarchical clustering requires installation of R (<https://www.r-project.org>). If users want to run this clustering in Windows OS, Git bush (<https://www.git-scm.com/download/win>) should be downloaded and installed for enabling users to enter the execution commands via character user interface (CUI).

Finally, in this article, we use pseudoknotted RNAs downloaded from PseudoBase++ (<http://pseudobaseplusplus.utep.edu/home>) [18] as BPSEQ formatted input data. The BPSEQ format is a simple format for representing structural information in three columns: (1) the index ordered from 5' to 3' for an RNA; (2) the label as one-letter notation, which is either A, U, C, G, X, or Y, where X and Y denote the nucleotide would by any base and C or U, respectively; and (3) the index of pairing partner for the nucleotide, where it is set 0 if the nucleotide is unpaired (Fig. 1).

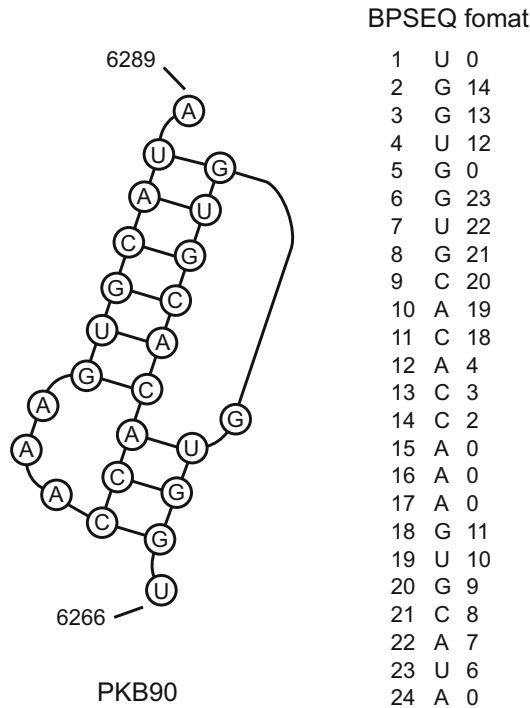


Fig. 1 Example of RNA secondary structure and its BPSEQ format (PKB90). The numbers of the endpoints of sequences indicates the positions of both ends of the bases. The BPSEQ format is a simple format for representing structure information described by the following three columns: the index ordered from 5' to 3' for an RNA sequence, the label with acronym of nucleotide, and the index of paired partner for each nucleotide

3 Methods

3.1 Topological Centroid Identification of Plane Graph

An RNA secondary structure can be mostly represented by a plane graph even when it has pseudoknot structure. As mentioned above, *planeGraph2tree* first identifies the topological centroid of an input plane graph and transforms it into a tree by **PEELING** algorithm [14].

The **PEELING** algorithm repeats the operations of searching for and deleting certain groups of edges, faces, and subgraphs called *singly exposed edges*, *singly exposed faces*, and *adjunct subgraphs* that have special properties explained later. Here, when planar embedding (i.e., edges do not have any common points other than the endpoints) is given for an undirected graph $G(V, E)$, where V is a finite set of vertices and E is a finite set of edges, G is called a plane graph. The regions inside and outside of G are called the inner faces and outer faces, respectively. Let C be the directed cycle composed of the edges of the outer face, where the edges are visited in clockwise order. If an edge is not belonging to an inner face and

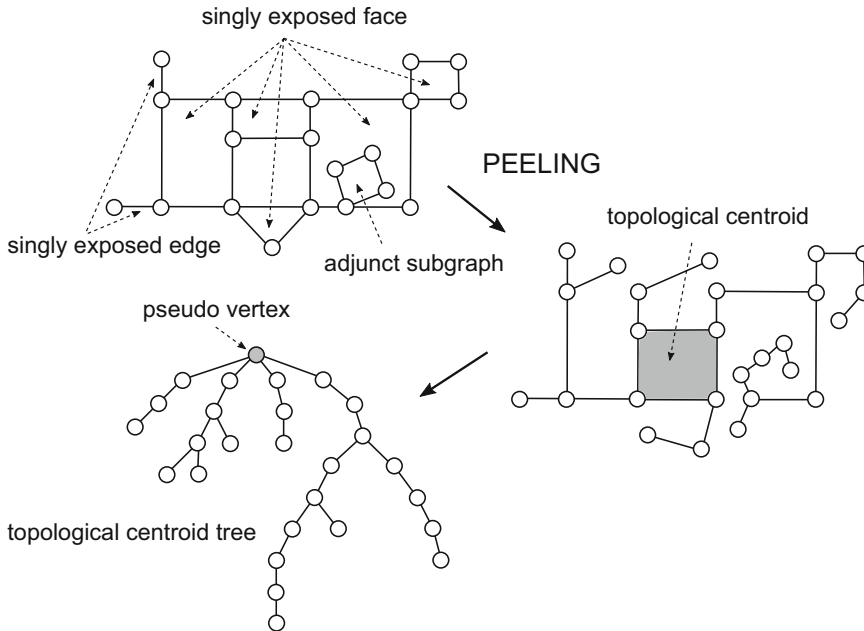


Fig. 2 Illustration of PEELING process. First, the **PEELING** algorithm identifies singly exposed edges and singly exposed faces of an input plane graph and then deletes outer edges and adjunct subgraphs in these exposed edges and faces. After repeating these processes, the **PEELING** algorithm finds the topological centroid of the input plane graph, and then a topological centroid tree is generated, where a pseudo vertex is added if the topological centroid is either an edge or a face

the outdegree of either endpoint is 1, the edge is called *singly exposed*. Similarly, an inner face of G is called *singly exposed* when its outer edges in C are connected ignoring direction of edges. Furthermore, each maximally connected subgraph that shares only one outer vertex and is surrounded by a singly exposed face is called an *adjunct subgraph*.

In order to identify the topological centroid, **PEELING** repeats the following procedures: (1) finds all singly exposed edges and singly exposed faces and then (2) deletes the outer edges and adjunct subgraphs in these edges and faces until singly exposed edges and singly exposed faces no longer exist. As a result, the topological centroid is either a vertex, an edge, or a face. After identifying the topological centroid, a tree is constructed by adding the deleted edges, faces, and adjunct subgraphs with decomposing cycles [14]. Hereafter, the tree containing the topological centroid obtained by **PEELING** is called a topological centroid tree (Fig. 2).

3.2 Comparison of Topological Centroid Trees by Tree Edit Distance

The topological centroid trees transformed from RNA secondary structures are compared by tree edit distance, which is one of the most widely used metrics for comparing tree-structured data [15]. The tree edit distance is defined by the cost of the minimum cost sequence of edit operations for transforming a tree into

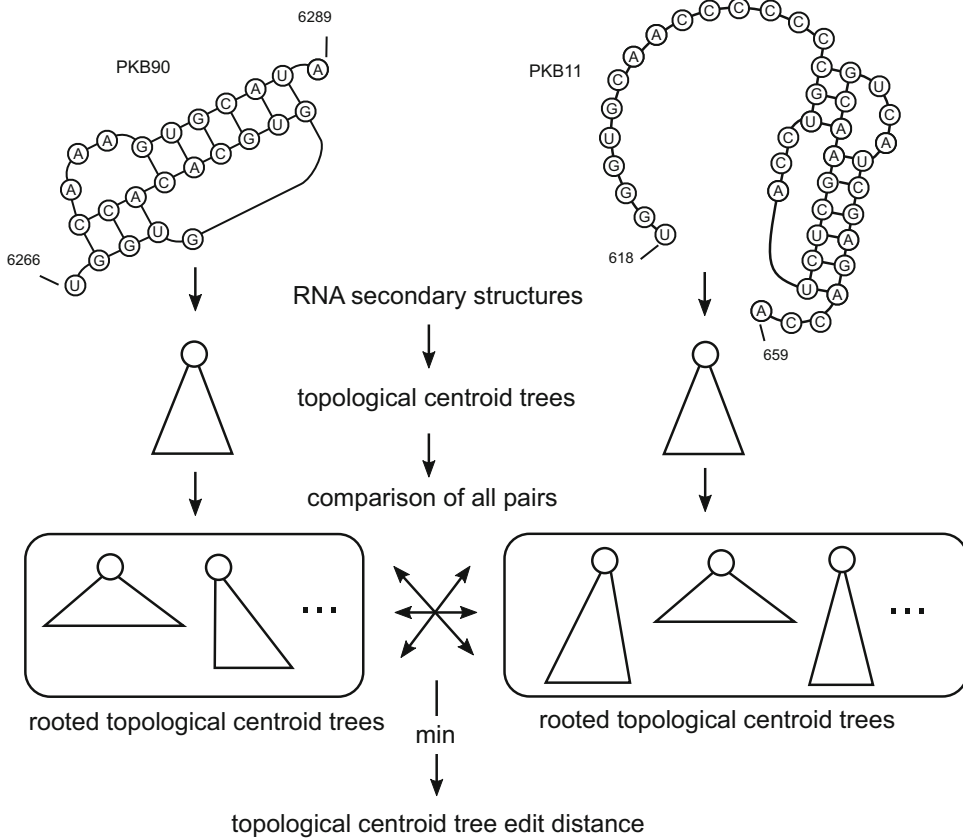


Fig. 3 Computation of topological centroid tree edit distance. When computing the topological centroid tree edit distance, first input RNA secondary structures are converted to rooted topological centroid trees by specifying one of the nodes of their topological centroid trees as the roots for each node. After that, compute the tree edit distance between all pairs of the rooted topological centroid trees. Finally, the minimum value of the tree edit distances among the all pairs is output as the topological centroid tree edit distance between the two input trees

another tree, where an edit operation is either a deletion of a non-root node, an insertion of nonroot node, and a substitution of the label of a node. The cost of each edit operation is 1 under the unit cost model. Here, the root is not specified in a topological centroid tree obtained by PEELING. Therefore, we calculate the tree edit distance for all cases when each node of the input trees is designated as the root and newly define the minimum distance of the tree edit distances as the topological centroid tree edit distance (Fig. 3).

3.3 Protocol: Run Topological Centroid Identification and Topological Centroid Tree Generation

The *planeGraph2tree* can be freely downloaded from the webpage at GitHub (<https://github.com/feiqiwang/planeGraph2tree>) by clicking the download button “Code” or cloning the repository. The software can work on both Microsoft Windows and Apple macOS (see Note 3). For Windows users, first open a solution file *CanonicalFormComputing/CanonicalFormComputing.sln* on

Microsoft Visual Studio and set an input file name to the variable “fileName” in the main file *CanonicalFormComputing.cpp* (i.e., string fileName = “example”). If multiple RNA secondary structures need to be transformed to trees, it is necessary to change the file name and execute this program one by one. In order to run the program, select “Build Solution” from “Build Solution” tab, and then click “Start Without Debugging” in “Debug” tab. After finishing the computation, the input BPSEQ formatted data is transformed to doubly connected edge list (DCEL) formatted data and output to *DCEL* directory as intermediate files, where the DCEL format is an edge-based data structure for easily manipulating and traversing a plane graph [14]. Finally, three output directories containing topological centroid trees are generated: *tree_result_inPseudoVer*, *tree_result_MultiRoot*, and *tree_result_allVerAsRoot*. The first two directories contain previous versions of rooted topological centroid trees described in [16], i.e., only topological centroid is set as the root or the nodes connected to the topological centroid are set as the root, respectively. Thus, the main output is rooted topological centroid trees saved in the directory *tree_result_allVerAsRoot*. In *planeGraph2tree*, only so-called bracket notation is supported as the representation of trees.

3.4 Protocol: Run Topological Centroid Tree Comparison

The *planeGraph2tree* provides an R script *clusteringComputation.r* and a shell script *comparisonProgram.sh* to compare topological centroid trees. For Windows users, downloading and installing Git bash are needed to run the program via CUI. First, copy the output directory *tree_result_allVerAsRoot* to the directory *ComparisonProgram*, and then open *comparisonProgram.sh* and specify *tree_result_allVerAsRoot* as an input directory by modifying a file path and the parameters such as the cost of each edit operation (each cost is set to 1 by default) in the script. Then, change the access permission of *comparisonProgram.sh* by the command “chmod 755 comparisonProgram.sh,” and finally run the script by entering “./comparisonProgram.sh” via CUI (see Notes 4 and 5).

3.5 Application Example

Here, we apply *planeGraph2tree* to real pseudoknotted RNA secondary structures downloaded from PseudoBase++. The BPSEQ format for each RNA secondary structures can be obtained by entering PKB number (e.g., PKB100) in the “Quick Search” field of the top page, checking “Download” box the next to the objective PKB number, and clicking “Generate output files” at the bottom of the search results screen to display the page containing the BPSEQ formatted data download link. As an example, we obtained ten pseudoknotted RNA secondary structures – PKB6, PKB7, PKB8, PKB10, PKB11, PKB22, PKB24, PKB90, PKB91, and PKB100 – and compared them by *planeGraph2tree*. The result of the comparison is shown in Fig. 4, where the topological centroid tree edit distance was computed under the unit cost model.

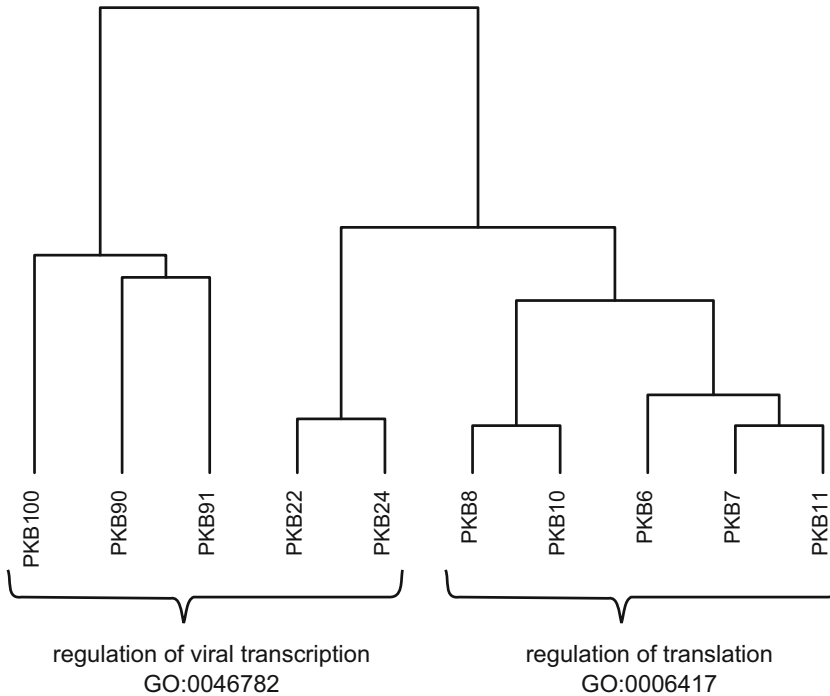


Fig. 4 Hierarchical clustering results of ten pseudoknotted RNA secondary structures based on the topological centroid tree edit distance. The pseudoknotted structures PKB22, PKB24, PKB90, PKB91, and PKB100 are related to *regulation of viral transcription* (GO:0046782), whereas PKB6, PKB7, PKB8, PKB10, and PKB11 are related to *regulation of translation* (GO:0006417). As shown in the result, the RNAs were classified well according to their related functions

The hierarchical clustering result was generated by *hclust* function with *ward's* method on R. According to the cross-references among PseudoBase++, European Nucleotide Archive (ENA) (<https://www.ebi.ac.uk/ena>), Rfam (<https://rfam.xfam.org>) [19], and Gene Ontology (GO) (<http://geneontology.org>) [20, 21], PKB22, PKB24, PKB90, PKB91, and PKB100 are related to *regulation of viral transcription* (GO:0046782), whereas PKB6, PKB7, PKB8, PKB10, and PKB11 are related to *regulation of translation* (GO:0006417). As shown in the clustering result, the RNAs were classified well according to their related functions.

Although it has already been described in [16] that *plane-Graph2tree* shows a better clustering result than the simple comparison of their sequences based on the string edit distance, it is recommended to use it after fully understanding the following points. First, a complete comparison between pseudoknotted RNA secondary structures is not achieved by the method since the secondary structural information is degenerated when transforming a plane graph to a tree. For example, when focusing on a certain branch of a topological centroid tree generated by **PEELING**, it cannot be determined whether it was originally a singly exposed edge or an edge generated by **PEELING** from the original

singly exposed face. Furthermore, when there are singly exposed edges and singly exposed faces with the same label sequence, it is difficult to distinguish between them since they become identical after **PEELING** in some cases. Second, when computing the topological centroid tree edit distance, the trees are considered as ordered trees. Indeed, the RNA secondary structures often transformed as ordered trees, but it is more appropriate to consider the topological centroid trees as unordered trees since there is no order relationship between vertices and branches disconnected by **PEELING**. However, the tree edit distance problem between unordered trees is NP-hard [22], that is, there is no effective algorithm which can compute it in polynomial time (unless $P = NP$). Therefore, *planeGraph2tree* treats the input trees as ordered trees to compute the topological centroid tree edit distance efficiently sacrificing some accuracy.

4 Notes

1. If a compiler of C++ has not been installed, it is necessary to download and install it from the GNU Compiler Collection (<https://gcc.gnu.org>). As for Mac users, a C++ compiler can be used after installing Xcode and Command Line Tools for Xcode from App Store application preinstalled to Mac machines. Alternatively, it can be installed via Homebrew (<https://brew.sh/index>).
2. If users have an older version of macOS, Xcode may not be able to download and install from App Store. In such a case, upgrade the macOS or download and install suitable versions of Xcode for users' environments directory from Apple Developer site (<https://developer.apple.com/develop/>).
3. As for Mac users, most of the procedures are the same as Windows users' procedures. First open the project file *CanonicalFormComputing_mac.xcodeproj* in *CanocalFormComputing* directory on Xcode instead of the solution file. Then, choose "Edit scheme" from the "Product" menu and select "CanonicalFormComputing_mac/ CanonicalFormComputing_mac" as the user's working directory. After that, run the program according to the user's Xcode environment (often by clicking the triangle icon in the top of the window).
4. To run *comparisonProgram.sh*, *wget* command, JDK, and Rscript should be installed and their PATHs (i.e., environmental variables) should be added to the user's system. For Windows users, PATH can be added from "System" in "Control Panel," whereas Mac users do not have to do anything for PATH setting in general, but if it does not work properly, set PATH by editing *.bash_profile* file at HOME directory.

5. Input BPSEQ files may not be recognized depending on the user's computer environment. In that case, make sure that the BPSEQ format is written correctly, and then check the newline character is compatible with user's system environment.

References

1. Lorenz R, Bernhart SH, Höner zu Siederdissen C et al (2011) Vienna RNA package 2.0. *Algorithms Mol Biol* 6(26):14
2. Akutsu T (2000) Dynamic programming algorithms for RNA secondary structure predictions with pseudoknots. *Discrete Appl Math* 104:45–62
3. Jiang T, Lin G, Ma B et al (2002) A general edit distance between RNA structures. *J Comput Biol* 9:371–388
4. Blin G, Denise A, Dulucq S et al (2010) Alignments of RNA structures. *IEEE/ACM Trans Comput Biol Bioinform* 7:309–322
5. Smit S, Yarus M, Knight R (2006) Natural selection is not required to explain universal compositional patterns in rRNA secondary structure categories. *RNA* 12:1–14
6. Andronescu M, Condon A, Hoos HH et al (2007) Efficient parameter estimation for RNA secondary structure prediction. *Bioinformatics* 23:i19–i28
7. Smit S, Rother K, Heringa J et al (2008) From knotted to nested RNA structures: a variety of computational methods for pseudoknot removal. *RNA* 14:410–416
8. Poolsap U, Kato Y, Akutsu T (2009) Prediction of RNA secondary structure with pseudoknots using integer linear programming. *BMC Bioinformatics* 10(Suppl 1):S38
9. Sato K, Kato Y, Hamada M, Akutsu T (2011) IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming. *Bioinformatics* 27:i85–i93
10. Hochsmann M, Voss B, Giegerich R (2004) Pure multiple RNA secondary structure alignments: a progressive profile approach. *IEEE/ACM Trans Comput Biol Bioinform* 1:53–62
11. Evans PA (2006) Finding common RNA pseudoknot structures in polynomial time. In: Lewenstein M, Valiente G (eds) *Combinatorial pattern matching. CPM2006. Lecture notes in computer science, vol 4009*. Springer, Berlin/Heidelberg
12. Chiu JKH, Chen YPP (2015) Pairwise RNA secondary structure alignment with conserved stem pattern. *Bioinformatics* 31:3914–3921
13. Möhl M, Will S, Backofen R (2008) Fixed parameter tractable alignment of RNA structures including arbitrary pseudoknots. In: Ferragina P, Landau GM (eds) *Combinatorial pattern matching. CPM2006. Lecture notes in computer science, vol 5029*. Springer, Berlin/Heidelberg
14. Akutsu T, de la Higuera C, Tamura T (2018) A simple linear-time algorithm for computing the centroid and canonical form of a plane graph and its applications. In: Navarro G, Sankoff D, Zhu B (eds) *Proceedings of combinatorial pattern matching. CPM2018, 105*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl
15. Bille P (2005) A survey on tree edit distance and related problems. *Theor Comput Sci* 337: 217–239
16. Wang F, Akutsu T, Mori T (2020) Comparison of pseudoknotted RNA secondary structures by topological centroid identification and tree edit distance. *J Comput Biol* 27(9): 1443–1451. <https://doi.org/10.1089/cmb.2019.0512>
17. Pawlik M, Augsten N (2011) RTED: a robust algorithm for the tree edit distance. *Proc VLDB Endow* 5(4):334–345
18. Taufer M, Licon A, Araiza R et al (2008) PseudoBase++: an extension of PseudoBase for easy searching, formatting and visualization of pseudoknots. *Nucleic Acids Res* 37(Suppl 1): D127–D135
19. Ioanna K, Janna A, Natalia QO et al (2017) Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. *Nucleic Acids Res* 46(D1):D335–D342
20. Ashburner M, Ball CA, Blake JA et al (2000) Gene ontology: tool for the unification of biology. *Nat Genet* 25:25–29
21. The Gene Ontology Consortium (2019) The gene ontology resource: 20 years and still going strong. *Nucleic Acids Res* 47(D1): D330–D338
22. Zhang K, Statman R, ShaSha D (1992) On the editing distance between unordered labeled trees. *Inform Process Lett* 42:133–139



RNA Secondary Structure Prediction Based on Energy Models

Manato Akiyama and Kengo Sato

Abstract

This chapter introduces the RNA secondary structure prediction based on the nearest neighbor energy model, which is one of the most popular architectures of modeling RNA secondary structure without pseudoknots. We discuss the parameterization and the parameter determination by experimental and machine learning-based approaches as well as an integrated approach that compensates each other's shortcomings. Then, folding algorithms for the minimum free energy and the maximum expected accuracy using the dynamic programming technique are introduced. Finally, we compare the prediction accuracy of the method described so far with benchmark datasets.

Key words RNA secondary structure prediction, Nearest neighbor model, Thermodynamic parameters, Machine learning, Minimum free energy, Maximum expected accuracy

1 Introduction

RNA secondary structure prediction based on thermodynamics has been studied since the 1980s and is still the most popular approach because it is the most natural way for RNA molecules to form a thermodynamically stable structure from the viewpoint of biophysics. For example, the free energy minimization-based algorithms are widely utilized such as Mfold/UNAFold [1, 2], Vienna RNA package [3, 4], and RNAstructure [5, 6]. Furthermore, since the conformation of RNA secondary structures follows the Boltzmann distribution, the maximum expected accuracy-based algorithms that consider all possible secondary structures have been proposed and implemented in Vienna RNA package [3, 4], CentroidFold [7, 8] and RNAstructure [5, 6].

However, RNA secondary structure prediction based on thermodynamics has several limitations, and thus the accuracy of the prediction is still not sufficient. RNA secondary structure prediction based on thermodynamics utilizes thermodynamic parameters that should be experimentally determined in advance. However,

due to technical difficulties in the experiments, the thermodynamic parameters determined by the experiments are not sufficiently accurate. To achieve more accurate RNA secondary structure prediction, we need more fine-grained parameterization that requires more complicated and labor-intensive experiments.

Therefore, instead of such experiments, machine learning-based approaches have recently been studied, determining weight parameters from training datasets including RNA sequences and their known secondary structures. For example, CONTRAfold [9, 10] and ContextFold [11] trained weight parameters, instead of the thermodynamic parameters, based on the architecture of RNA secondary structures similar to that in the thermodynamic models. In particular, ContextFold employed a much richer parameter set that cannot be measured by experimental methods and achieved state-of-the-art accuracy compared with existing RNA structure prediction methods. However, a rich parameterization could easily cause overfitting to the training data, resulting in disabling robust prediction for a wide range of RNA sequences. Indeed, Rivas [12] has pointed out that ContextFold failed to predict secondary structures for several RNA families not included in training data due to overfitting the training data.

Hybrid methods that combine the thermodynamic and machine learning-based approaches to compensate for each other's shortcomings have been developed. SimFold [13, 14] estimated more accurate thermodynamic parameters from training data including RNA sequences and their known secondary structures as well as their free energy of known secondary structures. MXfold [15] combined the thermodynamic energy parameters and rich-parameterized weight parameters. It learns more precise parameters for substructures observed in training data and avoids overfitting the rich-parameterized weight parameters to the training data by falling back to the thermodynamic parameters for unobserved substructures.

2 Nearest Neighbor Model

This section describes the nearest neighbor model [16–19] that has been widely employed for modeling RNA secondary structures by a number of algorithms for predicting RNA secondary structures without pseudoknots based on energy models. The nearest neighbor model decomposes a secondary structure into loop substructures enclosed by the nearest neighboring base pairs. The decomposed loops can be classified according to the number of enclosing base pairs (Fig. 1). A loop with a single base pair is a hairpin loop. A loop with two base pairs is a stacking, a bulge loop, or an internal loop. A loop with three or more base pairs is called a multibranch loop. A single strand region that is enclosed by no base pair is an external loop.

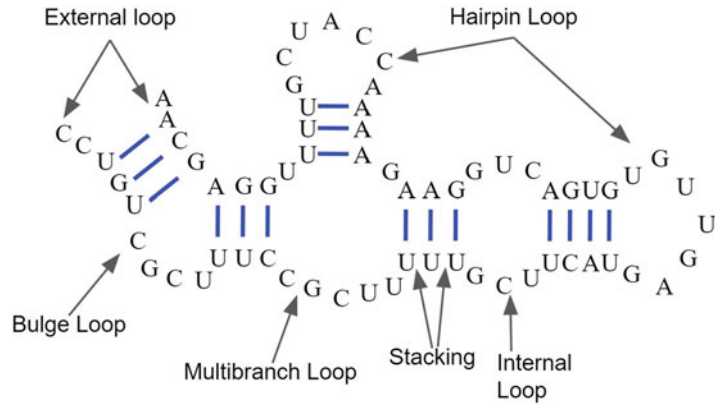


Fig. 1 Loop substructures defined in the nearest neighbor model

The nearest neighbor model is based on two assumptions. The first is that each loop's energy depends only on bases and base pairs that are part of the loop, meaning that the outside of the loop does not affect the loop energy. The second assumption is that the energy of the entire secondary structure is the sum of the energies of its decomposed loops. We will describe several approaches for calculating the energies of the loops in Subheading 3. These assumptions enable us to efficiently calculate the thermodynamically most stable secondary structure from an RNA sequence as described in Subheading 4.

3 Parameterization

As introduced in the previous section, the energy-based RNA secondary structure prediction calculates the energy of RNA secondary structures by summing over the energies of every loop into which the secondary structure is decomposed according to the nearest neighbor model. This section describes three approaches for determining the energies of the individual loops: the thermodynamic approach, the machine learning-based approach, and the integrated approach.

3.1 Thermodynamic Approach

The thermodynamic approach is the most popular approach to find the thermodynamically most stable secondary structure with the minimum free energy (MFE) such as Mfold/UNAFold [1, 2], Vienna RNA package [3, 4], and RNAstructure [5, 6]. The thermodynamic approach first adopted the nearest neighbor model. In the thermodynamic approach, the nearest neighbor loops are parameterized by several types of characteristic components, and then each component is assigned an energy parameter determined by experimental methods such as the optical melting experiment.

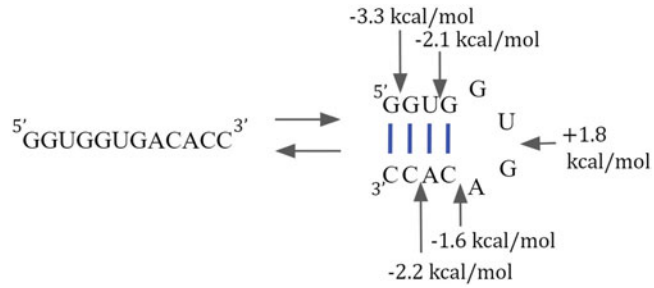


Fig. 2 Free energy change of three base pair stackings (GC–GC, GC–UA, and UA–GC), a terminal mismatch (GC–GA), and a hairpin loop (4 nt). Nearest neighbor parameters are taken from NNDB [18]

The most conventional parameterization of the nearest neighbor model is the Turner free energy parameter set [18, 20, 21], which introduces the following types of characteristic components: Watson-Crick helices, GU pairs, dangling ends, terminal mismatches, hairpin loops, bulge loops, internal loops, coaxial stacking, multibranch loops, and external loops.

Thermodynamic energy change of folding RNA oligomers can be measured by analyzing optical melting curves. Since these data assume that the oligomer strand can only exist in two structures, an unfolding structure and an MFE structure, this model is called the two-state model [22]. We assume that the energy change of the folding RNA oligomer is the sum of the free energy changes of RNA components that parameterize the decomposed loops of the nearest neighbor model. Then the free energy change of each component is derived by the regression analysis fitting to a collection of experimentally determined energy change of oligomers measured by the optical melting experiments [17, 18, 20]. For example, the free energy parameters of stackings, a terminal mismatch, and a hairpin loop are measured from the optical melting curves of the synthesized oligonucleotides, as shown in Fig. 2.

There are several limitations to the thermodynamic approach. The first is the experimental and systematic errors of the optical melting analysis. The energy parameters of the Watson-Crick helices are highly accurate with an error of less than 0.1 kcal/mol [17]. However, the free energy error for other components is about 0.5 kcal/mol, resulting from the errors of the optical melting experiments and the systematic errors from the nearest neighbor assumption [20]. The second limitation arises from the parameterization of the thermodynamic approach. There are not a few sequences that the current parameterization does not work well because they have not been experimentally studied. Therefore, we need more fine-grained parameterization that requires more complicated and labor-intensive experiments.

3.2 Machine Learning-Based Approach

Alternative to the thermodynamic approach whose parameters have been derived from the experimental methods, a machine learning-based approach that trains parameters from known reference structures has been available for the nearest neighbor model. The thermodynamic approach performs scoring by the free energy, while a probabilistic approach and a weight-based approach obtain the parameter values by statistical estimation or machine learning.

3.2.1 Probabilistic Approach

Generative models, including stochastic context-free grammars (SCFGs), have been adopted to model RNA secondary structure by a probabilistic approach. Hidden Markov models (HMMs) can be applied to the generative model of the primary sequence. However, HMMs cannot effectively deal with RNA secondary structure constraints. SCFGs can be applied for modeling an RNA secondary structure as a derivation tree of production rules. Each production has a probability learned from known secondary structure data. The joint probability of an RNA sequence x with a secondary structure y is the probability of the derivation tree that generates x and y calculated by the product of the production probabilities in the derivation tree.

SCFG was first applied for modeling RNA secondary structures by [23, 24]. Recently, TORNADO [25] has implemented the application of SCFG to the nearest neighbor model, which has achieved comparable performance as the weight-based approach described in Subheading 3.2.2.

3.2.2 Weight-Based Approach

One of the primitive weight-based approach for RNA folding is the Nussinov algorithm [26]. The Nussinov algorithm is an algorithm to search for a secondary structure with the maximum number of base pairs using dynamic programming.

While the Nussinov algorithm employs fixed parameters, recent algorithms employ weight parameters trained from known RNA secondary structures. In the weight-based approach, the parameters are real numbers and do not require thermodynamic interpretation. In other words, the thermodynamic approach can be considered to be part of the weight-based approach.

As the weight-based approach, conditional random fields (CRFs), one of the probabilistic graphical models, and conditional log-linear models (CLLMs), a generalization of CRFs, are commonly used to model RNA secondary structure prediction.

A scoring model $f(x, y)$ is a function that assigns real-valued scores to a secondary structure y of a sequence x . CLLMs assume a linear scoring model of RNA secondary structures as follows:

$$f(x, y) = \lambda^\top \Phi(x, y),$$

where Φ is the feature representation vector of (x, y) , each of whose elements is the number of occurrences of the individual component that parameterizes nearest neighbor loops, and λ is the weights for Φ . The conditional probability of a secondary structure y given a sequence x is:

$$P(y|x) = \frac{\exp f(x, y)}{\sum_{y' \in \mathcal{S}(x)} \exp f(x, y')},$$

where $\mathcal{S}(x)$ is a set of all possible secondary structure of the sequence x .

In determining parameter values based on machine learning, experimentally verified secondary structures are used as training data. CLLMs search for a set of the weight parameters that maximizes the conditional probability of N pairs of the known secondary structures $y^{(i)}$ given the sequences $x^{(i)}$:

$$\lambda_{\text{MLE}} = \arg \max_{\lambda} \prod_{i=1}^N P_{\lambda}(y^{(i)} | x^{(i)}).$$

CONTRAFold [9, 10] has implemented the nearest neighbor model based on CLLMs, enabling robust prediction by reducing the number of parameters from the Turner free energy model. Furthermore, since the machine learning-based approach is not limited by the experimental methods, it can employ highly flexible and rich parameterization, which is not employed by the Turner free energy model. For example, ContextFold [11] has implemented fine-grained secondary structure modeling with more than 200,000 parameters, resulting in state-of-the-art prediction accuracy. However, at the same time, it raises potential concerns about overfitting as reported in [12], suggesting that avoidance of overfitting is necessary for fine-grained models to achieve robustness.

3.3 Integrated Approach

In the thermodynamic approach, parameter values are experimentally determined as free energy. The experimental limitations cause unmeasurable parameters and measurement errors. On the other hand, there are several difficulties in determining parameter values by machine learning such as overfitting caused by a large number of parameters for detailed modeling. Furthermore, parameter values that do not appear in the training data cannot be obtained.

To overcome the shortcomings of these approaches, Akiyama et al. have proposed a novel scoring model that integrates the thermodynamic approach and the machine learning-based approach [15], which can be expected to complement each other's shortcomings. While the experimental measurement error is complemented by machine learning, the parameters not observed in the training dataset are complemented by free energy.

The integrated scoring model of a structure y given a sequence x is defined as:

$$\begin{aligned} f(x, y) &= f_T(x, y) + f_W(x, y) \\ f_T(x, y) &= \lambda_T^\top \Phi_T(x, y) \\ f_W(x, y) &= \lambda_W^\top \Phi_W(x, y), \end{aligned}$$

where $f_T(x, y)$ and $f_W(x, y)$ are, respectively, contributions of the thermodynamic model and the machine learning model to the scoring model. Both feature representations Φ_T and Φ_W are based on the nearest neighbor model [27]. The negative values of the Turner free energy parameter set [21] are employed as λ_T . The structured support vector machine (SSVM) [28] can be utilized to find a parameter λ_W that minimizes the following objective function \mathcal{L} for N pairs of RNA sequences $x^{(i)}$ and secondary structures $y^{(i)}$:

$$\mathcal{L}(\lambda_W) = \sum_{i=1}^N \left(\max_{\hat{y} \in \mathcal{S}(x^{(i)})} \left[f(x^{(i)}, \hat{y}) + \Delta(y^{(i)}, \hat{y}) \right] - f(x^{(i)}, y^{(i)}) + C \|\lambda_W\|_1 \right),$$

where $\|\cdot\|_1$ is the l_1 norm and C is a weight for the l_1 regularization term to avoid overfitting to training data. Here, $\Delta(y^{(i)}, \hat{y})$ is a loss term of \hat{y} for $y^{(i)}$ defined as:

$$\begin{aligned} \Delta(y^{(i)}, \hat{y}) &= \delta^{FN} \times (\# \text{ of false negative base pairs}) + \delta^{FP} \\ &\quad \times (\# \text{ of false positive base pairs}), \end{aligned}$$

where δ^{FN} and δ^{FP} are tunable hyperparameters to control the trade-off between sensitivity and specificity for learning the parameters ($\delta^{FN} = 8.0$ and $\delta^{FP} = 1.0$ by default). It is also known as the margin term for structured models, enabling robust predictions by maximizing the margin between $f(x^{(i)}, y^{(i)})$ and $f(x^{(i)}, \hat{y})$ for $\hat{y} \neq y^{(i)}$.

The integrated scoring model described here was implemented as MXfold, which is short for the MaX-margin-based RNA FOLDing algorithm [15]. The benefit of the integrated model is the robustness against the overfitting to training data. MXfold can employ rich-parameterized weight parameters expected to lead to highly accurate prediction without heavy overfitting to the training data by falling back to the thermodynamic parameters for unobserved substructures in the training data.

4 Folding Algorithms

Given an RNA sequence, the number of secondary structures that can be folded into is proportional to the exponential order of the sequence length. Thus, it is not practical to find an optimal secondary structure by enumerating all the possible secondary structures. Fortunately, when considering the pseudoknot-free secondary structure architecture described in Subheading 2, we can employ a dynamic programming technique to find the optimal secondary structure.

In this section, we will show two criteria of the optimality in RNA secondary structure prediction. One is the MFE, which finds the most thermodynamically stable secondary structure. The other is the maximum expected accuracy (MEA), which finds a secondary structure that maximizes the expected accuracy based on the ensemble of RNA secondary structures distributed under the Boltzmann distribution.

4.1 Minimum Free Energy

As shown in Subheading 2, we assume that a given RNA secondary structure can be decomposed into loop substructures according to the nearest neighbor model and that the free energy of the given RNA secondary structure can be calculated by summing the free energy of every loop. This section will describe a method of finding the secondary structure with the MFE among all the possible secondary structures given an RNA sequence.

Since it is not practical to enumerate all the secondary structures that can be folded into, we use a dynamic programming technique to calculate the secondary structure with the minimum free energy. The dynamic programming technique divides a given problem into smaller subproblems, each of which is solved separately, and then the original problem is solved based on the solutions of the subproblems. In RNA secondary structure prediction with the free energy minimization, a given sequence is divided into shorter subsequences based on the loop decomposition described in Subheading 2.

Figure 3 shows that $x_{i:j} = x_i, \dots, x_j$, which is a subsequence of $x = x_1, \dots, x_n$, is decomposed into shorter subsequences based on the loop types. If $x_{i:j}$ is an external loop F that is not closed by any base pairs, then it is recursively decomposed until finding a closed loop. A closed loop C closed by a base pair (i, j) is decomposed in one of the following three ways, depending on the number of closing base pairs. If the closed loop C is closed only by the base pair (i, j) , it is a hairpin loop. If C is closed by two base pairs (i, j) and (k, l) , it is decomposed into either a stacking (for $k = i + 1$ and $l = j - 1$), a bulge loop (for $k = i + 1$ or $l = j - 1$), or an internal loop (for $k > i + 1$ and $l < j - 1$). A loop closed by a base pair (i, j) and two or more base pairs inside (i, j) is called a multibranch

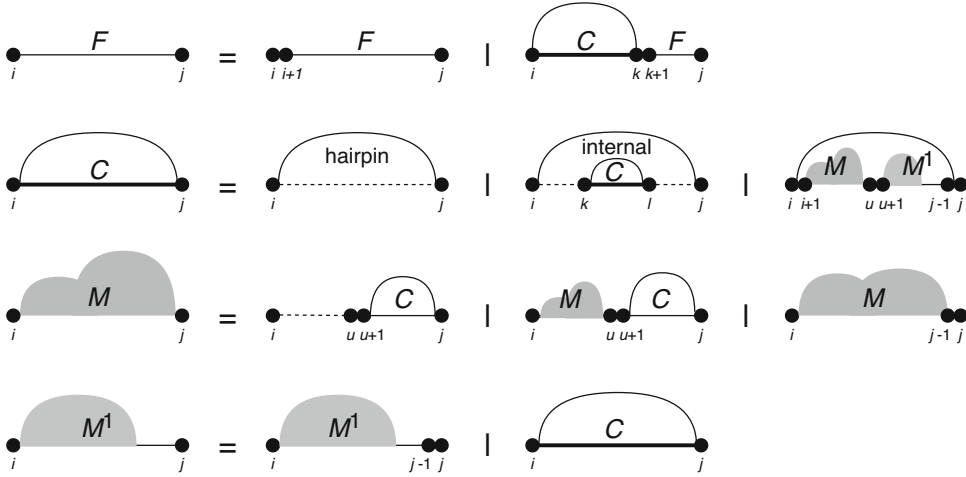


Fig. 3 Recursive decomposition of an RNA sequence for minimizing the free energy of the secondary structure according to the nearest neighbor model

loop. The multibranch loop is decomposed into the rightmost loop M^1 and the other loops M , where M^1 contains exactly one closed loop C and M contains one or more loops C , disambiguating the decomposition of RNA secondary structure.

Based on the decomposition shown in Fig. 3, the recursive equation for the dynamic programming algorithm to calculate the minimum free energy is as follows:

$$\begin{aligned}
 F_{ij} &= \min \left\{ F_{i+1,j}, \min_{i < k \leq j} C_{ik} + F_{k+1,j} \right\} \\
 C_{ij} &= \min \left\{ \mathcal{H}(i, j), \min_{i < k < l < j} C_{kl} + \mathcal{I}(i, j; k, l), \min_{i < u < j} M_{i+1,u} + M_{u+1,j-1} + a \right\} \\
 M_{ij} &= \min \left\{ \min_{i < u < j} (u - i + 1)c + C_{u+1,j} + b, \min_{i < u < j} M_{iu} + C_{u+1,j} + b, M_{i,j-1} + c \right\} \\
 M_{ij}^1 &= \min \left\{ M_{i,j-1}^1 + c, C_{ij} + b \right\} \\
 F_{ii} &= 0, C_{ii} = M_{ii} = M_{ii}^1 = \infty,
 \end{aligned} \tag{1}$$

where F_{ij} is the minimum free energy of the secondary structure of the subsequence $x_i : j$ and C_{ij} represents the MFE over closed structures. M_{ij} is the MFE of part of a multibranch loop that contains one or more loops, and M_{ij}^1 is the MFE of part of a multibranch loop with the rightmost loop.

Here, $\mathcal{H}(i, j)$ is the free energy of a hairpin loop $x_i : j$ closed by the base pair (i, j) . $\mathcal{I}(i, j; k, l)$ is the free energy of a stacking (for $k = i + 1$ and $l = j - 1$), a bulge loop (for $k = i + 1$ or $l = j - 1$), or

an internal loop (otherwise) closed by the base pairs (i, j) and (k, l) . The free energy of multibranch loop is approximated using parameters a , b , and c as follows:

$$a + b * (\text{\#of base pairs}) + c * (\text{\#of unpaired bases})$$

These parameters are determined by the methods shown in Subheading 3.

The algorithm starts with the smallest subsequences, i.e., empty strings, then fills F_{ij} , C_{ij} , M_{ij} , and M_{ij}^1 by using Eq. 1, and ends up with $F_{1, n}$ that will be filled by MFE. The algorithm requires $O(n^4)$ time for calculating C_{ij} of the free energy of the internal loops closed by the base pairs (i, j) and (k, l) . However, it can be reduced to $O(n^3)$ time by limiting the number of unpaired bases that form internal loops as L (generally $L = 30$). The MFE structure can be recovered by tracing back the recursive equations applied to construct MFE from $F_{1, n}$.

Since the MFE structure is thermodynamically most stable under the Boltzmann distribution, it has the highest probability over all the possible secondary structures, which corresponds to the maximum likelihood estimation (MLE).

4.2 Partition Function

An RNA sequence folds into not only the maximum free energy structure but also suboptimal structures distributed under the Boltzmann distribution. The probability of a secondary structure y for an RNA sequence x is as follows:

$$p(y|x) = \frac{1}{Z} \exp(-\beta E(x, y)) Z = \sum_{y \in \mathcal{S}(x)} \exp(-\beta E(x, y)),$$

where $\beta = 1/RT$ (R is the gas constant and T is the absolute temperature). Here, Z is called the partition function that calculates the sum of the Boltzmann factor for all possible secondary structures $\mathcal{S}(x)$. However, as with the MFE calculation, since it is not practical to enumerate all the possible secondary structures, we calculate Z using the dynamic programming technique similar to Eq. 1 used in the MFE calculation. The recursive equations to calculate the partition function Z are obtained by multiplying the free energy parameters (\mathcal{H} , I , a , b , and c) in Eq. 1 by $-\beta$ and replacing the “min” operator with the “logsumexp” operator, also known as the softmax function:

$$\text{logsumexp}(x_1, \dots, x_n) = \log \sum_{i=1}^n \exp(x_i).$$

Here are the recursive equations for calculating Z_{ij} , where $\log Z$ corresponds to F , and $\log Z^B$, $\log Z^M$, and $\log Z^{M^1}$ correspond to C , M , and M^1 , respectively:

$$\begin{aligned}
Z_{ij} &= Z_{i+1,j} + \sum_{i < k \leq j} Z_{ik}^B Z_{k+1,j} \\
Z_{ij}^B &= e^{-\beta \mathcal{H}(i,j)} + \sum_{i < k < l < j} Z_{kl}^B e^{-\beta \mathcal{G}(i,j;k,l)} + \sum_{i < u < j} Z_{i+1,u}^M Z_{u+1,j-1}^{M1} e^{-\beta a} \\
Z_{ij}^M &= \sum_{i < u < j} e^{-\beta(u-i+1)c} Z_{u+1,j}^M + \sum_{i < u < j} Z_{iu}^M Z_{u+1,j}^B e^{-\beta b} + Z_{i,j-1}^M e^{-\beta c} \\
Z_{ij}^{M1} &= Z_{i,j-1}^{M1} e^{-\beta c} + Z_{ij}^B e^{-\beta b} \\
Z_{ii} &= 1, Z_{ii}^B = Z_{ii}^M = Z_{ii}^{M1} = 0.
\end{aligned} \tag{2}$$

Focusing on a particular base pair (i, j) , we consider the proportion of secondary structures where x_i and x_j form a base pair in the ensemble of secondary structure $\mathcal{S}(x)$:

$$p_{ij} = \frac{\sum_{y \in \mathcal{S}_{ij}(x)} \exp(-\beta E(x, y))}{Z}, \tag{3}$$

where $\mathcal{S}_{ij}(x)$ is a subset of $\mathcal{S}(x)$, each of whose elements is a secondary structure with the base pair (i, j) . We refer to p_{ij} as the base pairing probability of the base pair (i, j) .

Equation 3 can be rewritten using the inside variable Z_{ij}^B , which corresponds to all the possible secondary structures of a subsequence inside the base pair (i, j) , and the outside variable \widehat{Z}_{ij}^B , which corresponds to all the possible secondary structures of subsequences outside the base pair (i, j) :

$$p_{ij} = \frac{Z_{ij}^B \widehat{Z}_{ij}^B}{Z}.$$

We can calculate \widehat{Z}_{ij}^B by applying Eq. 2 by the reverse order, that is, from longer sequences to shorter sequences:

$$\begin{aligned}
\widehat{Z}_{ij}^B &= Z_{1,i-1} Z_{j+1,n} \sum_{k(i,j)l} \widehat{Z}_{kl}^B \left\{ e^{-\beta \mathcal{G}(k,l;i,j)} + Z_{k+1,j-1}^M Z_{j+1,j-1}^M + Z_{j+1,l-1}^{M2} \right. \\
&\quad \left. \times e^{-\beta(a+(i-k-1)c)} + Z_{k+1,i-1}^{M2} e^{-\beta(a+(l-j-1)c)} \right\},
\end{aligned}$$

where $Z_{kl}^{M2} = \sum_u Z_{ku}^M Z_{u+1,l}^{M1}$. This algorithm is known as the McCaskill algorithm [29].

4.3 Maximum Expected Accuracy

The accuracy of RNA secondary structure prediction is assessed by the number of base pairs correctly predicted. In other words, we prefer a prediction $\widehat{y} \in \mathcal{S}(x)$ that maximizes the following gain function for a reference $y \in \mathcal{S}(x)$ defined as the weighted sum of the number of true positives (TP), that is, correctly predicted base pairs, and the number of true negatives (TN), that is, correctly predicted as non-base pairs:

$$G(y, \widehat{y}) = \gamma TP + TN,$$

where $\gamma > 0$ is a weight for base pairs. From the expectation of G with respect to the probability distribution of RNA secondary structures, we derive the following:

$$\begin{aligned} E_{y|x}[G(y, \hat{y})] &= \sum_{y \in \mathcal{S}(x)} G(y, \hat{y}) p(y|x) \\ &= \sum_{1 \leq i \leq j \leq |x|} [(\gamma + 1) p_{ij} - 1] \hat{y}_{ij} + C, \end{aligned} \quad (4)$$

where C is a constant that does not depend on \hat{y} . The derivation of this equation can be found in [7]. Here, y (resp., \hat{y}) $\in \mathcal{S}(x)$ is a binary-valued triangle matrix where y_{ij} (resp., \hat{y}_{ij}) = 1 if and only if bases x_i and x_j form a base pair in the reference structure (resp., the predicted structure). We can calculate \hat{y} that maximizes Eq. 4 with the following recursive equation similar to the Nussinov algorithm [26]:

$$M_{ij} = \max \begin{cases} M_{i+1,j} \\ M_{i,j-1} \\ M_{i+1,j-1} + (\gamma + 1)p_{ij} - 1 \\ \max_{i < k < j} M_{ik} + M_{k+1,j}. \end{cases}$$

In the Nussinov algorithm, the score for base pairs (i, j) is 1 for the canonical base pairs and -1 otherwise, whereas the above algorithm corresponds to a variant of the Nussinov algorithm with an alternative scoring function $(\gamma + 1)p_{ij} - 1$ with the base pairing probability p_{ij} .

The method based on Eq. 4 is called the generalized centroid estimator (GCE), which predicts RNA secondary structures by maximizing the expected accuracy for base pairs [7, 8].

In some cases, the accuracy is evaluated on per-base accuracy instead of per-base pair accuracy. In such cases, the gain function G' is defined as the weighted sum of the number of true positives bases (TP'), that is, correctly predicted as a one-half of a base pair, and the number of true negative bases (TN'), that is, correctly predicted as an unpaired base. As above, we derive the expectation of G' with respect to the distribution of RNA secondary structure:

$$E_{y|x}[G'(y, \hat{y})] = \sum_i \left[\sum_{j:j < i} (\gamma p_{ji} - q_i) \hat{y}_{ji} + \sum_{j:j > i} (\gamma p_{ij} - q_i) \hat{y}_{ij} \right] + C, \quad (5)$$

where $q_i = 1 - \sum_{j:j < i} p_{ji} - \sum_{j:j > i} p_{ij}$ and C is a constant that does not depend on \hat{y} . We can find \hat{y} that maximizes Eq. 5 using the following dynamic programming:

$$M_{ij} = \max \begin{cases} M_{i+1,j} \\ M_{i,j-1} \\ M_{i+1,j-1} + 2\gamma p_{ij} - q_i - q_j \\ \max_{i < k < j} [M_{ik} + M_{k+1,j}]. \end{cases}$$

It can be stated that the method based on Eq. 5 is the prediction of secondary structures by maximizing the expected accuracy not for base pairs but individual bases [9]. The above algorithm is traditionally referred to as the MEA of RNA secondary structure prediction.

5 Comparison of Methods

In this section, we present the results of comparing the performance of the RNA secondary structure prediction algorithms introduced so far. Performance comparison was conducted using the dataset established by Rivas [25], which is composed of TrainSetA, TestSetA, and TestSetB. TrainSetA and TestSetA were collected from the literature. Note that Train/TestSetA includes the datasets used in CONTRAfold, SimFold, and ContextFold. TestSetB was extracted from the Rfam database [30], where extracted sequences have 3D structure annotation and have less than 70% sequence identity with the sequences in TrainSetA. It is important to note that literature-based TrainSetA and Rfam-based TestSetB are structurally diverse, whereas it can be observed structural similarity between TrainSetA and TestSetA.

The accuracy of secondary structure prediction is evaluated by the Sensitivity (SEN) and Positive Predictive Value (PPV) of the base pairs, defined as follows:

$$\text{SEN} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

where TP is the number of correctly predicted base pairs, FP is the number of incorrectly predicted base pairs, and FN is the number of unpredicted base pairs in the reference structure. Since there is a trade-off between SEN and PPV, the F-value, which is the harmonic mean of SEN and PPV, is used for evaluation:

$$F = \frac{2 \times \text{SEN} \times \text{PPV}}{\text{SEN} + \text{PPV}}.$$

We benchmarked MXfold version 0.0.2 [15], CONTRAfold version 2.02 [9, 10], ContextFold version 1.00 [11], CentroidFold version 0.0.16 [7, 8], RNAfold in ViennaRNA package version 2.4.14 [3, 4], SimFold version 2.1 [13, 14], and RNAstructure version 6.2 [5, 6]. Among the machine learning-based methods, MXfold, CONTRAfold, and ContextFold trained their weight

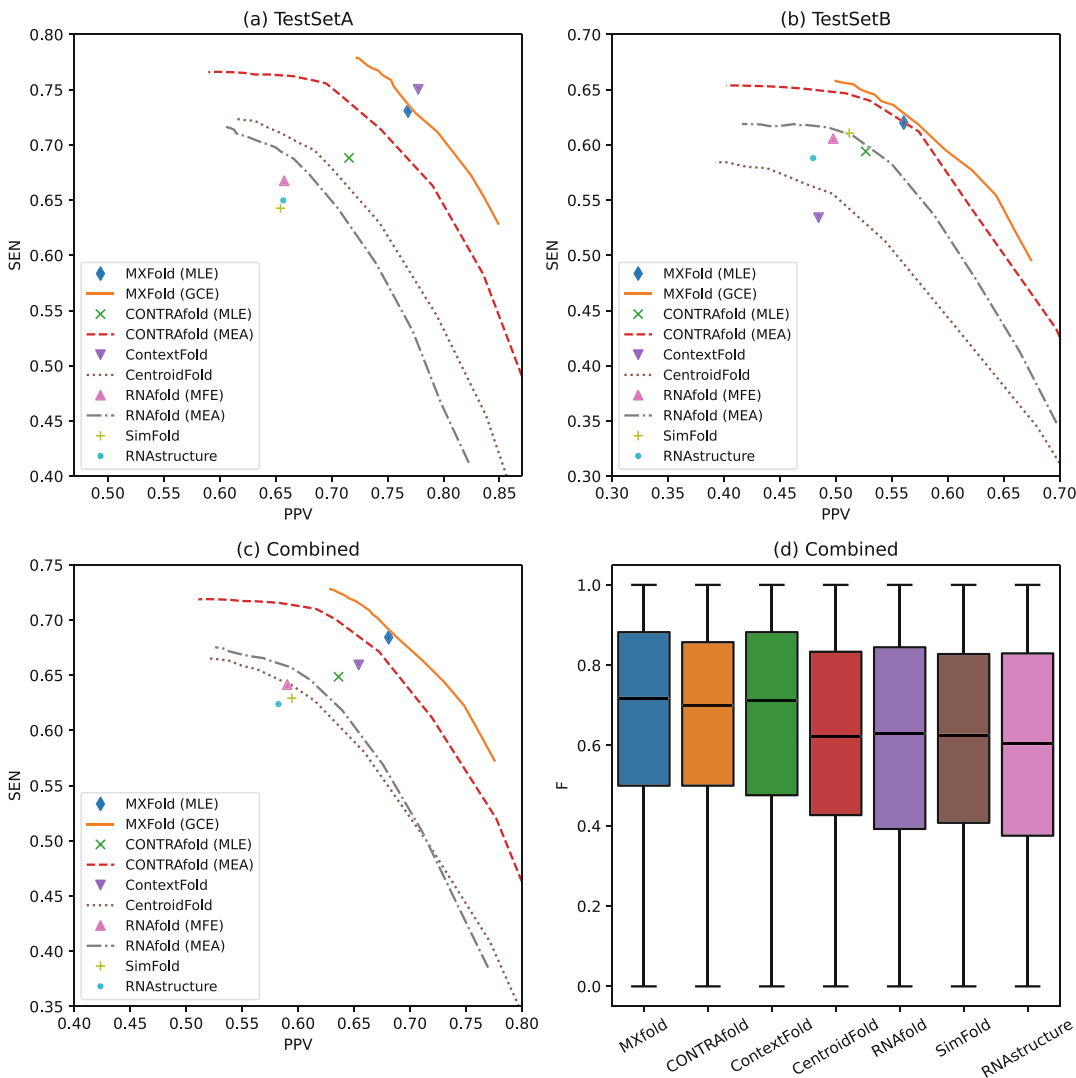


Fig. 4 PPV-SEN plots on (a) TestSetA, (b) TestSetB, and (c) the combined dataset with TestSetA and TestSetB for MXfold with MLE and generalized centroid estimator (GCE), CONTRAfold with MLE and maximum expected accuracy (MEA), ContextFold, CentroidFold, RNAfold with MFE and MEA, SimFold, and RNAstructure. (d) F -value on the combined test dataset for all the methods

parameters using TrainSetA. Since SimFold does not provide the training algorithm, we used the default parameter set trained on S-Full-Train dataset, which is a subset of TrainSetA. CentroidFold used the Boltzmann likelihood parameters [14] trained on S-Full-Train dataset.

Figures 4a, b show PPV-SEN plots of prediction accuracy on TestSetA and TestSetB, respectively. The MEA-based secondary structure prediction, MXfold (GCE), CentroidFold (GCE), CONTRAfold (MEA), and RNAfold (MEA), can control PPV and SEN

by the parameter γ , and therefore, their accuracies are shown by the curves for multiple $\gamma \in \{2^n \mid -5 \leq n \leq 10\}$.

For the predictions on TestSetA, ContextFold achieved the best accuracy ($F = 0.759$), followed by MXfold ($F = 0.754$ for GCE at $\gamma = 4.0$), MXfold ($F = 0.739$ for MLE), and CONTRAfold ($F = 0.719$ for MEA at $\gamma = 4.0$). On the other hand, for the predictions on TestSetB, which is structurally diverse from TrainSetA, we can observe that ContextFold achieved the worst precision ($F = 0.502$), as Rivas [12] has pointed out that ContextFold might fall in the overfitting. In contrast, we cannot observe heavy overfitting as in ContextFold for MXfold ($F = 0.591$ for GCE at $\gamma = 4.0$), MXfold ($F = 0.581$ for MLE), and CONTRAfold ($F = 0.573$ for MEA at $\gamma = 4.0$).

Figure 4c, d show the PPV-SEN plot and the distribution of F -value on the combined dataset with TestSetA and TestSetB. These results indicate that MXfold ($F = 0.685$ for GCE at $\gamma = 4.0$) and MXfold ($F = 0.673$ for MLE) achieved the best accuracy, followed by two machine learning-based methods, CONTRAfold ($F = 0.658$ for MEA at $\gamma = 4.0$) and ContextFold ($F = 0.651$), outperforming the thermodynamic based methods.

The two different prediction criteria mentioned in Subheading 4, namely, the MFE (or the MLE in the case of weight parameters) and the MEA, were compared using the same parameters. In all the methods, MEA is more accurate than its MFE or MLE counterpart. Especially for CONTRAfold, we observed a significant improvement of MEA ($F = 0.658$ at $\gamma = 4.0$ on the combined dataset) against MLE ($F = 0.628$), whereas for MXfold the difference between MLE ($F = 0.673$) and GCE ($F = 0.685$ at $\gamma = 4.0$) is small. In our opinion, this is a matter of the compatibility of prediction and training methods. It can be argued that MEA, which considers all the possible secondary structures in the prediction, is a successful approach that brings out the best in parameters from CLLM, which also considers all the possible secondary structures in training.

6 Prospects and Summary

In this chapter, we described RNA secondary structure prediction based on the energy models. In the performance comparison of the methods, MXfold, CONTRAfold, and ContextFold, which use machine learning based on energy models to estimate the weight parameters, achieve higher accuracy than the methods using thermodynamic parameters, suggesting that each RNA molecule does not form secondary structures based solely on the laws of biophysics we have ever known. Actual RNA molecules in vivo interact with other molecules including RNAs, proteins, and chemical compounds to form higher-order structures through more complex

processes. Therefore, predicting secondary structure, especially *in vivo*, is much more challenging. For more accurate secondary structure prediction, secondary structure profiles using secondary structure-specific chemical probes [31, 32] or RNase [33, 34], which have been actively studied in recent years, are expected to be a promising approach. Several methods based on thermodynamic parameters achieved much better accuracy by incorporating secondary structure profiles as pseudo-energy terms [35–37]. On the other hand, no methods based on machine learning support the secondary structure profiles to the best of our knowledge because only small amounts of public data on the secondary structure profiles were available for the training of machine learning. However, due to the increase in the secondary structure profile data available in recent years, machine learning-based methods will benefit from secondary structure profiles.

Acknowledgments

This chapter was partially supported by Grant-in-Aid for JSPS Fellows (No. 18J21767) from the Japan Society for the Promotion of Science (JSPS) to M. A. and Grant-in-Aid for Scientific Research (B) (No. 19H04210) and Challenging Research (Exploratory) (No. 19K22897) from JSPS to K.S.

References

- Zuker M (2003) Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res* 31:3406–3415
- Markham NR, Zuker M (2008) UNAFold: software for nucleic acid folding and hybridization. *Methods Mol Biol* 453:3–31
- Hofacker IL (2003) Vienna RNA secondary structure server. *Nucleic Acids Res* 31: 3429–3431
- Lorenz R, Bernhart SH, Höner Zu Siederdisen C et al (2011) ViennaRNA package 2.0. *Algorithms Mol Biol* 6:26
- Mathews DH, Andre TC, Kim J et al (1997) An updated recursive algorithm for RNA secondary structure prediction with improved thermodynamic parameters. *Mol Model Nucleic Acids* 15:246–257. <https://doi.org/10.1021/bk-1998-0682.ch015>
- Reuter JS, Mathews DH (2010) RNAstructure: software for RNA secondary structure prediction and analysis. *BMC Bioinformatics* 11:129
- Hamada M, Kiryu H, Sato K et al (2009) Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics* 25:465–473
- Sato K, Hamada M, Asai K et al (2009) CentroidFold: A web server for RNA secondary structure prediction. *Nucleic Acids Res* 37: 277–280
- Do CB, Woods DA, Batzoglou S (2006) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* 22:e90–e98
- Foo CS, Do CB, Ng AY (2008) Efficient multiple hyperparameter learning for log-linear models. In: Platt JC, Koller D, Singer Y et al (eds) *Advances in neural information processing systems*, vol 20, pp 377–384
- Zakov S, Goldberg Y, Elhadad M et al (2011) Rich parameterization improves RNA structure prediction. *J Comput Biol* 18:1525–1542
- Rivas E (2013) The four ingredients of single-sequence RNA secondary structure prediction. A unifying perspective. *RNA Biol* 10: 1185–1196
- Andronescu M, Condon A, Hoos HH et al (2007) Efficient parameter estimation for

- RNA secondary structure prediction. *Bioinformatics* 23:i19–i28
14. Andronescu M, Condon A, Hoos HH et al (2010) Computational approaches for RNA energy parameter estimation. *RNA* 16: 2304–2318
 15. Akiyama M, Sato K, Sakakibara Y (2018) A max-margin training of RNA secondary structure prediction integrated with the thermodynamic model. *J Bioinform Comput Biol* 16: 1840025
 16. Borer PN, Dengler B, Tinoco I Jr et al (1974) Stability of ribonucleic acid double-stranded helices. *J Mol Biol* 86:843–853
 17. Xia T, SantaLucia J Jr, Burkard ME et al (1998) Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs. *Biochemistry* 37:14719–14735
 18. Mathews DH, Sabina J, Zuker M et al (1999) Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J Mol Biol* 288:911–940
 19. Bloomfield VA, Crothers DM (2000) Nucleic acids: structures, properties and functions. *J Med Chem* 43(24):4721–4722
 20. Mathews DH, Disney MD, Childs JL et al (2004) Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proc Natl Acad Sci U S A* 101:7287–7292
 21. Turner DH, Mathews DH (2010) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res* 38:D280–D282
 22. Turner DH, Sugimoto N, Freier SM (1988) RNA structure prediction. *Annu Rev Biophys Chem* 17:167–192
 23. Sakakibara Y, Brown M, Hughey R et al (1994) Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Res* 22:5112–5120
 24. Eddy SR, Durbin R (1994) RNA sequence analysis using covariance models. *Nucleic Acids Res* 22:2079–2088
 25. Rivas E, Lang R, Eddy SR (2012) A range of complex probabilistic models for RNA secondary structure prediction that includes the nearest-neighbor model and more. *RNA* 18: 193–212
 26. Nussinov R, Jacobson AB (1980) Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc Natl Acad Sci U S A* 77:6309–6313
 27. Zuker M, Stiegler P (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res* 9:133–148
 28. Tsochantaridis I, Joachims T, Hofmann T et al (2005) Large margin methods for structured and interdependent output variables. *J Mach Learn Res* 6:1453–1484
 29. McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers* 29: 1105–1119
 30. Gardner PP, Daub J, Tate J, Moore BL, Osuch IH, Griffiths-Jones S, Finn RD, Nawrocki EP, Kolbe DL, Eddy SR, Bateman A (2011) Rfam: Wikipedia, clans and the “decimal” release. *Nucleic Acids Res* 39(Database issue): D141–D145. <https://doi.org/10.1093/nar/gkq1129>
 31. Lucks JB, Mortimer SA, Trapnell C et al (2011) Multiplexed RNA structure characterization with selective 2'-hydroxyl acylation analyzed by primer extension sequencing (SHAPE-Seq). *Proc Natl Acad Sci U S A* 108: 11063–11068
 32. Rouskin S, Zubradt M, Washietl S et al (2014) Genome-wide probing of RNA structure reveals active unfolding of mRNA structures in vivo. *Nature* 505:701–705
 33. Underwood JG, Uzilov AV, Katzman S et al (2010) FragSeq: transcriptome-wide RNA structure probing using high-throughput sequencing. *Nat Method* 7:995–1001
 34. Kertesz M, Wan Y, Mazor E et al (2010) Genome-wide measurement of RNA secondary structure in yeast. *Nature* 467:103–107
 35. Deigan KE, Li TW, Mathews DH et al (2009) Accurate SHAPE-directed RNA structure determination. *Proc Natl Acad Sci U S A* 106: 97–102
 36. Ouyang Z, Snyder MP, Chang HY (2013) SeqFold: genome-scale reconstruction of RNA secondary structure integrating high-throughput sequencing data. *Genome Res* 23: 377–387
 37. Lorenz R, Wolfinger MT, Tanzer A et al (2016) Predicting RNA secondary structures from sequence and probing data. *Methods* 103: 86–98



RNA Secondary Structure Alteration Caused by Single Nucleotide Variants

Risa Karakida Kawaguchi and Hisanori Kiryu

Abstract

A point mutation in coding RNA can cause not only an amino acid substitution but also a dynamic change of RNA secondary structure, leading to a dysfunctional RNA. Although *in silico* structure prediction has been used to detect structure-disrupting point mutations known as riboSNitches, exhaustive simulation of long RNAs is needed to detect a significant enrichment or depletion of riboSNitches in functional RNAs. Here, we have developed a novel algorithm Radiam (RNA secondary structure Analysis with Deletion, Insertion, And substitution Mutations) for a comprehensive riboSNitch analysis of long RNAs. Radiam is based on the ParasoR framework, which efficiently computes local RNA secondary structures for long RNAs. ParasoR can compute a variety of structure scores over globally consistent structures with maximal span constraints for the base pair distance. Using the reusable structure database made by ParasoR, Radiam performs an efficient recomputation of the secondary structures for mutated sequences. An exhaustive simulation of Radiam is expected to find reliable riboSNitch candidates on long RNAs by evaluating their statistical significance in terms of the change of local structure stability.

Key words RNA secondary structure, Local structure, Maximal span constraint, riboSNitch, SNP, Mutation

1 Introduction

RNA secondary structure is regulated by the primary sequence of RNA, in which stable base pairs are formed between complementary bases, such as A–U and C–G, as well as other minor pairs (e.g., wobble pairs between G and U). Each base type within substructures, such as base pairs, their combinations (stacking), or loops, differently contributes to the free energy changes of a specific structure [1]. Due to the variety of free energy changes for each substructure and base composition, the global structure can be drastically changed if the formation of substructures is partially inhibited by an external factor. A riboswitch is one such example, where ligand binding triggers a dynamic conformational change by inhibiting the formation of a local structure [2]. The characteristic

of a riboswitch is utilized for designing an intracellular sensor by modulating the efficiency of RNA functions, such as downstream translation or cleavage. Thus, even a local conformational change can impair RNA function by globally inducing dynamic conformational changes.

Similar to a riboswitch, a conformational variation can be induced through base differences among individuals, such as single nucleotide polymorphisms (SNPs) or variants (SNVs). The structure-disrupting variations, referred to as riboSNitches, were found using the *in silico* structure prediction method SNPfold [3]. A previous study also analyzed the effect of riboSNitches around the binding sites of the RNA binding protein (RBP) HuR using ViennaRNA [4]. Their results revealed the presence of SNPs that have a manyfold effect on binding affinity and significant under-selection of SNPs that decrease the binding affinity of the RBP. Because of potentially a wide range of biological regulations associated with RBPs, the relationship between riboSNitches and disease phenotypes has been investigated in specific biological contexts. For example, He et al. performed a statistical analysis of riboSNitch-enriched and -depleted elements in the cancer genome [5]. They found that riboSNitches are significantly enriched or depleted in long noncoding RNA (lncRNA) and mRNA untranslated regions (UTRs) in certain types of cancers, suggesting the influence of structure-disrupting mutations on pathogenesis.

Furthermore, *in vitro* genome-wide riboSNitch detection has been performed using high-throughput RNA structure analysis, which is an emerging technology for evaluating the likelihood of being paired at single nucleotide resolution. Wan et al. performed whole-transcriptome analysis of a human family trio using parallel analysis of RNA structure (PARS) to evaluate the structural impact of SNPs [6]. Their statistical analysis revealed selective depletion of riboSNitches compared to structurally synonymous SNVs within several types of functional regions, such as predicted target sites of miRNAs and RBPs. This dataset and mutate-and-map strategy from the RMDB database [7] were also applied to validate and improve riboSNitch detection using *in silico* prediction [8, 9].

Although these *in vitro* analyses are a powerful tool, *in silico* prediction plays an important role for riboSNitch detection because few structure datasets of human trios are found in structure databases (e.g., RNAex [10] or RMDB). However, the simulation of RNA secondary structure is a time-consuming task itself and hardly expanded to an exhaustive riboSNitch simulation. Thus, most of the previous studies have focused on the fast simulation of conformational changes for a limited number of observed mutations (Table 1). The RiboSNitchDB database has been also constructed from a set of known eQTL SNVs [11].

On the other hand, performing an exhaustive simulation of background mutations is critical for assessing the statistical impact

Table 1
Previous studies to evaluate the conformational impact of mutation on RNA secondary structure stability

Method	Description	Reference
SNPfold	Comparison of partition functions and structures obtained by Boltzmann sampling	Halvorsen et al. [3]
RNASnp	Empirical p -values for Euclidian distance of base pairing probability differences	Sabarinathan et al. [13]
remuRNA	Relative entropy of Boltzmann ensembles	Salari et al. [17]
mutaRNA	RNASnp + remuRNA	Miladi et al. [18]
Rchange	Changes of entropy of Boltzmann ensembles with the maximal span constraint	Kiryu and Asai [19]
RNAmutants	MFE structure and the partition function over all secondary structures for mutated sequences	Waldispühl et al. [12]
Johnson AD et al.	Differences of free energy changes between optimal and suboptimal structures	Johnson et al. [20]

of each mutation, particularly for the detection of significant enrichment or depletion of structure-disrupting mutations. To address this problem, Waldispühl et al. developed RNAmutants to efficiently evaluate conformational changes caused by multipoint mutations using the user-specified integer K [12]. RNAmutants demonstrates the computation of minimum free energy (MFE) structure and partition function using a dynamic programming (DP) technique. However, it is not feasible for long RNAs due to the high demand of computational time and resource. Specifically, its computational complexity is $O(N^3K^2)$ for the sequence of length N . RNASnp is another method that uses an empirical distribution obtained from random sequences with varying GC content to compute p -values for differences in base pairing probability [13]. However, the scale of conformational change may vary depending on both the mutated base and GC content of the whole sequence (discussed in Notes). Therefore, using fast mutation simulation within the input sequence may help evaluate statistical significance of each riboSNitch candidate more precisely.

In this chapter, we introduce a novel algorithm Radium (RNA secondary structure Analysis with Deletion, Insertion, And substitution Mutations), which can simulate RNA secondary structure alterations by single point mutations based on the ParasoR platform [14]. ParasoR is an algorithm designed to compute a variety of RNA secondary structure scores, such as stem probability or accessibility, for long RNAs. Using a structure database in the form robust for overflow and underflow errors, ParasoR enables

distributed computation of genome-level RNA sequences. In the Radium extension, the structure database is suitably updated after introducing the mutation. Although the original computational complexity is $O(N^2 W^2)$ for all possible mutations, where W is the maximal span for base pair distance, Radium can distribute the simulation of all mutations to multiple computational nodes with a computational complexity of $O(W^3)$ for a single mutation.

2 Methods

2.1 Radium Algorithm

We have developed Radium, which can efficiently simulate structural alterations caused by every possible SNV, using the criteria of structure score differences. Because Radium is based on the framework of the ParasoR algorithm [14], which is designed for genome-wide local structure prediction, Radium is also applicable to fast SNV simulation for long RNAs using a precomputed structure database.

To predict riboSNitches, Radium evaluates the level of conformational alteration induced by each SNV using two values, namely, the maximum difference and Pearson's correlation coefficient of surrounding stem probabilities with and without mutations. In the ParasoR framework, structure scores, such as stem probability, are computed for all possible structures under the constraint of maximal base pair distance W , where W is generally set as $\ll N$ (Fig. 1). The computation of a "local structure" can significantly decrease the computational complexity from $O(N^3)$ to $O(NW^2)$ while ensuring that a large portion of structure scores are less affected (also see the ParasoR chapter).

Further implementation in Radium is a function of recomputing the minimum required region potentially changed by the SNV in the DP ratio matrix of ParasoR. As shown in Fig. 2, the influence of a single point mutation is propagated to inside and outside matrices exclusively. For example, a mutation can affect the right and left regions of inside and outside Outer variables from its location, which correspond to the sum of free energy changes for structures of a subsequence as defined in Rfold model [16]. Other inside and outside variables (α and β , respectively) are also exclusively dependent on the presence of the mutation, which leads to

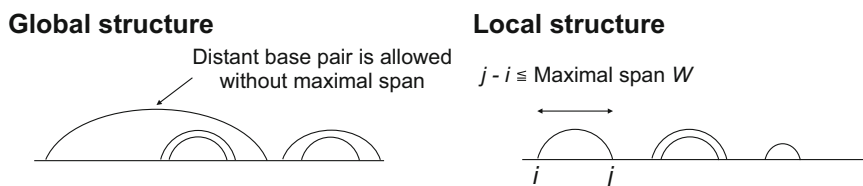


Fig. 1 Graphical examples of pseudoknot-free global and local RNA secondary structures

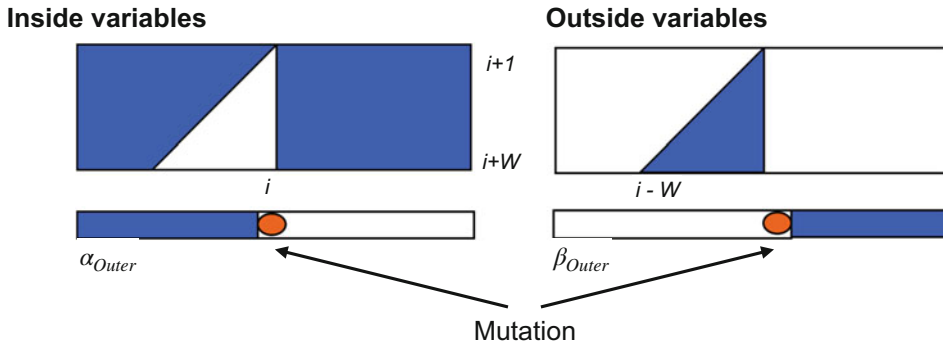


Fig. 2 Efficient recomputation of inside and outside variables using the Radium algorithm. The red circle corresponds to a mutated position. Rectangles at the bottom represent inside and outside Outer vectors, whereas rectangles at the top represent inside and outside matrices of all other states. The region in white shows values affected by mutations and is to be updated while the one in blue is independent from the mutations

saving half the computation time. Although the idea does not change its theoretical computational complexity $O(NW^2)$, a combination of minimum computation and local recomputation enables Radium to simulate the SNV-induced conformational change twice as fast as local probability computation using ParasorR.

Figure 3 shows a pseudocode of Radium algorithm, in which only inside and outside variables affected by a mutation are recomputed. For example, inside Outer variables, which are represented by $\alpha_{Outer}(i)$, correspond to the summation of free energy changes of all structures for a partial sequence from the first to the i th base. $\alpha_{Outer}(i)$ is iteratively obtained as follows:

$$\alpha_{Outer}(i) = \alpha_{Outer}(i-1) + \sum_{k=i-W-1}^{i-1} \alpha_{Outer}(k) \cdot \alpha_{Stem}(k, i) \cdot t(Outer \rightarrow Outer \bar{n} Stem)$$

where $\alpha_{Stem}(k, i)$ represents an inside Stem variable for the structures that have a base pair between the k and i th position and $t(Outer \rightarrow Outer \cdot Stem)$ represents a transition energy between Outer and Stem state. As such, the computation of other DP variables is performed iteratively following this equation. The partition function, which is required to compute the probability of each structure, is also obtained from $\alpha_{Outer}(N)$, where the sequence length is N . Using the location-specific dependency of inside and outside variables in Rfold model, Radium updates $\alpha_{Outer}(i)$ only when the mutated position m is equal to or larger than $i-1$. $\alpha_{Outer}(m-1)$ should be also updated because the m th base is taken into account in $\alpha_{Outer}(m-1)$ via a dangling energy. While Fig. 3 is an algorithm for Rfold model, Radium actually carries out

Algorithm 1 Radium algorithm

```

1:  $N :=$  sequence length
2:  $W :=$  maximal span of base pair
3:  $m :=$  position of a point mutation (starts from 0)
4: Compute DP matrices of inside ( $\alpha$ ) and outside ( $\beta$ ) variables for an original sequence.
5: Copy the DP matrices.
6: for  $j = m$  to  $N$  do
7:   for  $i = (j - 1)$  to  $(j - W - 1)$  do
8:     Update  $\alpha(i, j)$ 
9:   end for
10:  Update  $\alpha_{Outer}(j)$ 
11: end for
12: for  $j = N$  to 0 do
13:   for  $i = (j - W - 1)$  to  $(j - 1)$  do
14:     if  $i < (m - 1)$  and  $(m + 1) < j$  then
15:       Update  $\beta(i, j)$ 
16:     end if
17:   end for
18:   if  $j \leq (m + 1)$  then
19:     Update  $\beta_{Outer}(j)$ 
20:   end if
21: end for

```

Fig. 3 A pseudocode of Radium algorithm based on Rfold platform

the same recomputation based on ParasoR model [14], in which DP variables are stored in the form of the ratio of adjacent DP variables to overcome overflow and underflow problems.

2.2 Preprocessing of Radium to Construct ParasoR Databases

The preprocess step of Radium is similar to that of ParasoR and includes the construction of the structure database. Below is an example of two steps using two computer nodes, namely, DP ratio matrix and base pairing probability matrix computation. The name of the structure database is set using the “name” option, while an input sequence is set using either the f option ($-f$ [sequence]) for a character string or the “input” option ($--input$ [fasta file]) to read from a file.

Examples:

```

ParasoR -i 0 -k 2 -name=[database] # run using the first node
ParasoR -i 1 -k 2 -name=[database] # run using the second node
ParasoR -i 2 -k 2 -name=[database] # connect the ratios of
inside and outside variables

```

Examples:

```

ParasoR -i 0 -k 2 -name=[database] --stemdb
# compute the stem probability for the first half of the
sequence.

```

```

ParasoR -i 1 -k 2 -name=[database] --stemdb
# compute the stem probability for the last half of the
sequence.
ParasoR -i 2 -k 2 --name=[database] -stemdb
# connect the stem probability files

```

Based on the computed database, Radium partially recomputes the ratios of inside and outside variables for a sequence with a single point mutation.

2.3 Simulation of Single Point Mutations

The trigger of Radium computation is the *m* option with either of the initials of substitution, insertion, or deletion as Examples 1, 2, and 3, respectively. Other options related with the database and structure prediction follow the ParasoR framework, including “--constraint” for the maximal span and “--window” for the target window size, used for feature computation. The output can be selected from the difference of stem probability, accessibility, or base pairing probability using --stem, --acc, and --bpp options, respectively. While Examples 1–3 specify a target region based on each chunk size (e.g., the first of two chunks for a whole sequence in Examples), the exact start and end point for the mutation location can be set as Example 4. The mout option can set the output file name to write the result of Radium simulation.

Example 1:

```

ParasoR -i 0 -k 2 --name=[database] --mout=[file name] -m S --
stem # insertion

```

Example 2:

```

ParasoR -i 0 -k 2 --name=[database] --mout=[file name] -m I --
stem # insertion

```

Example 3:

```

ParasoR -i 0 -k 2 --name=[database] --mout=[file name] -m D --
stem # deletion

```

Example 4:

```

ParasoR -s 1 -e 100 --name=[database] --mout=[output file
name] -m S --stem # substitution

```

Below is an example output of p53 mRNA substitution analysis using the --stem option. Each column of the output exhibits information on mutation location, base type after mutation (0: deletion, 1: A, 2: C, 3: G, and 4: U), maximum difference of stem probability within the region $(-W, W)$ from the mutation, Pearson’s correlation coefficient between stem probabilities of original and mutated sequences within the region $(-W, W)$, and the same coefficient within the region of the specified window size $(-window/2,$

$window/2$) around the mutation. The window size can be set using the “window” option.

Example outputs:

```
pos base_type max_diff correlation correlation(win)
1 2 0.049896 0.999977 0.998630
1 3 0.989273 0.995392 0.988951
1 4 0.0637124 0.999934 0.998632
2 1 0.814961 0.991746 0.763569
2 2 0.804554 0.984432 0.953104
2 3 0.810627 0.994746 0.88879
...
```

2.4 Computation of p -Values

In the ParasoR source code directory, we prepared an R script `plot_mut_max_diff.R` for the basic analysis of the Radium output. It can compute the empirical p -value from the ranking of the maximum difference of stem probability for any base type and each base individually.

Example: `Rscript plot_mut_max_diff.R [Radium output file name]`

3 Notes

For detecting a riboSNitch, Radium computes the maximum difference and correlation coefficient of stem probabilities between the original and mutated sequences. These scores and other values obtained from the correlation coefficient were applied for riboSNitch detection in a previous study [6]. Most other scores used in previous studies, such as relative entropy of structures and comparison of MFE structures [12, 20], are not feasible for long RNA analysis or exhaustive mutation simulation. Radium can reproduce the computation of a classical structure prediction only with the limitation for the maximal base pairing span and, therefore, has wide applicability in computing multiple structure scores, such as a variety of distances for the base pairing probability matrix [13]. Although a suitable score for riboSNitch detection was evaluated using *in vitro* structure data [9], a significant difference has not been found between score choices for *in silico* methods. Hence, the use of other features for computation would be an important future work to identify the best criteria for detecting conformational changes.

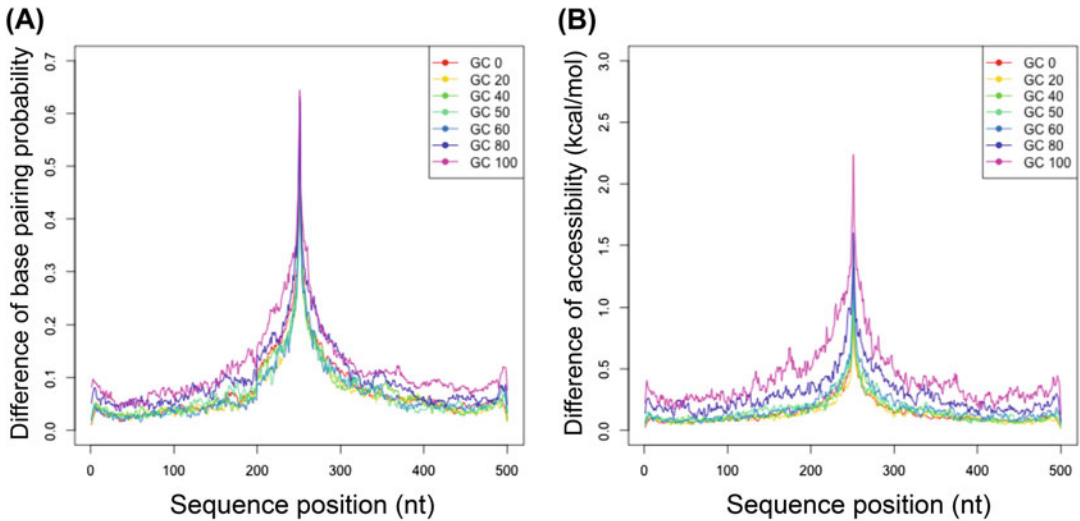


Fig. 4 Relationship between the impact of point mutations and GC content according to the base pairing probability difference ($W = 200$). The average of maximum differences of base pairing probability for each position is computed for 200 random sequences, which are 500 nt long and have a GC content ranging from 0% to 100%, with a step of 20%. A single point substitution is introduced at the center of the sequence

For local structure analysis using the ParasoR framework, the propagation of conformational changes becomes weak in general with the distance from the mutation. For example, the average of the maximum difference of base pairing probabilities and accessibilities between sequences with and without substitutions shows a sharp peak around the mutation and modest changes over the entire region (Fig. 4). Regarding the conformational changes induced by deletions and insertions, we computed the maximum difference of base pairing probability for each base of random sequences with deletions and insertions. For structure prediction, several maximal constraints are applied, ranging from 50 to 300, which is considered a reasonable range for local structure analysis [14]. A substantial impact is observed within range $(-W, W)$ from the mutation regardless of the constraint difference (Fig. 5). Moreover, impact scores are almost consistent across different maximal constraints, except for the shortest one ($W = 50$). Because the recommended maximal constraint ranges from 150 to 200 [15, 16], the evaluation of conformational changes within the range $(-W, W)$ from the mutation used in Radium would be a promising approach for local riboSNitch detection.

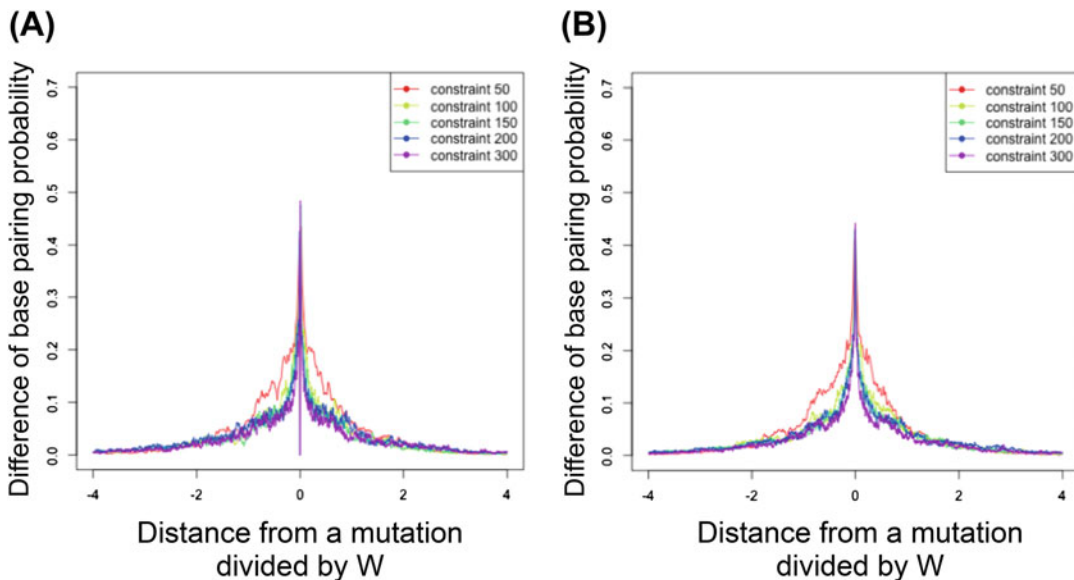


Fig. 5 The average of maximum differences of base pairing probability produced by (a) deletion and (b) insertion. Base pairing probability matrices are computed 1000 times for random sequences of 10,000 nt, with GC content = 50%. The x -axis shows the distance from the mutation relative to the maximal span $W = 50, 100, 150, 200,$ and 300

The advantage of comprehensive simulation of single point mutations using structure databases of ParasoR is that we can extract riboSNitch candidates using a statistical criterion beyond known biases for conformational vulnerability. GC content of the original sequence is the most well-known bias for the structure stability as well as base pairing probability [13, 14]. In general, sequences with high GC% tend to contain more base pairs due to a higher contribution of GC-included stacking loops to structural stability. This may contribute to the results of Fig. 4, where sequences with a higher GC content tend to exhibit stronger and wider impacts caused by a mutation as expected.

To evaluate the relationship between the base type and strength of deleterious effect, we show examples of riboSNitch analysis for a substitution introduced into the p53 mRNA sequence ($N = 1749$). Using an empirical distribution approach for the maximum difference of stem probability around the mutation, the conformational impact score of each mutation is converted to a p -value based on the ranking of all scores (Fig. 6). Consequently, the mutation introduced to a G base is most biased to have lower p -values or cause the stronger changes compared to other base types. The high

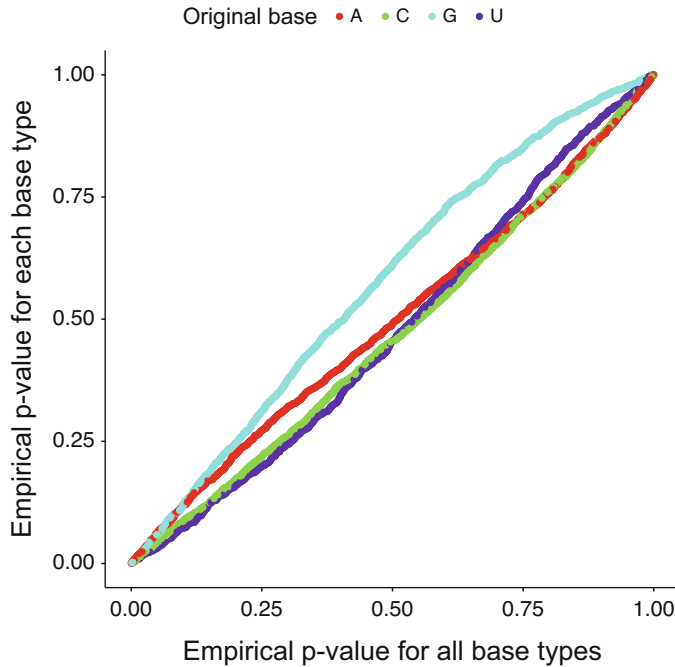


Fig. 6 Comparison of the p -values converted from the ranking of the maximum differences of stem probability around the mutation. The conformational impact scores are obtained by comparing the original and mutated p53 mRNA sequence ($N = 1749$) for every possible point substitution. The scores are then converted to the p -values based on the empirical distribution at once (y-axis) or within the groups according to the original base type at the mutated position (x-axis)

potential of G to form stable base pairs either with C or U may explain its specific tendency for p -value distribution.

To assess significant enrichment or depletion of riboSNitches in a specific RNA or region, selecting an appropriate way to normalize frequency and background impact on local structure stability is critical. While each mutation type is expected to cause a different conformational effect as shown in Fig. 5, the substitution pattern has also a different probability of appearing by different biological mechanisms, such as the high C to U mutation rate due to UV radiation [21, 22]. Using our Radiam extension, the maximum difference and correlation coefficient of stem probability between the original and mutated sequences are computed for each substitution type (Figs. 7 and 8). They show a roughly opposite tendency for each substitution pattern in terms of their medians, suggesting the similar direction of the conformational changes. However, there is a difference between these two scores that the maximum

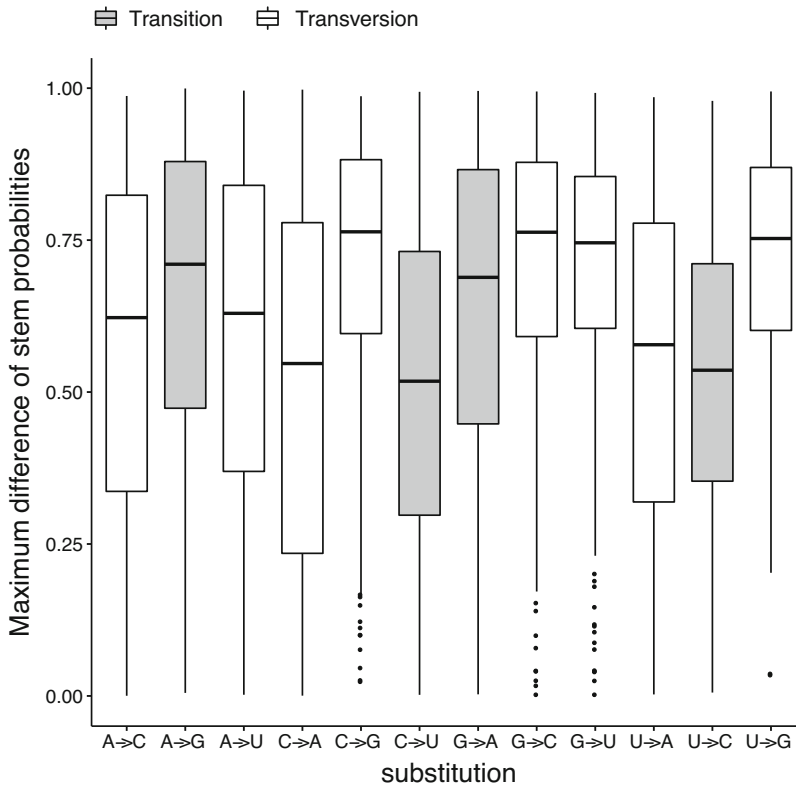


Fig. 7 Distribution of maximum difference of stem probabilities between the original and substituted p53 mRNA

difference tends to show less overlap between the boxes of the interquartile range. This is possibly because the maximum difference is likely to be obtained from a drastic conformational change from the proximal region while the computation of correlation highly depends on the modest changes over the entire region as well as the proximal one. The maximum difference is, therefore, expected to detect riboSNitch candidates that have a strong impact on the proximal region, whereas the correlation of stem probabilities may be suitable for detecting those which cause a modest but wider impact on RNA structure.

Thus, Radiam is a promising method that has a high potential in identifying conformational impacts of mutations from various aspects, considering known biases for riboSNitch detection. The flexibility of Radiam analysis can be used to explore the evolutionary selection of a mutation through dynamic changes in local secondary structures.

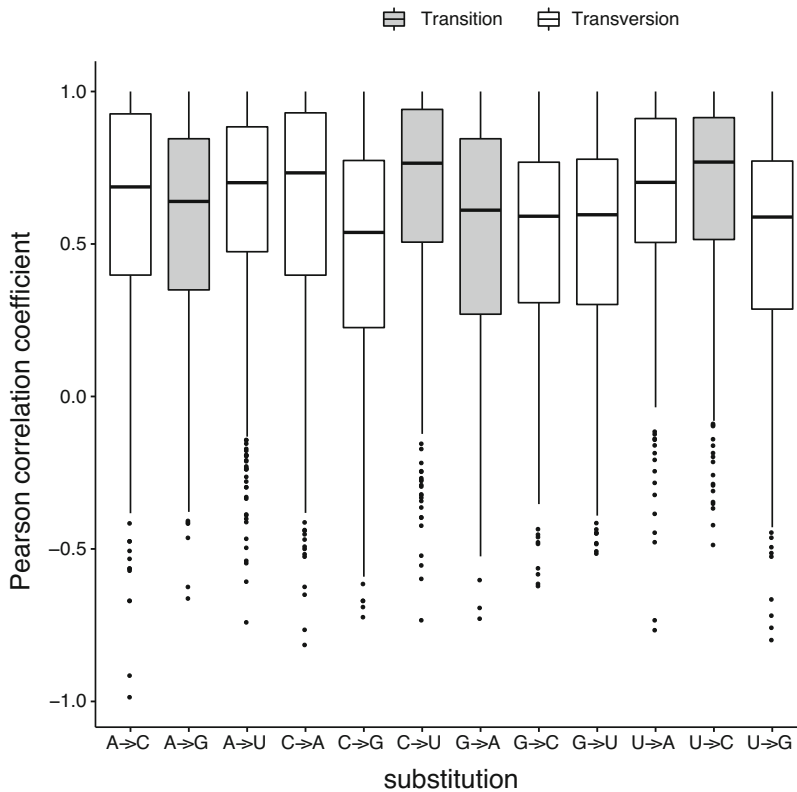


Fig. 8 Distribution of Pearson's correlation coefficient for stem probabilities between the original and substituted p53 mRNA

References

1. Mathews DH, Disney MD, Childs JL, Schroeder SJ, Zuker M, Turner DH (2004) Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proc Natl Acad Sci U S A* 101:7287–7292. <https://doi.org/10.1073/pnas.0401799101>
2. Serganov A, Nudler E (2013) A decade of riboswitches. *Cell* 152:17–24. <https://doi.org/10.1016/j.cell.2012.12.024>
3. Halvorsen M, Martin JS, Broadaway S, Laederach A (2010) Disease-associated mutations that alter the RNA structural ensemble. *PLoS Genet* 6:e1001074. <https://doi.org/10.1371/journal.pgen.1001074>
4. Shatoff E, Bundschuh R (2020) Single nucleotide polymorphisms affect RNA-protein interactions at a distance through modulation of RNA secondary structures. *PLoS Comput Biol* 16:e1007852. <https://doi.org/10.1371/journal.pcbi.1007852>
5. He F, Wei R, Zhou Z, Huang L, Wang Y, Tang J, Zou Y, Shi L, Gu X, Davis MJ, Su Z (2019) Integrative analysis of somatic mutations in non-coding regions altering RNA secondary structures in cancer genomes. *Sci Rep* 9:8205. <https://doi.org/10.1038/s41598-019-44489-5>
6. Wan Y, Qu K, Zhang QC, Flynn RA, Manor O, Ouyang Z, Zhang J, Spitale RC, Snyder MP, Segal E, Chang HW (2014) Landscape and variation of RNA secondary structure across the human transcriptome. *Nature* 505:706–709. <https://doi.org/10.1038/nature12946>
7. Yesselman JD, Tian S, Liu X, Shi L, Li JB, Das R (2018) Updates to the RNA mapping database (RMDb), version 2. *Nucleic Acids Res* 46:D375–D379. <https://doi.org/10.1093/nar/gkx873>
8. Corley M, Solem A, Qu K, Chang HY, Laederach A (2015) Detecting riboSNitches with RNA folding algorithms: a genome-wide

- benchmark. *Nucleic Acids Res* 43:1859–1868. <https://doi.org/10.1093/nar/gkv010>
9. Woods CT, Laederach A (2017) Classification of RNA structure change by ‘gazing’ at experimental data. *Bioinformatics* 33:1647–1655. <https://doi.org/10.1093/bioinformatics/btx041>
 10. Wu Y, Qu R, Huang Y, Shi B, Liu M, Li Y (2016) RNAex: an RNA secondary structure prediction server enhanced by high-throughput structure-probing data. *Nucleic Acids Res* 44:W294–W301. <https://doi.org/10.1093/nar/gkw362>
 11. Lin J, Chen Y, Zhang Y, Ouyang Z (2020) Identification and analysis of RNA structural disruptions induced by single nucleotide variants using Riprap and RiboSNitchDB. *NAR Genomics Bioinformatics* 2:3. <https://doi.org/10.1093/nargab/lqaa057>
 12. Waldispühl J, Devadas S, Berger B, Clote P (2009) RNAmutants: a web server to explore the mutational landscape of RNA secondary structures. *Nucleic Acids Res* 37(Issue suppl_2):W281–W286. <https://doi.org/10.1093/nar/gkp477>
 13. Sabarinathan R, Tafer H, Seemann SE, Hofacker IL, Stadler PF, Gorodkin J (2013) The RNAsnp web server: predicting SNP effects on local RNA secondary structure. *Nucleic Acids Res* 41(Web Server Issue):W475–W479. <https://doi.org/10.1093/nar/gkt291>
 14. Kawaguchi R, Kiryu H (2016) Parallel computation of genome-scale RNA secondary structure to detect structural constraints on human genome. *BMC Bioinformatics* 17:203. <https://doi.org/10.1186/s12859-016-1067-9>
 15. Lange SJ, Maticzka D, Möhl M, Gagnon JN, Brown CM, Backofen R (2012) Global or local? Predicting secondary structure and accessibility in mRNAs. *Nucleic Acids Res* 40:5215–5226. <https://doi.org/10.1093/nar/gks181>
 16. Kiryu H, Kin T, Asai K (2008) Rfold: an exact algorithm for computing local base pairing probabilities. *Bioinformatics* 24:367–373. <https://doi.org/10.1093/bioinformatics/btm591>
 17. Salari R, Kimchi-Sarfaty C, Gottesman MM, Przytycka TM (2013) Sensitive measurement of single-nucleotide polymorphism-induced changes of RNA conformation: application to disease studies. *Nucleic Acids Res* 41:44–53. <https://doi.org/10.1093/nar/gks1009>
 18. Miladi M, Raden M, Diederichs S, Backofen R (2020) MutaRNA: analysis and visualization of mutation-induced changes in RNA structure. *Nucleic Acids Res* 48:W287–W291. <https://doi.org/10.1093/nar/gkaa331>
 19. Kiryu H, Asai K (2012) Rchange: algorithms for computing energy changes of RNA secondary structures in response to base mutations. *Bioinformatics* 28:1093–1101. <https://doi.org/10.1093/bioinformatics/bts097>
 20. Johnson AD, Trumbower H, Sadee W (2011) RNA structures affected by single nucleotide polymorphisms in transcribed regions of the human genome. *Webmedcent Bioinformatics*, p 2
 21. Brash DE (2015) UV signature mutations. *Photochem Photobiol* 91(1):15–26. <https://doi.org/10.1111/php.12377>
 22. Knies JL, Dang KK, Vision TJ, Hoffman NG, Swanstrom R, Burch CL (2008) Compensatory evolution in RNA secondary structures increases substitution rate variation among sites. *Mol Biol Evol* 25(8):1778–1787. <https://doi.org/10.1093/molbev/msn130>



Evolutionary Conservation of RNA Secondary Structure

Maria Beatriz Walter Costa

Abstract

Noncoding RNAs, ncRNAs, naturally fold into structures, which allow them to perform their functions in the cell. Evolutionarily close species share structures and functions. This occurs because of shared selective pressures, resulting in conserved groups. Previous efforts in finding functional RNAs have been made in detecting conserved structures in genomes or alignments. It may occur that, within a conserved group, species-specific structures arise after species split due to positive selection. Detecting positive selection in ncRNAs is a hard problem in biology as well as bioinformatics. To detect positive selection, one should find species-specific structures within a conserved set. This chapter provides protocols to detect and analyze positive selection in ncRNA structures with the SSS-test and other free software.

Key words Noncoding RNA, Secondary structure, Evolution, Conservation, Positive selection

1 Introduction

Conservation of noncoding RNA structures is a fundamental topic in evolution [1–5]. A conserved set of ncRNAs is a result of negative selection on a set of species. Positive, or adaptive, selection causes a branch (or species) to acquire species-specific functions. In this case, the positively selected molecule distinguishes itself from its orthologs [6]. Detecting positive selection helps us understand speciation and is therefore extremely valuable. It is however a hard problem in bioinformatics.

In this chapter, we will discuss evolution of ncRNA structures with focus on positive selection. The introduction contains theoretical principles of structure prediction, structure conservation, detection of positive selection, and visualization. Afterward, protocols will be provided to detect and analyze positive selection with the SSS-test and other free software.

This chapter is targeted to researchers from a biological background who wish to study ncRNA evolution. The SSS-test and many other tools for studying ncRNAs were developed for Unix platforms. For this reason, a Unix platform is required, such as

Linux or MacOS operating systems. Alternatively, a Windows virtual machine for a Linux system could be used. Advanced skills in programming are however not required.

1.1 Structure Prediction

The structure of a ncRNA is key to its function. They naturally fold into 3D structures in the cell, which are extremely difficult to predict. Fortunately, approximations in the form of 2D, or secondary structures, can be accurately predicted from the primary sequence. The folding occurs through a series of chemical interactions of base pairs: A-U, G-C, and wobble G-U. Each stacked base pair contributes with energy to the structure. Neighboring base pairings also form structural motifs, such as hairpins, stems, bulges, and loops. The idea behind most prediction software is to find the optimal combination of base pairs so that the energy of the whole structure is minimized and the structure is the most stable [7].

There are two main approaches to this: thermodynamics and statistics. Thermodynamic methods use pre-calculated tables of free-energy parameters derived from physical experiments of RNA melting [8]. Statistical methods train their parameters by building probabilistic models from closely related families of ncRNAs [9]. An advanced technique of computer science named dynamic programming is used for finding the optimal structure by recursive reconstruction.

The stability of the structure is described in entropic terms by the uncertainty of prediction, which depends on the number of possible structures of the ensemble. The uncertainty is calculated by the partition function Q , defined by:

$$Q = \sum e^{-E(s)/RT}$$

where $s \in \Omega$ is each structure s that belongs to the ensemble Ω , $E(s)$ is the energy of the structure, R is the gas constant, and T is the absolute temperature. The equilibrium partition function of a base pair i,j is defined by:

$$P_{i,j} = \frac{\sum e^{-E(s)/RT}}{Q}$$

where $s \in \Omega$ is a sequence that contains the base pair i,j [10]. As results of the partition function, we get probabilities of base pairs. The more probable they are, the more stable the structure will be and less fluctuation it will have in the ensemble. This can be intuitively seen through visualizations of secondary structures.

The minimum free energy (MFE) is the optimal and most stable secondary structure (Fig. 1). The centroid structure (Fig. 1) better represents the whole ensemble, because it is formed by the base pairs that are more likely to occur in all possible structures. Visually analyzing the secondary structure of the centroid can yield valuable clues as to how stable the ensemble actually is. Base pairings are often represented in color code, indicating probabilities (Fig. 1).

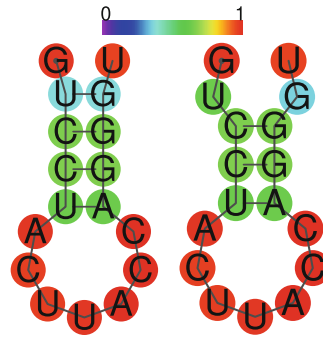


Fig. 1 Secondary structures of a theoretical hairpin. The MFE structure (left) is the most stable structure. The centroid structure (right) is composed of the base pairs that are more likely to occur in all possible structures. In addition to the structures, individual bases are color-coded according to their probability of being paired with another base or of being completely unpaired. The color code is depicted on the top, with lower probabilities in purple and blue, intermediate probabilities in green, and higher probabilities in orange and red. As an example, paired bases in green have around ~50% probability of being paired, while unpaired bases in red have around ~100% probability of being unpaired

As known from the biochemical properties of base pairings, G-C pairs are formed by three hydrogen bonds, while A-U and G-U pairs by two bonds. The energy required to break a bond depends not only on these intrinsic properties but also on the neighboring pairs or lack thereof. This algorithm is called nearest neighbor model, which is built into the Turner model for structure prediction [8, 11]. This model also contains rule sets for motifs such as helices and loops, which are coded into the Nearest Neighbor Database [12, 13].

The Turner model (as described in [12]) is implemented in the Vienna RNA package 2.0 [7], a package for analysis of ncRNA structures. RNAfold is its main program which computes the partition function, the MFE, and centroid secondary structures from an ncRNA primary sequence. One informative optional output of RNAfold is a PostScript file containing the base pair probabilities of the given sequence. This file can be viewed as text or image (Fig. 2). The top right part of the figure shows the partition function and the bottom left the MFE base pairing, which corresponds to Fig. 2.

1.2 Tools for Studying ncRNAs

RNAfold applies global folding. This means that the entire sequence of the ncRNA is considered in the structure prediction. This approach is ideal for small ncRNAs (<200 nucleotides, nts), especially since the majority of the base pair interactions occur within a span of 150–200 nts [14]. Long ncRNAs, lncRNAs for short (>200 nt), can span thousands of nts and do not always use their structures for function [15–17]. In this case, it is more

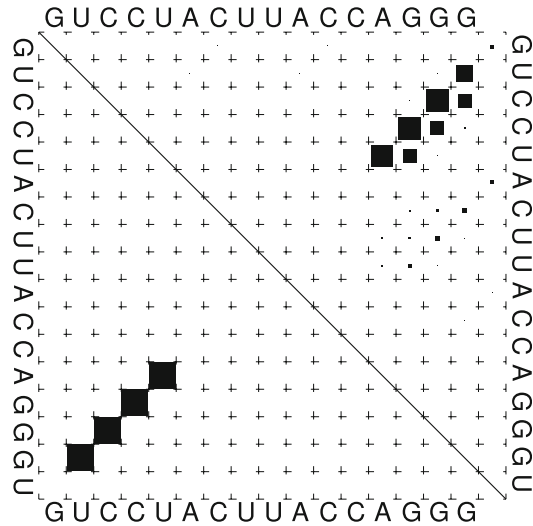


Fig. 2 Dot plot of a theoretical hairpin and loop structure. The rows and columns indicate the input sequence. The match between a row and a column represents the probability of base pair i,j , with bigger squares indicating larger probabilities. The ensemble of possible base pairs is represented in the top right corner, while the MFE base pairing is represented in the bottom left corner

sensible to consider local approaches. RNALfold, also part of the Vienna RNA package, uses RNAfold to scan sequences and output local stable structures.

RNAalifold is another tool of the Vienna package that is useful when studying RNA evolution [18]. It searches for consensus structures in FASTA alignments. In addition, a difficult problem in RNA biology is to compare structures of different lengths. RNAforester can do that by using a tree alignment model [19].

RNAsnp calculates the structural impact of a single nucleotide polymorphism, SNP for short, in a ncRNA structure [20]. Firstly, RNAsnp uses RNAfold internally to fold the original and the SNP-containing sequences. It then compares the partition functions of the two structures and afterwards compares the result with a precomputed database. Lastly, it outputs p -values that indicate structural impacts.

In addition to probing structures directly, studying the primary sequence of ncRNAs is also paramount for understanding their evolution. Muscle is a multiple sequence aligner [21] that produces alignments from multi-FASTA files. With a multiple alignment, it is possible to detect species-specific substitutions as well as species-specific indels (insertions and deletions).

1.3 Evolutionary Conservation and Orthology Annotation

Orthology annotation is key to analyzing ncRNA evolution. The principle behind it is finding similar structures in the group of studied species. Similar structures indicate common selective pressures. This problem has been extensively studied in RNA bioinformatics, and there are many tools that perform well when detecting conservation in RNAs, such as RNAz [2], SSISSz [22], EvoFold [3], CMfinder [4], AlifoldZ [23], and qrna [24], among others.

Although sequence similarity, e.g., BLAST [25], is still used for RNA orthology in the computational biology community, BLAST is not ideal for ncRNAs due to their weaker sequence conservation. Evolution of ncRNAs is rather constrained by structure. Different sequences can fold into the same structures. This principle is used in Infernal [26], which calculates covariance models based on structure similarity. It then uses these models to find new orthologs. RNAz [2] is another alternative for orthology and works on species alignments. Importantly, these tools are adequate for small RNAs (<200 bp).

Orthology of lncRNAs has different principles, due to the fact that only some of their sequences fold into stable structures. Their sequences are poorly conserved, making BLAST a suboptimal choice for orthology annotation. Their splice sites however are well conserved [27, 28], which can be used for this purpose. The SpliceMap tool [29] annotates ortholog splice sites in different species given a multiple alignment and a list of lncRNAs of a reference species. The output splice sites can then be fed into a second tool, buildOrthologs (unpublished), which reconstructs complete ortholog transcripts.

Orthologs of ncRNAs in a multi-FASTA file constitute the input to the SSS-test, which outputs selection scores. The matter of noncoding orthology is out of scope for the test itself and as such should be handled previously by the user.

1.4 Positive Selection Annotation

Genes under positive (or adaptive) selection indicate specialized function in a species of interest. The main idea to detect positive selection is to compare the observed changes with the expectation of neutral evolution. The K_a/K_s , or dN/dS , test [30] implements this idea and has been routinely used for protein coding genes (PCGs):

$$\frac{K_a}{K_s} = \frac{\frac{\text{non-synonymous-substitutions}}{\text{non-synonymous-sites}}}{\frac{\text{synonymous-substitutions}}{\text{synonymous-sites}}}$$

If the ratio of K_a is larger than K_s , the value is larger than 1.0 and positive selection is more likely. Conversely, if K_a is smaller than K_s , negative selection is more likely. Importantly, a non-synonymous substitution in the PCG will change the protein structure because it changes the amino acid of the translated protein. Conversely, a synonymous substitution will not change the

translated protein. It can be said that the effect of the combined substitutions in the protein can be inferred from the effect of the individual substitutions in the PCG.

Adapting this test to ncRNAs is in principle possible but yields spurious results [6] and should therefore not be pursued. Although there are substitutions that either change or maintain the structure, which could be deemed synonymous/non-synonymous, it is not trivial to systematically predict these in a noncoding context. Differently from proteins, the effect of the combined changes in the ncRNA cannot be easily inferred from the effect of the individual changes in the noncoding gene. In addition, insertions and deletions (indels) also impact the structure and should therefore be included in the calculations of positive selection.

The SSS-test is the first attempt, to my knowledge, that predicts positive selection in ncRNAs [6]. The tool receives a multi-FASTA as input and outputs selection scores for each species (Fig. 3). The main steps of the pipeline are described next. Note that all of them are already implemented in the SSS-test script, so the user does not have to concern oneself about any implementation details.

1.4.1 Species-Specific Changes

If the input given by the user is already aligned, it is submitted directly to a module that detects species-specific changes. Otherwise, muscle is used for the alignment. Species-specific changes can be either substitutions or indels. These two types will be processed separately in the pipeline.

To detect species-specific substitutions, each column of the alignment will be considered separately. Every base will be counted, and the most frequent one will be considered the “dominant base” if it occurs more frequently than a predefined threshold (Fig. 4). A base that differs from the dominant one will be classified as a species-specific substitution. Importantly, the species-specific substitution will be disconsidered if it forms a compensatory base pairing in the species structure, when compared to the consensus.

To detect species-specific indels, a two-way alignment will be considered. One of the two sequences is from the analyzed species and the other from a consensus produced by RNAalifold. This consensus contains all species except the one being analyzed, in order to exclude any bias from it. A consecutive sequence of gaps will be classified as a deletion if the species contains gaps while the consensus contains bases. A consecutive sequence of bases will be classified as an insertion if the species contains bases while the consensus contains gaps.

RNA_{snp}

The structural impact of each substitution is calculated with RNA_{snp}. The p -values calculated by RNA_{snp} indicate the structural impact of the substitution. High p -values indicate low

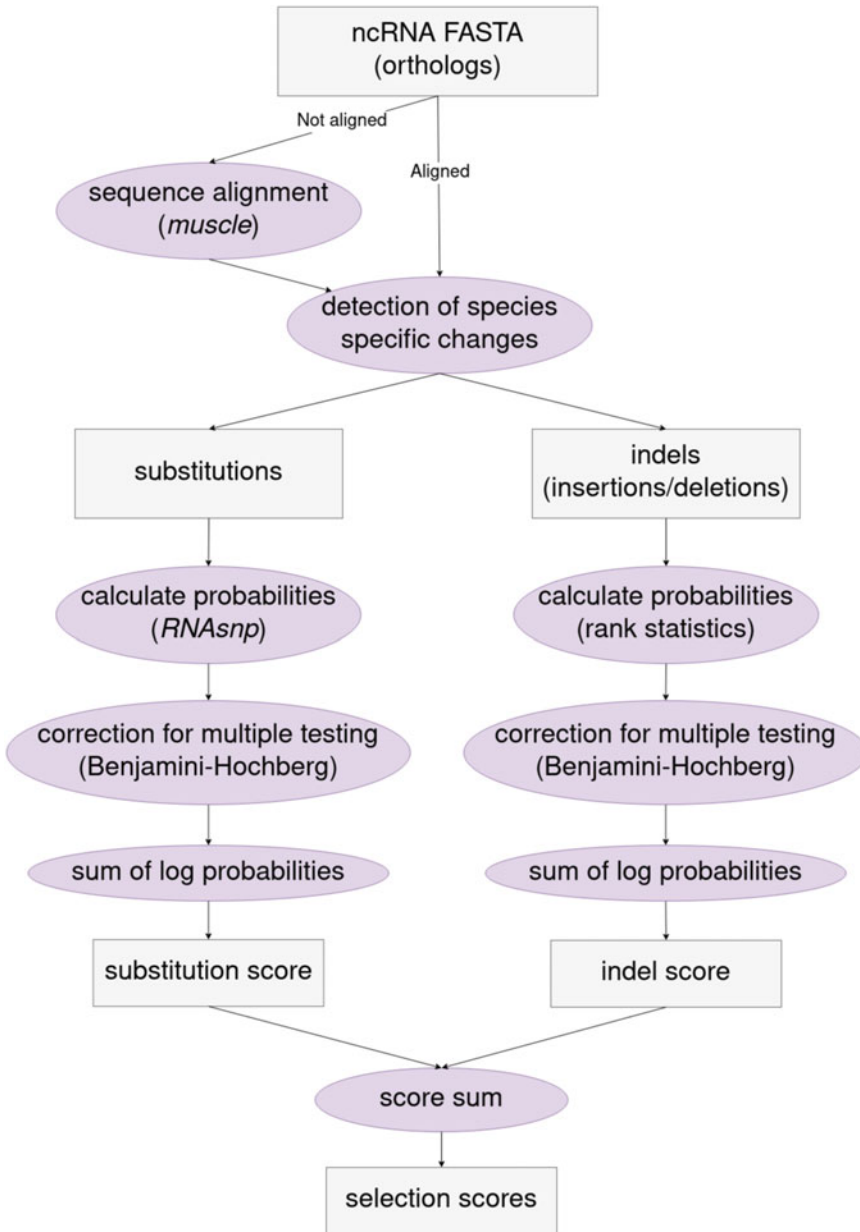


Fig. 3 The pipeline of the SSS-test

structural impact and vice-versa. Due to the multiple observation issue, all p -values are corrected using the Benjamini-Hochberg test [31]. Afterward, the corrected log p -values are summed, yielding a substitution score.

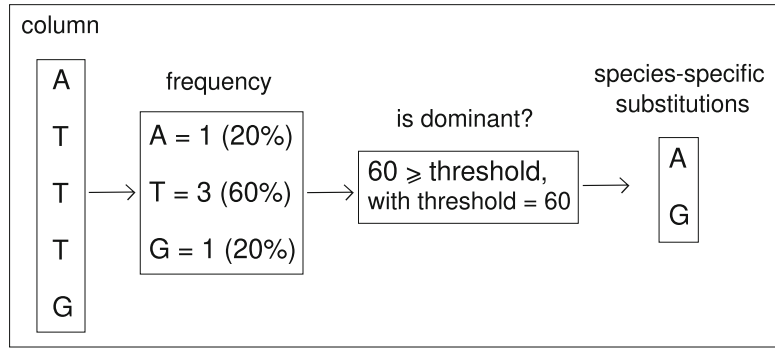


Fig. 4 Graphical representation of the SSS-test module to detect species-specific substitutions. The input of the module is a multiple sequence alignment, and each column is probed individually. First, the frequency of each base is calculated. Afterward, the frequency of the most frequent base is compared to a threshold to decide if it is a “dominant base.” If so, species-specific substitutions are detected whenever a base differs from the dominant one. If there is no “dominant base” in the column, no base is considered to be species-specific

1.4.2 Rank Statistics

The problem of indel impact on ncRNA structures is still open, since there are currently no accurate models for it. In the SSS-test, rank statistics were used to produce a *p*-value for an observed indel that indicates its structural impact.

For that, the distance between the structures of the species and the family’s consensus is first calculated. Afterward, it is calculated how likely it is to observe this indel within a neutral model. For that, the consensus ncRNA is mutated all along its sequence to obtain deletions of the same size of the observed indel. The structural distances are then calculated between the mutated sequence with and without the constraint of the original structure. All these calculations yield a series of values that form the ranks. If the observed indel occurs in the higher ranks (unlikely indels), it receives a low *p*-value. If the observed indel occurs in the lower ranks (likely indels), it receives a high *p*-value. The structural distances are calculated with RNAforester [32].

As for the substitutions, all indel *p*-values are corrected for multiple testing. Afterward, the corrected log *p*-values are also summed, yielding an indel score.

1.4.3 Selection Scores

To produce a selection score for each of the input species, the substitution and indel scores of the species are summed using the formula:

$$\text{score}_{\text{selection}} = 2 \times \text{score}_{\text{substitution}} + \text{score}_{\text{indel}}$$

The substitution score is given a weight of 2, while the indel score is given a weight of 1. This is an heuristic decision based on the knowledge that the calculation of the structural impact of the substitution is grounded in a well-established model, while the indel impact still lacks a stronger model.

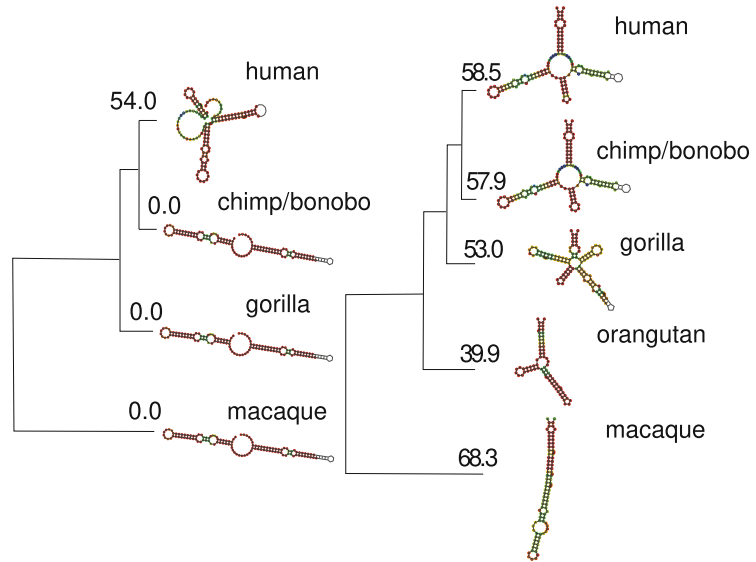


Fig. 5 Lowly diverged ncRNA family, left, and highly diverged ncRNA family, right (adapted from the supplemental file of [6]). In each panel, the primate phylogenetic tree is shown on the left along with the divergence score of each species in relation to the family's consensus. On the right side of each panel, the species structures are shown. The family divergence score is the median species score: $d = 0.0$ (left) and $d = 57.9$ (right)

Finally, the species score will inform the user about how unexpected the detected changes are, which is a measure of positive selection.

1.5 Family Divergence

Positive selection is only assumed to occur in a species if the ncRNA family went through a high level of negative selection beforehand. Therefore, this type of analysis is only appropriate in families that have low structural divergence (Fig. 5). The SSS-test outputs a structural divergence score to help the user filter the families. This score is based on the comparison of the partition functions of the species and the family's consensus (Fig. 5). To choose an appropriate cutoff, refer to the protocols Sect. 2.7.

1.6 Evolutionary History of a Structure

After identifying a structure that is under positive selection, it is possible to study its evolutionary process through time in a detailed manner using dynamic programming [33]. HAR1 is a perfect example for that, since it has accumulated 18 species-specific substitutions in only 1 lineage, human, and remained very well conserved in other species [34]. This is the fastest-evolving region in the human genome [34]. In addition, HAR1 is also interesting from a biological perspective because of its human structure, which differs from the other species [35–37]. Although these topics

are still under investigation, it seems that HARI is under positive selection [33] and has an important role in brain development and that its 118 bp local structure is key to its function [34, 35].

The mutationOrder software [33] was applied to HARI to study its evolutionary history. That has been done by analyzing all 0.64×10^{16} possible combinations of order of substitutions (mutations) from the last common ancestor from human and chimpanzee until the current human version. From these results, it could be concluded that the current human structure evolved to be more stable and is likely under positive selection in humans [33]. As exemplified by HARI, studying the evolutionary history of a non-coding structure can yield valuable information.

1.7 Visualization of Base Pair Probabilities

A secondary structure of a ncRNA, such as the MFE, can be easily visualized in a 2D form, as shown in Fig. 1. However, as discussed in Sect. 1.1, a ncRNA does not fold into one unique structure but rather fluctuates between some stable structures. Therefore, analyzing the MFE alone does not yield the complete picture or inform the viewer of how stable a structure is. Traditionally, researchers turn to dot plots (Fig. 2) to get a more comprehensive picture of base pair probability.

Although very informative, dot plots are not intuitive, especially for researchers from biological fields. A more intuitive alternative to dot plots are circular plots that show the MFE as well as all other possible base pairs (Fig. 6). The CS²BP²-Plot [38] is an easy-to-use tool that allows for an intuitive visualization of the evolution of a structure (see [33] for the HARI evolution), as well as the comparison of two or more structures [38]. The tool also allows for comparison of structures of different lengths, which is an improvement over other comparative tools, such as diffRNABow [39].

2 Methods

2.1 Software Installation

The SSS-test is a free software, available at the link <https://github.com/waltercostamb/SSS-test>, along with its help and usage pages. It was developed for Linux but runs on MacOS systems as well. The RNAfold program from the Vienna RNA package is used within the SSS-test to predict secondary structures and their partition functions. In addition, the SSS-test uses RNAalifold and RNAforester, also part of the Vienna package, as well as the muscle aligner and three Perl packages. To install the required software, follow the instructions of the links below on your Unix system.

RNAasp: <http://rth.dk/resources/rnasnp/software>

Muscle aligner: <http://www.drive5.com/muscle>

Vienna RNA package: <https://www.tbi.univie.ac.at/RNA/#download>

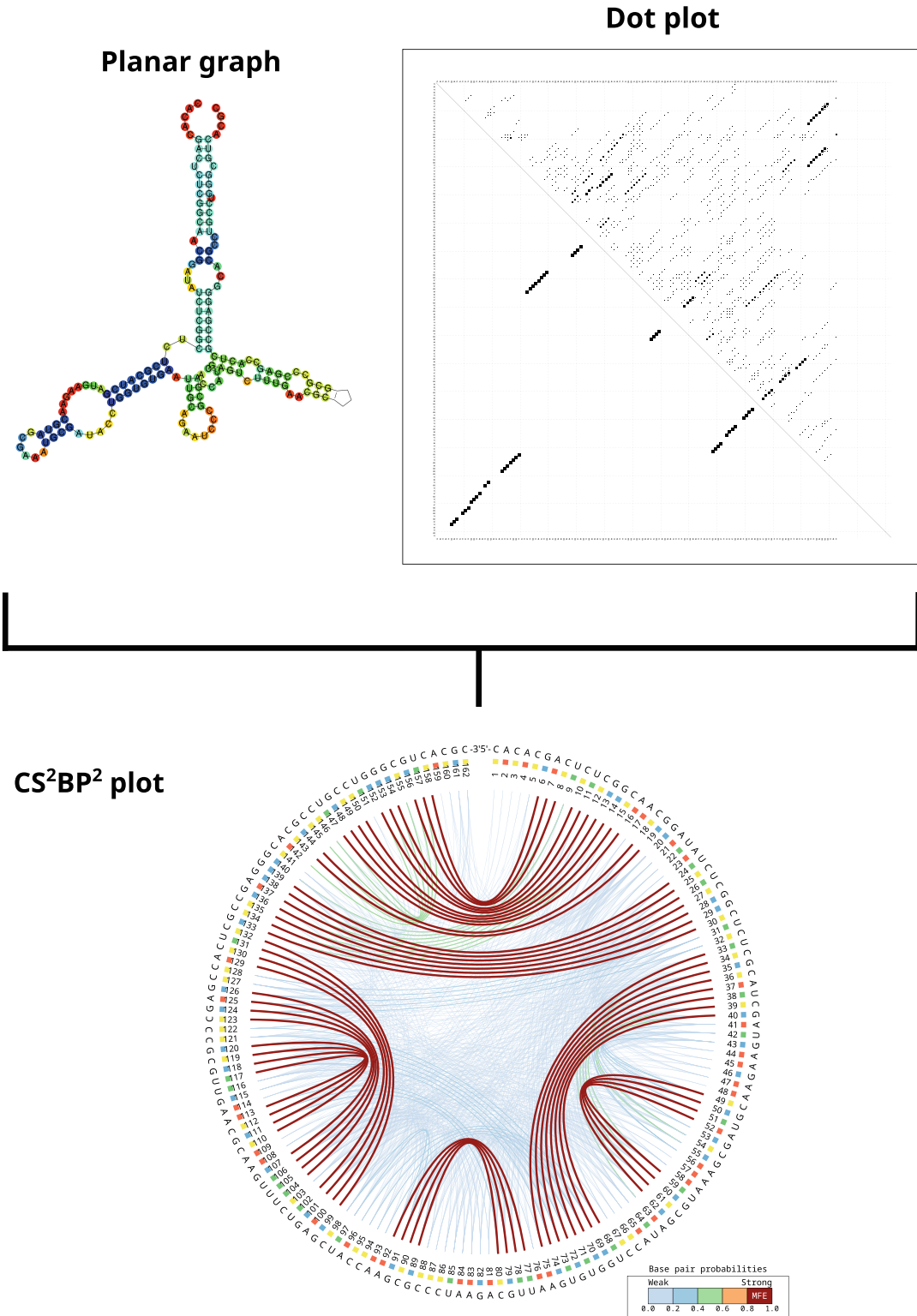


Fig. 6 Different visualization techniques for ncRNA structures. The CS²BP²-Plot on the right combines information of both the planar graph and dot plot (adapted from [38]). The most stable MFE base pairs are colored red, while the other base pair probabilities are colored in blue, red, green, or yellow, according to their values

Bio::AlignIO from cpan: <https://metacpan.org/pod/Bio::AlignIO>

fasconvert from cpan: <http://search.cpan.org/dist/FAST/bin/fasconvert>

Statistics::R from cpan: <http://search.cpan.org/~gmpassos/Statistics-R-0.02/lib/Statistics/R.pm>

After installation, if necessary, copy the lib/distParam directory from the RNAsnp sources over to the RNAsnp installation location. You may also need to export the path to RNAsnp installation location:

```
export RNASNPPATH=/path-to/RNAsnp-1.2
```

In addition, make sure that all binaries of the required software are in the bash path, with /path-to/, /RNAsnp/ /ViennaRNA/, etc. replaced appropriately.

```
export PATH = $PATH:/path-to/RNAsnp/bin:/path-to/ViennaRNA/bin:/path-to/ViennaRNA/\
```

```
share/ViennaRNA/bin:/path-to/muscle/bin
```

Note that the command above corresponds to one command line. The backslash symbol indicates that the line continues below.

A common error when running the SSS-test is when the bash does not find the script relplot.pl, from the Vienna package. To solve this, simply export its path to the bash \$PATH variable. Note that relplot.pl is provided in a subfolder of the share folder of the Vienna package.

An easier alternative that does not require you to install any software is to download the SSS-test bundle directly available at <http://www.bioinf.uni-leipzig.de/Software/SSS-test/>.

After download, the software can be used directly.

2.2 Usage of the SSS-Test

The help page of the SSS-test can be accessed by typing the following command line:

```
./SSS-test
```

The general command line of the SSS-test is:

```
./SSS-test -i FOLDER/FILE -f FORMAT -s STRUCTURE
```


The parameter `-i` takes as argument the subfolder and input file. The parameter `-f` takes as argument the format of the input file, which can be either `fasta` (not previously aligned) or `aligned` (if previously aligned). The parameter `-s` takes as argument either `Yes` or `No` depending on if the user wishes the structure files to be saved in the subfolder `FOLDER/FILE_structures`.

A valid command line of the SSS-test is:

```
./SSS-test -i example/input.fasta -f fasta -s Yes
```

In the example, the input file is `input.fasta`, which is located in subfolder: `example/`. It is a requirement of the SSS-test that the input file should be located inside a subfolder as exemplified above. Note that, independently of being aligned or not, the input file should always be in the classic FASTA format, with a header marked by the symbol “>” in the first line and sequences in the subsequent lines. Space characters are not allowed in the header. An example of a valid input follows below:

```
>species1
ACACAGAGGGCCCTCTCTGCCTCTGGCCACCACAGCCCCTAGGCCAGGCATGGGTATT-
TATTCTTAGGTA TGT TGC TTTT AAGAAGATGTAATCAGCATCTT-
GAGCCGGGCCTCCCTTTGTGA
>species2
ACACAGAGGGCCCTCTCTGCCTCTGGCCACCACAGCCCCTAGGCCAGGCATGGGTATT-
TATTCTTAGGTA TGT TGC TTTT AAGAAGCTGTAATCAGCATCTT-
GAGCCGGGCCTCCCTTTGTGA
>species3
ACACAGAGGGCCCTCTCTGCCTCTGGCCACCACAGCCCCTAGGCCAGGCATGGGTATT-
TATTCTTAGGTA TGT TGC TTTT AAGAAGATGTAATCAGCATCTT-
GAGCCGGGCCTCCCTTTGTGA
```

The parameter `-s` in the valid command line above received a `Yes` as an argument, indicating that the user wanted the secondary structures to be saved. You can see those files with the command below:

```
ls -lh example/input_structures/
```

Inside this folder you will find six files for each input species. Three of them correspond to the species itself and three correspond to the consensus of the group without the species. Those are:

- Centroid secondary structure for the species (FILE-SPECIES-centroid_rss.ps)
- MFE secondary structure for the species (FILE-SPECIES-MFE_rss.ps)

- Dot plot for the species (FILE-SPECIES_dp.ps)
- Centroid secondary structure for the consensus of the group, without the species (FILE-SPECIES-consensus-centroid_rss.ps)
- MFE secondary structure for the consensus of the group, without the species (FILE-SPECIES-consensus-MFE_rss.ps)
- Dot plot for the consensus of the group, without the species (FILE-SPECIES-consensus_dp.ps)

The three parameters detailed above, `-i`, `-f`, and `-s`, are mandatory. The parameter `-d` is optional and can be changed, according to the user. It refers to the threshold in percent of the dominant base and can take any value from 0 to 100. This threshold is used when detecting species-specific substitutions (Fig. 4). As detailed in Sect. 1.4, a species-specific substitution is detected when it differs from the column's dominant base, which has a frequency equal to or higher than the threshold. If not indicated, it will receive a default value of 60.

The lines below correspond to a general command and a valid command changing the threshold to 50, as an example.

```
./SSS-test -i FOLDER/FILE -f FORMAT -s STRUCTURE -d DOMINANT_
BASE_THRESHOLD
```

```
./SSS-test -i example/input.fasta -f fasta -s Yes -d 50
```

In this example, if your input has six species, the SSS-test will already consider a dominant base if it occurs in at least three species. Say, for a substitution s with $s_1 = \text{"A,"}$ $s_2 = \text{"A,"}$ $s_3 = \text{"A,"}$ $s_4 = \text{"T,"}$ $s_5 = \text{"T,"}$ and $s_6 = \text{"G,"}$ the dominant base will be "A" and s_4 , s_5 , and s_6 will be considered as species-specific substitutions. Unless you have specific reasons for changing this threshold, you should use the default.

Once you run the SSS-test for the first time for any multi-FASTA file, intermediate files will be created within the input subfolder. The next time you run the test again for the same file, the calculation of the scores will be considerably faster. Note that you may receive errors from the test in case you make modifications in the input file after the first calculations and try to run it again. In this case, remove all intermediate files produced by the test and run it one more time.

The main output of the SSS-test is a file with the species scores. You will find it in the subfolder where the input file is located. The contents of this file will also be output in the terminal. If the input name is `input.fasta`, the name of the score file will be `input.sss`.

2.3 Usage of the Local Structure Pipeline

The SSS-test uses RNAfold internally to predict global folding of structures. This approach is appropriate for small RNAs. Long ncRNAs, lncRNAs, have a different dynamic, and do not always fold into stable structures. In this case, it is more sensible to consider local instead of global structures. If you wish to study selection in lncRNAs, it is advisable to first use the local pipeline to find conserved local structures and then submit these to the SSS-test.

The help page of the local structure pipeline can be accessed by typing the following command line:

```
./local-structure-pipeline
```

The general command line and a valid example are:

```
./local-structure-pipeline -i FOLDER/FILE -f FORMAT -o OUTPUT_FOLDER
```

```
./local-structure-pipeline -i example/family.fasta -f fasta -o local_family
```

As for the SSS-test, the input file is indicated by parameter `-i`. The file in the example above is `family.fasta`, which is located in subfolder: `example/`. The requirements for the input are the same, including the FASTA contents. Also like the SSS-test, the parameter `-f` indicates the format of the input file, which can be either `fasta` or `aligned`. The last parameter is `-o`, which indicates where the pipeline should save the output files. If the folder does not exist, the pipeline will create it.

The pipeline will search for conserved local structures and will output all their FASTA files in the output folder. These FASTA files are ready to be input to the SSS-test. In addition to the FASTA files, you will also find in the output folder a file with species positions (`OUTPUT_FOLDER/FILE-local_positions.txt`). This file follows the following format:

```
Block Gorilla Human Macaque Orangutan Pan
sub1 3-126 284-360 11-112 3-126 3-126
sub2 - 370-490 - - -
sub3 - 537-622 - - -
sub4 605-674 840-911 686-764 - 711-853
sub5 - 1072-1181 - - 981-1090
sub6 - 1188-1248 - - 1097-1157
```

The block column refers to the local structure. For example, `sub1` refers to file `OUTPUT_FOLDER/FILE_sub1.fasta`. The following columns refer to each of the input species and the positions of their local sequences. For example, `sub1` of Gorilla is located in the gorilla lncRNA sequence between the third and the 126th base.

2.4 Running the SSS-Test with an Example

After installing or downloading the SSS-test to your computer, you will find folder `examples/`, which contains two FASTA files: `SIX3-AS1sub10.fa` and `H19X.fa`. The first file is a local structure block from the lncRNA SIX3-AS1. This block can be submitted to the SSS-test directly. The second file, `H19X.fa`, contains orthologs of a long ncRNA, which should be submitted first to the local structure pipeline. After finding conserved local structure blocks with the pipeline, these can be submitted to the SSS-test.

To run the SSS-test for the local block 10 of the SIX3-AS1 lncRNA, use the following command line:

```
./SSS-test -i example/SIX3_AS1sub10.fa -f fasta -s Yes
```

You will receive in the terminal notification messages of the test along with the selection scores. In addition, an output file will be produced, `examples/SIX3_AS1sub10.sss`, which will contain the selection scores of each input species. Intermediate files of two types will also be produced for the input species. The first file is an RNAsnp report, obtained with the command below:

```
RNAsnp --pvalue1 = 10.0 -pvalue2 = 10.0 -f SIX3_AS1sub10-  
consensus_without_the\_ SPECIES.alg -m 3
```

The remaining intermediate files are indel reports from the gap modeling pipeline that is used internally in the SSS-test. These are named `FILE-SPECIES-i.indel`, with the `i` being the indel size. An example of a file of this type of file with `i = 2` follows:

```
RNA_length gap_start gap_length structural_distance  
10 1 2 15  
10 2 2 15  
10 3 2 12  
10 4 2 0  
10 5 2 0  
10 6 2 0  
10 7 2 0  
10 8 2 12  
10 9 2 15  
10 10 2 15
```

This file summarizes the structural impact of a gap `i = 2`, indicated in the third column, in an example sequence of length = 10, as indicated in the first column. The gap `i` is applied from the first to the last base, as indicated in the second column, and its structural impact as calculated by RNAforester is indicated in the fourth column.

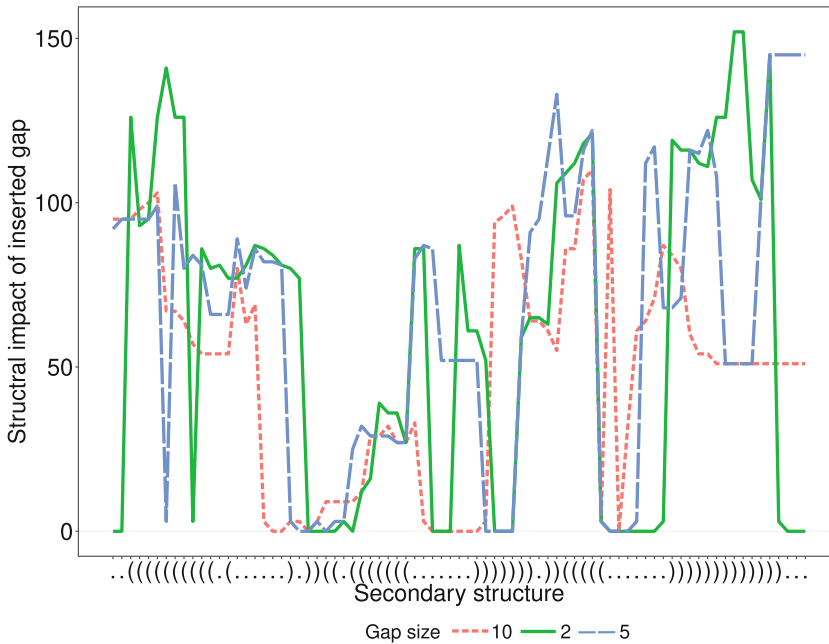


Fig. 7 Structural impact of differently sized gaps inserted in a transfer RNA, tRNA, with a length of 78 nucleotides (adapted from the supplemental file of [6]). The secondary structure is given as a reference on the x axis in dot bracket format, with dots representing unpaired bases and brackets paired bases. The structural impact was calculated with RNAforester

Generally, high impacts are observed whenever the gap is applied to a base paired position, while low to zero structural impacts are observed in unpaired positions (Fig. 7). Interestingly, the size of the gap does not seem to matter (Fig. 7). This is a recent topic in RNA biology that has not yet been fully covered. Future advances in this area will surely benefit the SSS-test and studies in positive selection in ncRNA structures.

The secondary structures are also output by the SSS-test and are located in the folder: `./examples/SIX3_AS1sub10_structures/`. To learn how to analyze and interpret the results, proceed to section “Interpreting Results.”

2.5 Running the Local Structure Pipeline with an Example

To measure structural selection locally in the H19X lncRNA, you should first calculate local structure blocks and then apply the SSS-test separately for each block. For calculating conserved blocks, use the command line below.

```
./local-structure-pipeline -i example/H19X.fa -f fasta -o
H19X_local_structures
```

The command will create a new folder, `H19X_local/`, with all the local structure blocks as well as a file indicating the sequence positions of each block. Now that you calculated the local blocks,

you can submit them to the SSS-test in the “not aligned” mode (fasta). If you want to calculate a block individually, say for local block 2, type:

```
./SSS-test -i H19X_local/H19X_sub1.fa -f fasta -s Yes
```

If you wish to run the SSS-test for all local blocks at once with a bash loop, type:

```
./loop_SSS-test.sh -i H19X_local/*fasta -f fasta -s Yes
```

2.6 Interpreting Results

After running the SSS-test, the results should be carefully considered. The following aspects should be taken into account: (i) the family divergence, (ii) the selection scores, and (iii) the secondary structures. The main output of the SSS-test is a .sss file, in which each line corresponds to an input species. The lines below refer to the .sss file of the example input and were slightly modified for better visualization.

```
seqID nr_ch sp_dist sp_len alg_len sc_ind sc_ch sp_sc famil-
y_div
Orang 0 0.0 123 123 0 0 0.0000 0.0
Human 1 33.4 123 123 0 12.1497 12.1497 0.0
Pan 0 0.0 123 123 0 0 0.0000 0.0
```

This file contains nine columns: (1) ID of the file and species; (2) number of species-specific substitutions; (3) structural distance between the species and the consensus; (4) length of the species sequence; (5) length of the alignment with all species; (6) indel score; (7) substitution score; (8) species selection score, considering both indel and substitutions score; and (9) structural divergence of the family (median value of column 3).

2.7 Analyzing and Filtering Families

Analyzing structural selection for a species is only sensible if its family is conserved. The family divergence score given in the output of the SSS-test (ninth column) indicates this context. Low values, e.g., ≤ 10.0 , indicate a very conserved family, while high values, e.g., > 45.0 , indicate a diverged family. You should consider your species set critically in this aspect and choose an empirical threshold. For that you can select families with different divergence scores and analyze them manually. As an example, for primate species, 12 families of lncRNAs (Fig. 8) were chosen with scores ranging from 0.0 to 65.0 (supplemental file of [6]). These were classified according to their visual profile (Table 1).

As a result of this visual analysis, a threshold of ≤ 10.0 was empirically chosen for conserved families. Note that primates are evolutionarily close. For another set of species, a different threshold

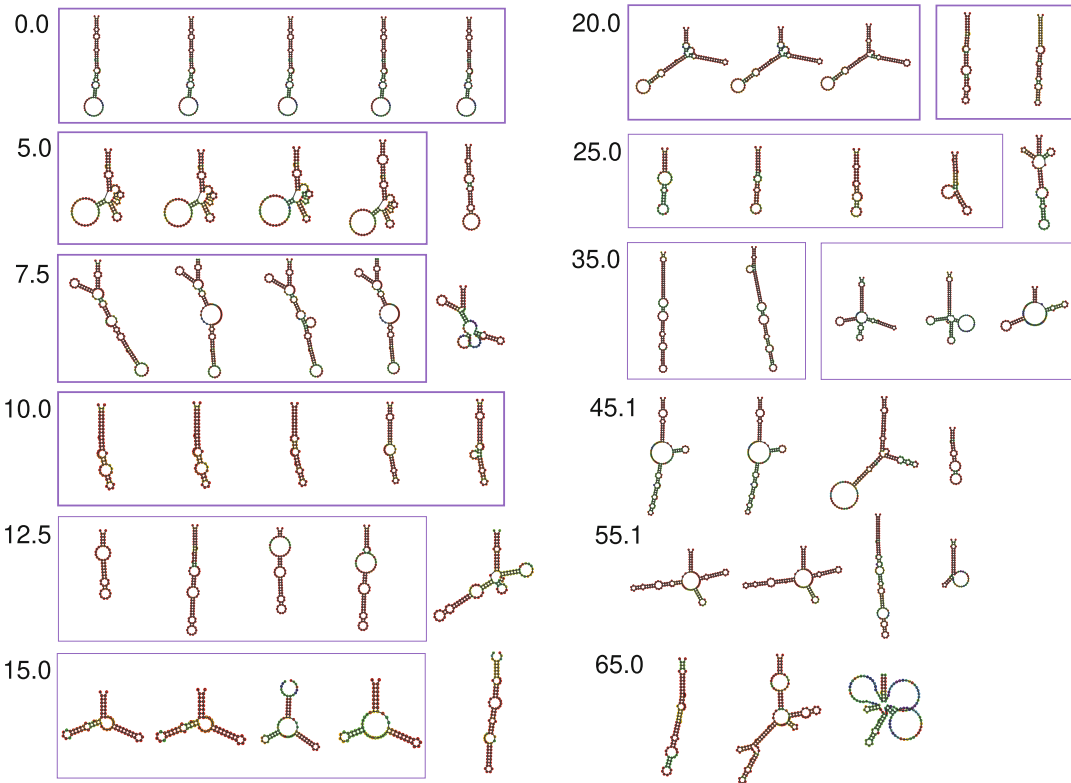


Fig. 8 Centroid structures of conserved local blocks of lncRNAs with increasing divergence scores (adapted from the supplemental file of [6]). Left panel: scores from 0.0 to 15.0; right panel: scores from 20.0 to 65.0. In each panel from left to right: human, chimpanzee/bonobo, orangutan, gorilla, and rhesus macaque. Visual structural trends are marked in purple rectangles, thicker rectangles refer to clear trends, and thinner rectangles refer to possible trends. Note that the values refer to the divergence scores of each conserved block

may be chosen. After choosing a threshold, you can filter results systematically.

2.8 Analyzing Selection Scores

After filtering the families by divergence, you should analyze the selection scores of each species. First, choose appropriate thresholds of negative and positive selection. For this you can follow a similar idea as for choosing a threshold of divergence score. As an example, also for primates, ten conserved blocks of lncRNAs (Fig. 9) were chosen with the human selection score ranging from 0.0 to 30.0 (supplemental file of [6]). These were classified according to their visual profile (Table 2).

In this case, thresholds of ≤ 3.0 and ≥ 10.0 were empirically chosen for negative and positive selection, respectively. After defining your thresholds, you can filter your species sequences systematically.

The final step of considering your results is to visually analyze the secondary structures and critically decide if they match the selection scores. If they do, they become candidates of positive selection and can proceed to the wet lab for in vitro analysis.

Table 1
Summary of the visual analysis of 12 conserved local blocks of lncRNAs

Family ID and local structure	Score	Visual profile
blastnMouse.CUFF.110475 (sub2)	0	1 clear trend
blastnMacaque.Locus 40302 (sub4)	5	1 clear trend
ENSG00000237166 (sub4)	7.5	1 clear trend
blastnPan.Locus 7625 (sub1)	10	1 clear trend
ENSG00000243012 (sub2)	12.5	1 possible trend
blastnOpossum.Locus 375118 (sub1)	15	1 possible trend
ENSG00000256802 (b-1)	20	2 clear trends
blastnMacaque.Locus 429229 (sub1)	25	1 possible trend
ENSG00000236466 (sub1)	35	2 possible trends
blastnPan.Locus 105878 (sub1)	45	No trend
ENSG00000226526 (sub3)	55	No trend
blastnPan.Locus 566 (sub1)	65	No trend

Adapted from the supplemental file of [6]

Score refers to the divergence score of the local block yielded by the SSS-test

2.9 Visual Analysis

The selection scores for SIX3_AS1sub11 are 12.2, 0.0, and 0.0 for human, orangutan, and chimpanzee/bonobo, respectively. This indicates positive selection in the human branch. As a visual interpretation of the centroid structures of the species, we see that the human species is more stable, especially considering the lower stem (Fig. 10). This stabilization could have occurred due to positive selection.

To get a more comprehensive view of the structures of each species, and to compare them directly, you can go to the CS²BP²-Plot webpage, <https://nrcmonsrv01.nrc.ca/cs2bp2plot/>, and input the FASTA sequences. You will obtain two circular plots (Fig. 11) and will be able to change the displayed species at the “Sequence” option on top. Secondary structures (planar graph diagrams) are available on the bottom along with arc and circular diagrams and dot bracket structures. A statistics summary and an alignment report are also available below. This information will help you to further characterize your positive selection candidates.

In our example, there is only one base substitution between the human and the other two sequences (base 87 from A to C, which is marked with red circles in Fig. 11). Since the sequences for

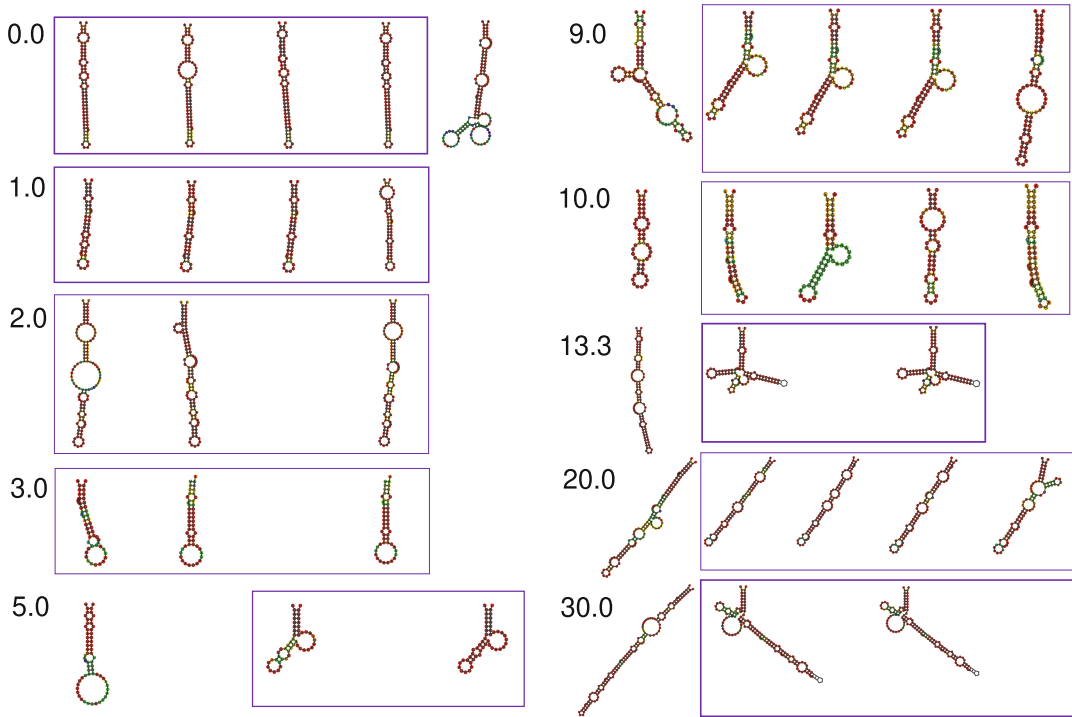


Fig. 9 Centroid structures of conserved local blocks of lncRNAs with increasing selection scores of the human structure (adapted from the supplemental file of [6]). Left panel: scores from 0.0 to 5.0; right panel: scores from 9.0 to 30.0. In each panel from left to right: human, chimpanzee/bonobo, orangutan, gorilla, and rhesus macaque. Visual structural trends are marked in purple rectangles, thicker rectangles refer to clear trends, and thinner rectangles refer to possible trends. Note that the values refer to the selection scores of the human structure only (the leftmost structure in each panel)

Table 2

Summary of the visual analysis of ten human structures in conserved lncRNA blocks

Family ID and local structure	Score	Visual profile
blastnMacaque.Locus 61692 (sub1)	0	Similar form and stability
ENSG00000224711 (sub5)	1	Similar form and stability
blastnMacaque.Locus 62244 (sub4)	2	Similar form and stability
blastnMacaque.Locus 473621 (sub6)	3	Similar form, higher stability
blastnPan.CUFF.296990 (sub7)	5	Slight different form, lower stability
blastnMacaque.Locus 474656 (sub2)	9	Different form, similar stability
Locus 193583 (subb3)	10	Shorter form, higher stability
ENSG00000227509 (sub7)	13.3	Different form, higher stability
blastnPan.Locus 17197 (sub1)	20	Longer form, lower stability
blastnMacaque.Locus 210980 (sub8)	30	Different form, higher stability

Adapted from the supplemental file of [6]

Score refers to the selection score of the human structure yielded by the SSS-test. The visual profile refers to form and stability of the human structure in comparison with its orthologs

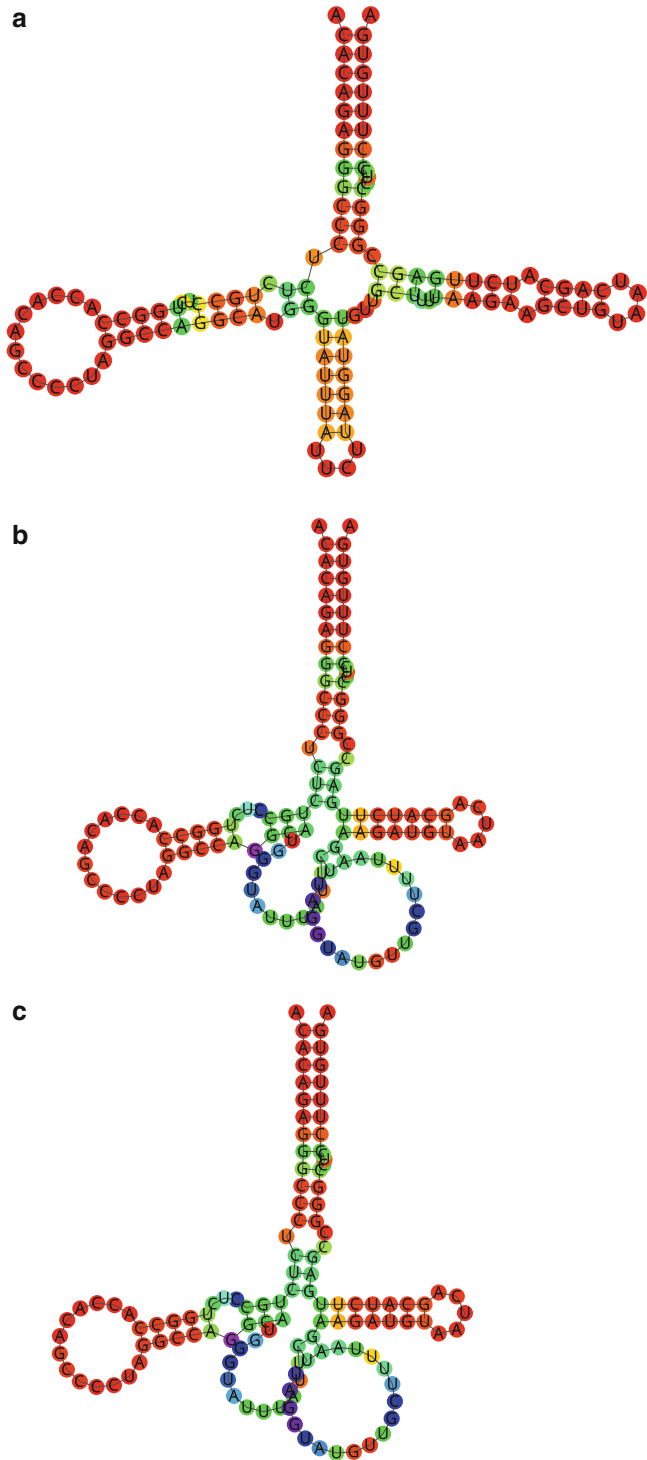


Fig. 10 Local structures of lncRNA SIX3-AS1sub10 of the following species: (a) human, (b) chimpanzee/bonobo, and (c) orangutan

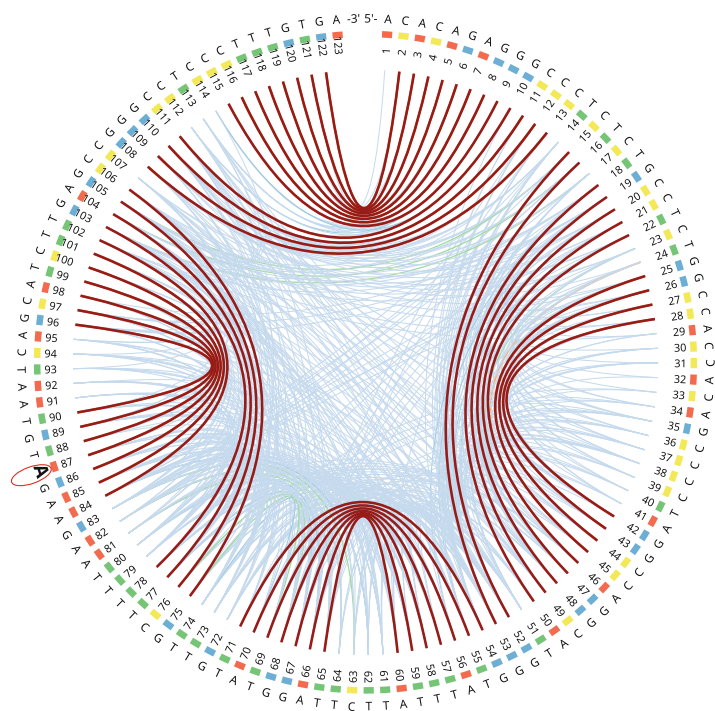
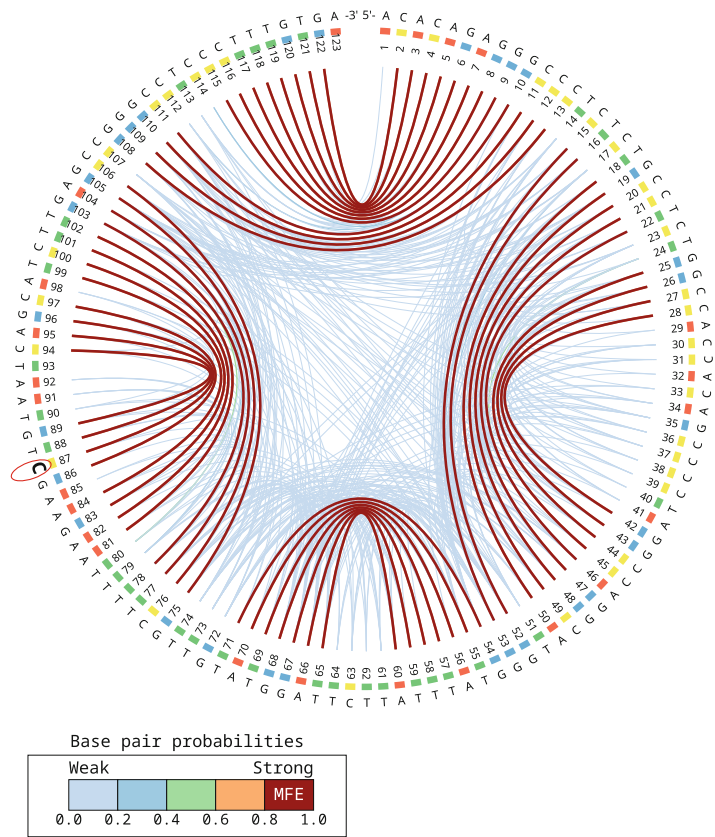


Fig. 11 Base pairing probabilities of local structures SIX3-AS1sub10 of human (top) and orangutan (bottom). The sequences match perfectly with the exception of base 87, which is marked with red circles. The MFE base pairings are depicted in red, while other base pairings are marked in other colors from orange to blue, depending on their probabilities

orangutan and chimpanzee/bonobo are the same, it is reasonable to assume that the human sequence acquired this substitution, A87C, after the split from these two species.

Was this substitution caused by positive selection in the human branch? The results of the SSS-test indicate so. However, to answer this question, more evidence should be collected. By visual analysis, it seems that the substitution stabilized the human structure (Fig. 11). This can be inferred by an absence of intermediately strong pairings in the human structure, depicted by orange and green, and their presence in the other species. Notice (i) the green connections in the orangutan and chimpanzee/pan structures between bases 15, 16, 17, and 103, 104, and 105; (ii) the orange connection between bases 24 and 46; and (iii) the green connections between bases 62–70 and 74–84 (Fig. 11). All these are absent in the human structure, strengthening the MFE and making it more likely. This corroborates the hypothesis that the human structure is under positive selection.

An analysis of SNPs in human populations would also give clues about the evolutionary history of this local structure. In addition, wet lab experiments would inform of functionality and differences between the species.

3 Notes

There are many aspects of ncRNA evolution that are not yet fully understood. Conservation of ncRNA structures, caused by negative selection, has been extensively researched [1–5], yielding many research tools and solid models. Positive selection, in turn, has not been as extensively researched. The SSS-test is the first tool (to my knowledge) that addresses this point and offers a practical solution [6].

It is important to keep in mind, though, that solid models are still missing for grounding the theory of positive selection in ncRNA structures. For this reason, the SSS-test uses heuristic decisions whenever a model is lacking, so that a selection score can be calculated. One of the most important aspects that will benefit from future research is the calculation of indel impact. Currently, the SSS-test uses rank statistics for calculating p -values that indicates structural impact.

Taking this into consideration, an empiric intuition of the analyst is valuable, so that he or she uses visual analysis together with the selection scores. The SSS-test is most powerful when used in a larger dataset that would be difficult to be analyzed manually. In this case, one can screen hundreds of thousands of families and obtain a smaller set that could be individually analyzed.

Acknowledgments

This chapter was developed with the support of the University of Leipzig and the Free State of Saxony. The author would like to acknowledge Christian Höner zu Siederdisen for reviewing the text and João Luiz Pacini Costa for the helpful discussions.

References

- Smith MA, Gesell T, Stadler PF, Mattick JS (2013) Widespread purifying selection on RNA structure in mammals. *Nucleic Acids Res* 41(17):8220–8236
- Gruber AR, Findeiß S, Washietl S, Hofacker IL, Stadler PF (2010) RNAz 2.0: improved noncoding RNA detection. *Bioinformatics* 15: 69–79
- Pedersen JS, Bejerano G, Siepel A, Rosenbloom K, Lindblad-Toh K, Lander ES, Kent J, Miller W, Haussler D (2006) Identification and classification of conserved RNA secondary structures in the human genome. *PLoS Comput Biol* 2(4):e33
- Yao Z, Weinberg Z, Ruzzo WL (2006) CMfinder – a covariance model based RNA motif finding algorithm. *Bioinformatics* 22:445–452
- Washietl S, Hofacker IL, Lukasser M, Hüttenhofer A, Stadler PF (2005) Mapping of conserved RNA secondary structures predicts thousands of functional noncoding RNAs in the human genome. *Nat Biotechnol* 23(11):1383
- Walter Costa MB, Höner zu Siederdisen C, Dunjić M, Stadler PF, Nowick K (2019) SSS-test: a novel test for detecting positive selection on RNA secondary structure. *BMC Bioinformatics* 20(1):151
- Lorenz R, Bernhart SH, Höner zu Siederdisen C, Tafer H, Flamm C, Stadler PF, Hofacker IL (2011) ViennaRNA Package 2.0. *Algorithms Mol Biol* 6(1):26
- Mathews DH, Turner DH (2006) Prediction of RNA secondary structure by free energy minimization. *Curr Opin Struct Biol* 16(3): 270–278
- Rivas E, Lang R, Eddy SR (2012) A range of complex probabilistic models for RNA secondary structure prediction that includes the nearest-neighbor model and more. *RNA* 18(2):193–212
- McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers* 29(6–7):1105–1119
- Tinoco I, Borer PN, Dengler B, Levine MD, Uhlenbeck OC, Crothers DM, Gralla J (1973) Improved estimation of secondary structure in ribonucleic acids. *Nature New Biology* 246(150):40
- Mathews DH, Disney MD, Childs JL, Schroeder SJ, Zuker M, Turner DH (2004) Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proc Natl Acad Sci U S A* 101(19):7287–7292
- Turner DH, Mathews DH (2009) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res* 38(suppl_1): D280–D282
- Lange SJ, Maticzka D, Möhl M, Gagnon JN, Brown CM, Backofen R (2012) Global or local? Predicting secondary structure and accessibility in mRNAs. *Nucleic Acids Res* 40(12):5215–5226
- Laurent GS, Wahlestedt C, Kapranov P (2015) The landscape of long noncoding RNA classification. *Trends Genet* 31(5):239–251
- Perdomo-Sabogal A, Kanton S, Walter MBC, Nowick K (2014) The role of gene regulatory factors in the evolutionary history of humans. *Curr Opin Genet Dev* 29:60–67
- Johnsson P, Lipovich L, Grandér D, Morris KV (2014) Evolutionary conservation of long non-coding RNAs; sequence, structure, function. *Biochim. Biophys. Acta – Gen. Subj* 1840(3):1063–1071
- Bernhart SH, Hofacker IL, Will S, Gruber AR, Stadler PF (2008) RNAalifold: improved consensus structure prediction for RNA alignments. *BMC Bioinformatics* 9(1):474
- Hoechsmann and Matthias (2005) The tree alignment model: algorithms. Bielefeld University, Implementations and Applications for the Analysis of RNA Secondary Structures
- Sabarinathan R, Tafer H, Seemann SE, Hofacker IL, Stadler PF, Gorodkin J (2013) RNAsnp: efficient detection of local RNA secondary structure changes induced by SNPs. *Hum Mutat* 34:546–556

21. Edgar RC (2004) MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics* 5(1):113
22. Gesell T, Washietl S (2008) Dinucleotide controlled null models for comparative RNA gene prediction. *BMC Bioinformatics* 9:248
23. Washietl S, Hofacker IL (2004) Consensus folding of aligned sequences as a new measure for the detection of functional RNAs by comparative genomics. *J Mol Biol* 342:19–30
24. Rivas E, Eddy SR (2001) Noncoding RNA gene detection using comparative sequence analysis. *BMC Bioinformatics* 2:8
25. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. *J Mol Biol* 215(3):403–410
26. Nawrocki EP, Eddy SR (2013) Infernal 1.1: 100-fold faster RNA homology searches. *Bioinformatics* 29(22):2933–2935
27. Chodroff RA, Goodstadt L, Sirey TM, Oliver PL, Davies KE, Green ED, Molnár Z, Ponting CP (2010) Long noncoding RNA genes: conservation of sequence and brain expression among diverse amniotes. *Genome Biol* 11(7):R72
28. Ponjavic J, Ponting CP, Lunter G (2007) Functionality or transcriptional noise? Evidence for selection within long noncoding RNAs. *Genome Res* 17(5):556–565
29. Nitsche A, Rose D, Fasold M, Reiche K, Stadler PF (2015) Comparison of splice sites reveals that long noncoding RNAs are evolutionarily well conserved. *RNA* 21(5):801–812
30. Hurst L (2002) The K_a/K_s ratio: diagnosing the form of sequence evolution. *Trends in Genetics* 18:486–489
31. Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J Roy Statist Soc Ser B* 57:289–300
32. Schirmer S, Ponty Y, Giegerich R (2014) Introduction to RNA secondary structure comparison. RNA sequence, structure, and function: computational and bioinformatic. *Methods*:247–273
33. Walter Costa MB, Höner zu Siederdisen C, Tulpan D, Stadler PF, Nowick K (2018) Temporal ordering of substitutions in RNA evolution: uncovering the structural evolution of the human accelerated region 1. *J Theor Biol* 438:143–150
34. Pollard KS, Salama SR, Lambert N, Lambot MA, Coppens S, Pedersen JS, Katzman S, King B, Onodera C, Siepel A et al (2006) An RNA gene expressed during cortical development evolved rapidly in humans. *Nature* 443(7108):167–172
35. Lares MR (2019) Synthesis, purification and crystallization of a putative critical bulge of HAR1 RNA. *PLoS One* 14(11):e0225029
36. Ziegeler M, Cevc M, Richter C, Schwalbe H (2012) NMR studies of HAR1 RNA secondary structures reveal conformational dynamics in the human RNA. *Chembiochem* 13(14):2100–2112
37. Beniaminov A, Westhof E, Krol A (2008). Distinctive structures between chimpanzee and human in a brain noncoding RNA. *RNA* 14 (7): 1270–1275
38. Léger S, Walter Costa MB, Tulpan D (2019) Pairwise visual comparison of small RNA secondary structures with base pair probabilities. *BMC Bioinformatics* 20(1):293
39. Aalberts DP, Jannen WK (2013) Visualizing RNA base-pairing probabilities with RNAbow diagrams. *RNA* 19:475–478



Network-Based Structural Alignment of RNA Sequences Using TOPAS

Chun-Chi Chen, Hyundoo Jeong, Xiaoning Qian, and Byung-Jun Yoon

Abstract

TOPAS (TOPological network-based Alignment of Structural RNAs) is a network-based alignment algorithm that predicts structurally sound pairwise alignment of RNAs. In order to take advantage of recent advances in comparative network analysis for efficient structurally sound RNA alignment, TOPAS constructs topological network representations for RNAs, which consist of sequential edges connecting nucleotide bases as well as structural edges reflecting the underlying folding structure. Structural edges are weighted by the estimated base-pairing probabilities. Next, the constructed networks are aligned using probabilistic network alignment techniques, which yield a structurally sound RNA alignment that considers both the sequence similarity and the structural similarity between the given RNAs. Compared to traditional Sankoff-style algorithms, this network-based alignment scheme leads to a significant reduction in the overall computational cost while yielding favorable alignment results. Another important benefit is its capability to handle arbitrary folding structures, which can potentially lead to more accurate alignment for RNAs with pseudoknots.

Key words RNA structural alignment, Network-based RNA alignment, Network alignment

1 Introduction

Sequence alignment has become a staple in modern biomedical research, as it provides effective computational means for comparative analysis of biological sequences. Nowadays, sequence alignment techniques lie at the core of various bioinformatics tools that are used for predicting novel genes and studying the function and structure of biomolecules. This is also the case for RNA research, where RNA alignment methods are widely used for comparative studies of RNA families and the prediction of novel non-coding RNAs (ncRNAs). For RNAs, their folding structure is known to play critical roles in carrying out their function, hence also well conserved among members in the same family. As a result, most RNA sequence alignment techniques consider the underlying

structure of the RNAs to be aligned to ensure that the obtained alignment results are biologically meaningful.

While such a structural alignment approach can enhance the overall accuracy of the RNA alignment, incorporating structural aspects of the RNAs into the alignment process tends to be computationally expensive. The first such algorithm was proposed by Sankoff based on a dynamic programming approach, aiming to simultaneously predict the best alignment and the consensus secondary structure for a given set of RNA sequences whose structure is unknown [1]. The computational cost for simultaneous alignment and folding of two RNA sequences of length n is $O(n^6)$ for the Sankoff algorithm, which makes it impractical for long sequences. To remedy this problem, various Sankoff-style alignment algorithms have been developed to date, which incorporate various speed improving heuristics while preserving the benefits of structural alignment [2–11].

Recently, with the availability of large-scale biological networks – especially, protein-protein interaction (PPI) networks – there have been significant research efforts to develop efficient tools for network alignment [12, 13]. While the general network alignment problem is computationally more complex compared to sequence alignment, this high complexity has inspired various computationally efficient network alignment techniques that can predict an accurate alignment of two or more networks with arbitrary topology. Notable examples are random walk-based Schemes [13], which will be discussed in further details in the next section. Building on recent advances in network alignment, TOPAS (**TOP**ological network-based **Al**ignment of **Str**uctural RNAs) takes a network-based approach for an efficient and accurate structural alignment of RNA sequences [14]. TOPAS first constructs a topological network for each RNA to be aligned, by integrating its sequence and structural properties. Subsequently, the resulting topological networks are efficiently aligned to each other using network alignment techniques. This network-based structural RNA alignment approach has several benefits, including enhanced accuracy, reduced computational cost, and higher flexibility. In fact, TOPAS can effectively handle RNAs with arbitrary folding structures, including RNA pseudoknots.

This chapter is organized as follows: In Subheading 2, we briefly discuss the network alignment problem and present existing network alignment approaches with a focus on random walk-based schemes. In Subheading 3, we provide an overview of TOPAS followed by a detailed description of the algorithm. Instructions for using the algorithm and examples will be provided in Subheading 4. We conclude the chapter in Subheading 5 with a brief discussion about TOPAS and some future perspectives regarding network-based RNA alignment.

2 Network Alignment

Network-based data representation provides an intuitive means to describe complex relationships between multiple variables. Due to the distinctive advantages of the network-based data representation, diverse network-based data interpretation algorithms have been developed and successfully applied to the analysis of social networks [15, 16], images [17–19], single-cell RNA sequencing data [20, 21], and biological networks [12, 13, 22–26].

Recently, thanks to various high-throughput measurement and large-scale data collection techniques, a growing number of biological networks with functional annotations have been archived in public databases. This has asked for the development of computational methods that can effectively analyze and utilize the collected biological networks. One possible solution is biological network alignment, which aims to identify similarities and differences between two (or more) biological networks by finding their best mapping [13]. As network alignment can identify conserved nodes or subnetworks across different biological networks, through the comparative analysis of the biological network of well-studied species and that of less-studied species, it enables knowledge transfer between species. For example, prior knowledge (e.g., functional annotation) regarding biomolecules or pathways may be transferred based on the predicted network mapping, which may in turn help the computational prediction of their functions (and other features of interest) without time-consuming and expensive biological experiments (Fig. 1).

To date, diverse network alignment algorithms have been proposed based on the different strategies and optimization objectives. For example, graphlet-based algorithms have been successfully applied to the network alignment and have been shown to yield reliable results [27–30]. The GRAAL (GRAph ALigner) family of

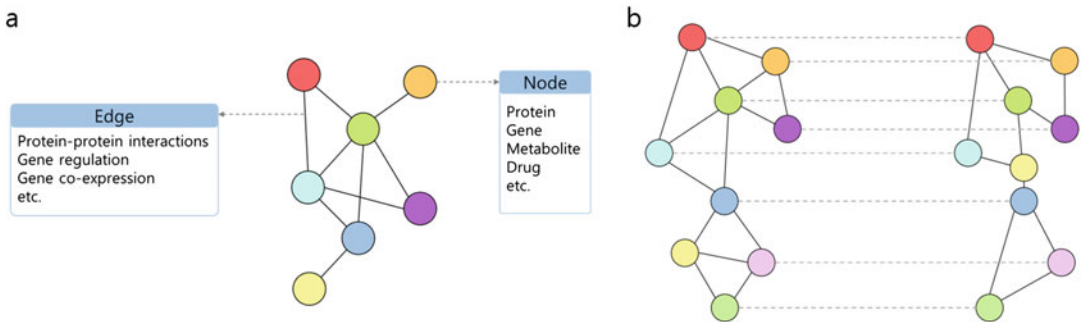


Fig. 1 Illustration of a biological network (left) and global network alignment (right). A network consists of nodes and edges, where the nodes represent biomolecules such as proteins or genes, and their relationships, interactions, or regulations can be represented by edges. Global network alignment aims to identify the best overall mapping between nodes across different networks

algorithms utilizes the graphlet signature for each node as the feature vector representing the topological structure of its surrounding local subnetwork. These algorithms yield the final network alignment based on the similarities of the given graphlet signature vectors. Another popular approach adopted by many network alignment algorithms is seed-and-extension, which has been shown to yield good results while being computationally efficient [31–33]. Network alignment algorithms in this category typically adopt a two-step approach, where they first compute a node alignment score that combines both topological and node-level similarities and then use a seed-and-extension scheme to find the network alignment that maximizes the overall alignment score. To estimate reliable alignment scores, various similarity measures can be exploited, such as GO similarity [31], topological significance [32], and so forth. Alternatively, various optimization techniques based on different objective functions have been employed to derive reliable network alignments [34–36]. These direct optimization-based algorithms first define an objective function that can quantify the quality of a network alignment based on the topological similarity and the node-level similarity between networks. Then, they optimize the objective function by using simulated annealing [34] or genetic algorithms [35, 36] to construct the optimal network alignment.

Recently, several network alignment algorithms have been developed based on a variety of random walk models, which have demonstrated accurate alignment performance at a relatively low computational cost [13, 22–26]. Random walk models are used to perform simultaneous random walk on a pair of networks, where the stationary probabilities resulting from the random walk can be used to quantify the similarity between nodes across networks in a way that integrates topological similarity as well as node-level similarity (e.g., sequence similarity). The resulting node correspondence scores (or node alignment probabilities) can be used for constructing the pairwise or multiple network alignment that maximizes the overall alignment score. IsoRank [22] is probably one of the first algorithms based on this approach, which adopted a random walk with restart model. SMETANA adopts a semi-Markov random walk (SMRW) model to derive multiple network alignments [23]. A context-sensitive random walk (CSRW) model that can switch its mode of random walk depending on the context (e.g., local network similarities) was used in [24], yielding accurate network alignment results. There are several distinctive advantages of random walk-based network alignment algorithms compared to other alignment methods. First of all, random walk models have been demonstrated to be very effective for integrating different types of similarities, such as topological similarity and node similarity, which is pivotal for obtaining accurate and biologically significant network alignment algorithms. Furthermore, random walk

models can provide a flexible framework for estimating the node correspondence for networks with arbitrary topological structures – i.e., without restricting them to simple structures such as linear paths or trees. Another important advantage of random walk models is that they can be efficiently implemented by taking advantage of the rich theoretical results in linear algebra and graph theory.

3 Overview of the TOPAS Algorithm

TOPAS is an efficient network-based RNA alignment algorithm that considers the structural similarity between RNAs. This algorithm takes a widely different approach from other classical Sankoff-style algorithms. By constructing *topological networks* that capture the potential folding structures underlying the RNA sequences to be aligned, TOPAS utilizes efficient network diffusion and alignment techniques to predict a structurally sound RNA alignment, where both sequence and structural similarities are considered simultaneously. The overall computational complexity is $O(n^2)$ for aligning two RNA sequences of length n , which is fairly efficient, and it practically outperforms many Sankoff-style alignment algorithms in terms of the actual CPU time needed for finding the structural alignment.

For each RNA, TOPAS constructs the topological network by first taking the RNA sequence as the backbone of the topological network. Each node in the topological network represents a nucleotide base in the RNA, following the original sequential order, and the edges in this backbone connect the neighboring nucleotides. To incorporate the structural information of the corresponding RNA into the topological network, weighted edges are added into the backbone network for node pairs that can form base-pairs. The edge weight is proportional to the base-pairing probability. Edges with low probability values (less than a threshold P_{Th}) can be removed from the network, which helps TOPAS to focus on the most likely folding structure and also improves its computational efficiency.

Figure 2 illustrates the topological network constructed from an RNA sequence, whereas Fig. 3 illustrates the alignment of two RNAs based on their corresponding topological networks. The network $G_n = (V_n, E_n)$ denotes the n th topological network constructed from the n th RNA sequence, where V_n represents the set of nodes (i.e., nucleotides) and E_n consists of the edges (i.e., both the backbone edges and the weighted edges based on the base-pairing probability).

To construct a structurally sound RNA alignment, TOPAS considers integrating three different types of similarities that capture different aspects of the RNAs to be aligned. The three similarities – the structure similarity R_S , the connected similarity

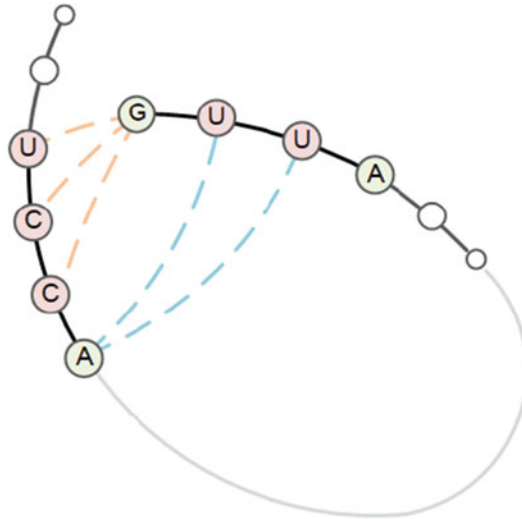


Fig. 2 Illustration of the topological network construction based on an RNA sequence. The backbone edges for neighboring nodes are shown in solid lines. The dashed lines show the weighted edges between nodes that can potentially form base-pairs

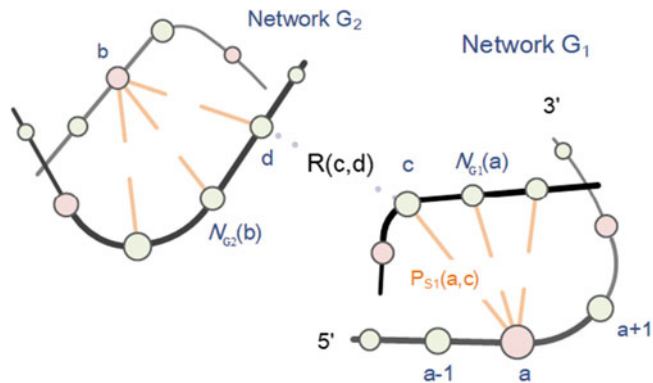


Fig. 3 Illustration of the RNA alignment based on topological networks. Network G_1 represents the first RNA and the network G_2 represents the second RNA. TOPAS utilizes network diffusion and alignment techniques to align the networks, thereby predicting the structural alignment of the original RNAs. $R(c, d)$ denotes the pairwise similarity that captures the sequence similarity between the nucleotide bases represented by the corresponding nodes at position c in network G_1 and position d in network G_2 . $P_{S_1}(a, c)$ is the base-pairing probability for the nodes at positions a and c in network G_1 . The set of the neighbors of the node at position a is denoted by $N_{G_1}(a)$ if there exist base-pairing interactions in network G_1 . (Reprinted with permission from Ref. [14])

R_C , and the sequence similarity R_E – are combined to measure the overall similarity R between nodes.

The **structure similarity** R_S is used to guarantee that the resulting RNA alignment incorporates the similarity between the

underlying structures of the two RNAs and the presence of a common consensus structure. This similarity is defined as:

$$R_S(a, b) = \sum_{\substack{c \in N_{G_1}(a) \\ d \in N_{G_2}(b)}} \frac{P_{S_1}(a, c)P_{S_2}(b, d)}{D(c)D(d)} R(c, d), \quad (1)$$

where $N_{G_n}(v)$ is the set of connected neighbors of the node v in the network G_n . $P_{S_1}(a, c)$ is the base-pairing probability between nodes a and c in network G_1 . Similarly, $P_{S_2}(b, d)$ denotes the base-pairing probability between nodes b and d in network G_2 . The weighted degree of nodes c in network G_1 is given by $D(c) = \sum_{u \in N_{G_1}(c)} P_{S_1}(u, c)$, and similarly, the weighted degree of nodes d in the network G_2 is given by $D(d) = \sum_{v \in N_{G_2}(d)} P_{S_2}(v, d)$.

The **connected similarity** R_C is used to measure the similarity between neighborhoods. The connected similarity is defined as:

$$R_C(a, b) = \frac{1}{2}(R(a-1, b-1) + R(a+1, b+1)). \quad (2)$$

As shown here, the overall similarity R is used to capture the sequence and structural similarities for the nodes adjacent to (a, b) in the respective networks. It is critical to note that R_S and R_C are defined in a recursive fashion to enable effective network-based similarity scoring in TOPAS that captures the global sequence and structure conservation across the given RNAs.

In addition to the similarities R_S and R_C , the **sequence similarity** R_E aims to capture the nucleotide-base similarity between the RNAs in terms of their sequence composition. In TOPAS, the sequence similarity R_E between two nodes in different topological networks is estimated by computing the sequence-level alignment probability P_A by using a sequence alignment algorithm.

The overall similarity R between any node pair across the topological networks is computed by integrating the structure similarity R_S , the connected similarity R_C , and the sequence similarity R_E as follows:

$$R(a, b) = \alpha R_S(a, b) + \beta R_C(a, b) + (1 - \alpha - \beta) R_E(a, b), \quad (3)$$

where α and β are topological weighting parameters for the structure similarity R_S and connected similarity R_C respectively. The topological weighting parameters are nonnegative and $\alpha + \beta \leq 1$. These parameters can be used to control and balance the trade-off among the three similarity measures. Note that Eq. 3 is also recursive if we plug in Eqs. 1 and 2.

Equation 3 can be written in a matrix form as $R = AR$. The matrix A is the overall weighting coefficient matrix for the linear combination of the three similarities (R_S, R_C, R_E) , and it can be derived from Eq. 3. Based on this recursive matrix equation, we can efficiently estimate the overall network-based similarity R by using the power method through the following iterative updates:

$$R^{(k+1)} \leftarrow \frac{AR^{(k)}}{|AR^{(k)}|_1}, \quad (4)$$

where $R^{(k)}$ is the estimation of the overall similarity score R in the k -th iteration. The initial similarity $R^{(0)}$ can be set to a nonnegative random matrix with the constraint $|R^{(0)}|_1 = 1$. The convergence rate of the power method is dominated by the second-largest eigenvalue of the matrix A . This iterative procedure based on the power method resembles the random walk, belief propagation, or network diffusion procedure on the “alignment network” in the IsoRank network alignment algorithm [22]. In practice, we can limit the number of iterations to a fixed number N_{It} if needed.

Based on the estimated network-based similarity score R , we can find the optimal pairwise alignment of the topological networks by maximizing the sum of the overall similarity scores of the aligned nodes. While finding the exact solution for general networks is computationally costly, the inherent ordering of the nodes in each topological network (arising from the original base positions in the corresponding RNA) allows us to find the optimal solution using dynamic programming, namely, the well-known Needleman-Wunsch algorithm.

The **pseudo-code** of TOPAS is shown below.

TOPAS: network-based RNA structural alignment algorithm

Output: structural alignment $(\widehat{S}_1, \widehat{S}_2)$

Input: RNA sequences (S_1, S_2) , probabilistic model (P_A, P_{S_1}, P_{S_2})

Parameters: $(\alpha, \beta, N_{It}, P_{Th})$

```

1. Construct topological networks
  for n=1 to 2
    Construct  $G_n = (V_n, E_n)$  from the sequences data  $(S_n, P_{S_n}, P_{Th})$ 
2. Run power method to estimate similarity  $R$ 
  Initialize the similarity  $R^{(0)}$  with a nonnegative random unit vector
  for k=1 to  $N_{It}$ 
    Initialize  $R_S, R_C$  to 0
    for a= 1 to length( $V_1$ )
      for b= 1 to length( $V_2$ )
        Update structure similarity
        for each  $(c, d) \in (N_{G_1}(a), N_{G_2}(b))$ 
           $R_S(a, b) + = R^{(k-1)}(c, d)[P_{S_1}(a, c)P_{S_2}(b, d)/D(c)D(d)]$ 
        Update connected similarity
        if Exist  $R(a-1, b-1)$ 
           $R_C(a, b) + = \frac{1}{2}R^{(k-1)}(a-1, b-1)$ 
        if Exist  $R(a+1, b+1)$ 
           $R_C(a, b) + = \frac{1}{2}R^{(k-1)}(a+1, b+1)$ 
        Update overall similarity
         $R_A^{(k)}(a, b) = \alpha R_S(a, b) + \beta R_C(a, b) + (1 - \alpha - \beta)R_E(a, b)$ 
      end
    end
  end
  Normalize and update the overall similarity
   $R^{(k)} = AR_A^{(k)} / |AR_A^{(k)}|_1$ 

```

(continued)

```

Stop criterion
if  $|R^{(k)} - R^{(k-1)}| < Tolerance$ 
    break
end

```

3. Run dynamic programming (Needleman-Wunsch algorithm) to find the alignment that maximizes the overall similarity score $R(\widehat{S}_1, \widehat{S}_2)$
4. Output the corresponding RNA structural alignment $(\widehat{S}_1, \widehat{S}_2)$

The computational complexity of the TOPAS algorithm is dominated by the step that estimates the overall similarity R . The memory complexity of TOPAS is $O(N^2)$ while its time complexity is $O(kd_1d_2N^2)$, where k is the number of iterations in the power method and d_n is the number of effective base-pairing edges in the network G_n . In general, for topological networks constructed from real RNAs, the effective base-pairing interaction edges tend to be sparse in the constructed networks; hence, we have $kd_1d_2 \gg N^2$.

4 Running the TOPAS Algorithm

TOPAS is implemented in Matlab and it can be downloaded at <https://github.com/bjyoontamu/TOPAS>.

One can use TOPAS to predict the structural alignment of two RNAs by invoking the following Matlab function:

```
[output_sequence1, output_sequence2] = TOPAS(input_sequence,
file_basepair1, file_basepair2, file_alignment, Alpha, Beta)
```

The first argument `input_sequence` specifies the file that contains the RNA sequences to be aligned. The file should contain two RNA sequences in FASTA format. The second and third arguments `file_basepair1` and `file_basepair2` specify the files containing the base-pairing probabilities. The fourth argument `file_alignment` refers to the file that contains the (sequence-based) alignment probabilities. Finally, `Alpha` and `Beta` are, respectively, used to specify the weighting parameter α for the topological similarity and the parameter β for the connected similarity in Eq. 3, which are used to compute the overall network-based similarity score R . The input file formats are summarized in the following Table 1.

We now showcase the usage of TOPAS using the RNA sequences taken from the Rfam database [37] as examples. In the examples that will follow, the base-pairing probabilities and base alignment probabilities are estimated by the RNAstructure package [38]. RNAstructure is a software package for RNA and DNA secondary structure analysis that provides RNA secondary structure prediction and the sequence alignment based on a pair-HMM model.

Table 1
Summary of input file formats

Input sequences	FASTA format
Base-pairing probabilities	Position1 Position2 probability
Alignment probabilities	Position1_of_sequence1 Position2_of_sequence2 probability

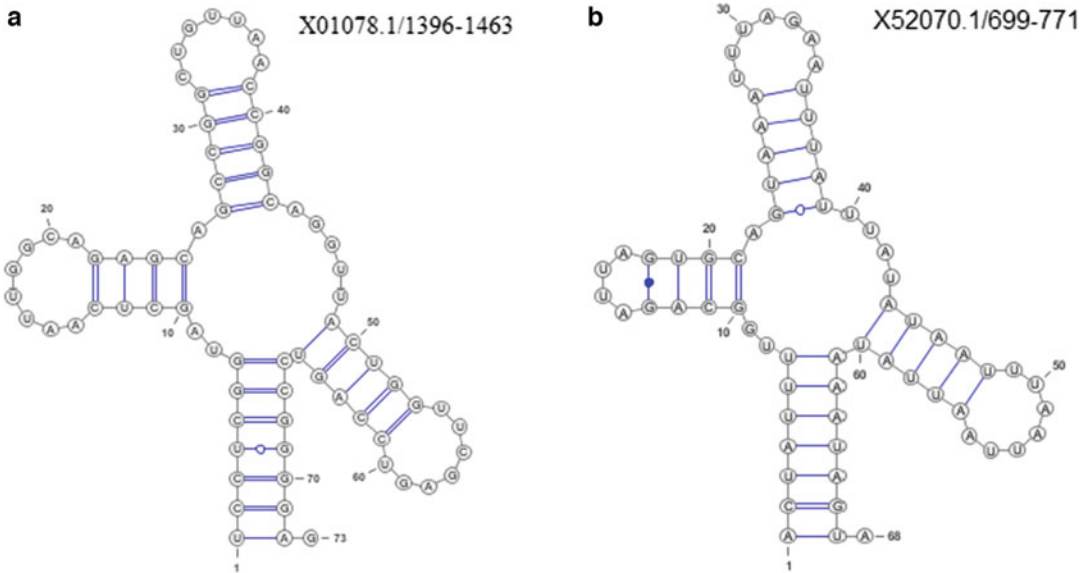


Fig. 4 Illustration of the secondary structures of the tRNAs X52070.1/699-771 and X01078.1/1396-1463. The secondary structures are drawn using VARNa [39]. (a) The secondary structure of tRNA X52070.1/699-771. (b) The secondary structure of tRNA X52070.1/699-771

Example 1: Alignment of tRNAs

In this example, the tRNA pair X52070.1/699-771 and X01078.1/1396-1463 are aligned using TOPAS. The secondary structures of these tRNAs are shown in Fig. 4. Furthermore, the tRNA sequences are shown in Table 2, where the bases forming the clover structure are colored. Each color represents one of the leaves in the clover structure for readability. The goal of structural alignment is to align the two RNAs such that the alignment result faithfully reflects the matching sequence and structure. As a comparison, we also show the sequence-based alignment result using a pair-HMM.

By comparing the predicted RNA alignment with the ground truth (obtained from Rfam), one can evaluate the performance of the alignment by estimating sensitivity (SEN), positive predictive value (PPV), and accuracy (ACC), which are defined as follows:

Table 2
The tRNA sequences and the comparison of the alignment

X52070.1/699-771

UCCUCGGUAGCUCAAUUGGCAGAGCAGCCGGCUGUUAACCGGCAGGUUACUGGUUCGA
 GUCCAGUCCGGGGAG

X01078.1/1396-1463

ACUAAUUUGGCAGAUUAGUGCAGUAAAUUUAGAAUUUAUUUAUAUAAUUUAAUUAAUU
 AUAAAUAGUA

Ground truth alignment in Rfam:

UCCUCGGUAGCUCAAUUGGCAGAGCAGCCGGCUGUUAACCGGCAGGUUACUGGUUCGA
 GUCCAGUCCGGGGAG

ACUAAUUUGGCAGAUU----AGUGCAGUAAAUUUAGAAUUUAUUUA-
 UAUAUUUAAUUAAUUUAUAAAUAGUA

Sequence-based alignment using pair-HMM:

UCCUCGGUAGCUCAAUUGGCAGAGCAGCCGGCUGUUAACCGGCAGGUUACUGGUUCGA
 GUCCA-----GUCCGGGGAG

-----ACUAAUUUGGCAGAUUAGU--

GCAGUAAAUUUAGAAUUUAUUUAUAUAAUUUAAUUAAUUUAUAAAUAGUA

Structural alignment based on TOPAS ($\alpha=0.50$, $\beta=0.48$):

UCCUCGGUAGCUCAAUUGGCAGAGCAGCCGGCUGUUAACCGGCAGGUUACUGGUUCGA
 GUCCAGUCCGGGGAG

ACUAAUUUGGCAGAUU----AGUGCAGUAAAUUUAGAAUUUAUUU-
 AUUAUUUAAUUAAUUUAUAAAUAGUA

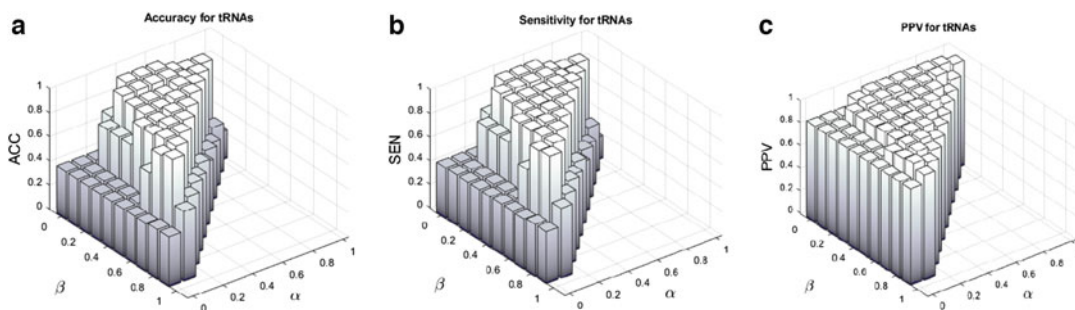


Fig. 5 Illustration of the structural alignment performance for the tRNA pair X52070.1/699-771 and X01078.1/1396-1463. **(a)** Accuracy (ACC) for the tRNA pair. **(b)** Sensitivity (SEN) for the tRNA pair. **(c)** Positive predictive value (PPV) for the tRNA pair

$$\text{SEN} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

The TP, TN, FP, and FN are the number of true positives, true negatives, false positives, and false negatives, respectively. They can be calculated by comparing the predicted alignment with the target alignment (i.e., the ground truth). The accuracy of the tRNA alignment based on the pair-HMM is only 13.7%, which clearly shows that sequence-based RNA alignment may not yield structurally sound alignment results. The accuracy of the tRNA pairwise alignment reaches 97.3% when one uses TOPAS ($\alpha=0.50$, $\beta=0.48$, $N_{\text{It}} = 30$, $P_{\text{Th}} = 0.01$). The structural alignment obtained from TOPAS is shown in Table 2. Figure 5 shows SEN, PPV, and ACC of TOPAS for the given tRNA pair for various combinations of weight parameters (α , β). The plots in Fig. 5 show that one needs to utilize all three different types of similarities in TOPAS to attain the best results possible. If one has confidence in the RNA structure predictions, a larger weight may be assigned to the topological similarity score. In general, by incorporating both topological similarity and connected similarity and assigning a relatively small (but nonzero) weight to sequence similarity, one can obtain accurate alignment results using TOPAS.

Example 2: Alignment of Downstream-Peptide RNAs

TOPAS can also effectively handle the alignment of RNAs whose structures contain pseudoknots, which is illustrated in this example. The downstream-peptide RNAs AACY021152154.1/3-79 and AACY023856379.1/52-113 both have non-nested secondary structures that contain crossing base-pairs. This is shown in Fig. 6, where base-pairs in different stems are shown in different colors so that it is easy to see that these base-pairing interactions cross each other. The alignment results for the give RNA pair are shown in Table 3. In this example, the accuracy of the downstream-peptide RNA alignment obtained using a pair-HMM was only 54.55%, while the accuracy reached 87.01% when TOPAS was used ($\alpha=0.50$, $\beta=0.48$, $N_{\text{It}} = 30$, $P_{\text{Th}} = 0.01$). Figure 7 shows

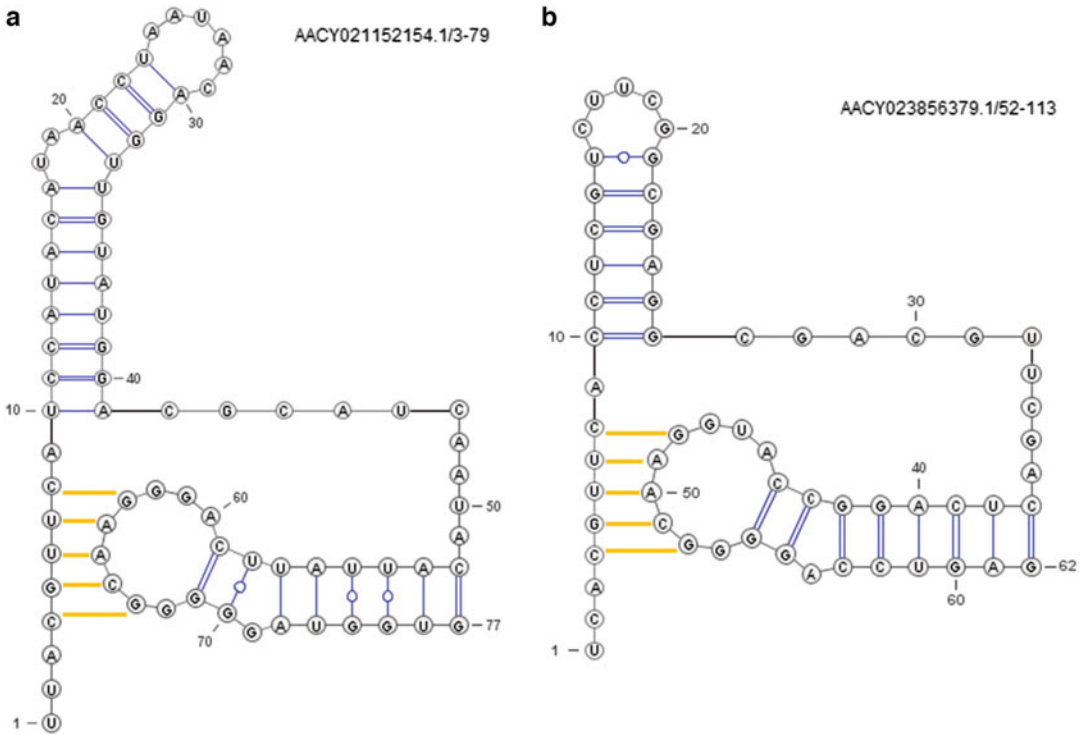


Fig. 6 Illustration of the secondary structure for the downstream-peptide RNA AACY021152154.1/3-79 and AACY023856379.1/52–113. The secondary structures are modified from the drawing by using VARNA [4]. **(a)** The secondary structure of downstream-peptide RNA AACY021152154.1/3-79. **(b)** The secondary structure of downstream-peptide RNA AACY023856379.1/52-113

the SEN, PPV, and ACC of TOPAS for using different weight parameters (α , β) for finding the structural alignment. The overall trends are similar as in the previous example. As illustrated in this example, TOPAS is capable of predicting an accurate RNA structural alignment even when the RNAs have pseudoknot structures.

5 Notes

While Sankoff-style algorithms dominate the field, a network-based RNA structural alignment, such as TOPAS, may potentially provide several benefits over those methods. In fact, comprehensive performance assessment based on several RNA families and the BRAlIbase 2.1 K2 dataset has shown that TOPAS outperforms several popular Sankoff-style RNA alignment algorithms in many cases, in terms of both accuracy and computational efficiency [14]. Furthermore, TOPAS represents RNA sequences as topological networks, which enables handling arbitrary folding structures (such as

Table 3
The downstream-peptide RNA sequences and the comparison of the alignment

<p>AACY021152154.1/3-79</p> <p>UUACGUUCAUCCAUAACAUAACCUAUAACAGGUUGUAUGGACGCAUCAUAUUAUUCAGGGAACGGGGAUGGUG</p> <p>CAGGGAACGGGGAUGGUG</p>
<p>AACY023856379.1/52-113</p> <p>UCACGUUCACCUCGUCUUCGGCGAGGCGCAGUUCGACUCAGGCCAUGGAACGGGGACC</p> <p>UGAG</p>
<p>Ground truth alignment in Rfam:</p> <p>UUACGUUCAUCCAUAACAUAACCUAUAACAGGUUGUAUGGACGCAUCAUAUUAUUCAGGGAACGGGGAUGGUG</p> <p>UCACGUUCACCUCGU-----CUUCG-----GCGAGGCGCAGUUCGACUCAGGCCAUGGAACGGGGACCUGAG</p>
<p>Sequence-based alignment using pair-HMM:</p> <p>UUACGUUCAUCCAUAACAUAACCUAUAACAGGUUGUAUGGACGCAUCAUAUUAUUCAGGGAACGGGGAUGGUG</p> <p>UCACGUUCACCUCGUCUUCGGCGAGGCGCAGUUCG-----ACUCAGGCCAUGGAACGGGGACCUGAG</p>
<p>Structural alignment based on TOPAS ($\alpha=0.50, \beta=0.48$):</p> <p>UUACGUUCAUCCAUAACAUAACCUAUAACAGGUUGUAUGGACGCAUCAUAUUAUUCAGGGAACGGGGAUGGUG</p> <p>UCACGUUCACCUCGUCUUC-----GGCGAGGCGCAGUUCGACUCAGGCCAUGGAACGGGGACCUGAG</p>

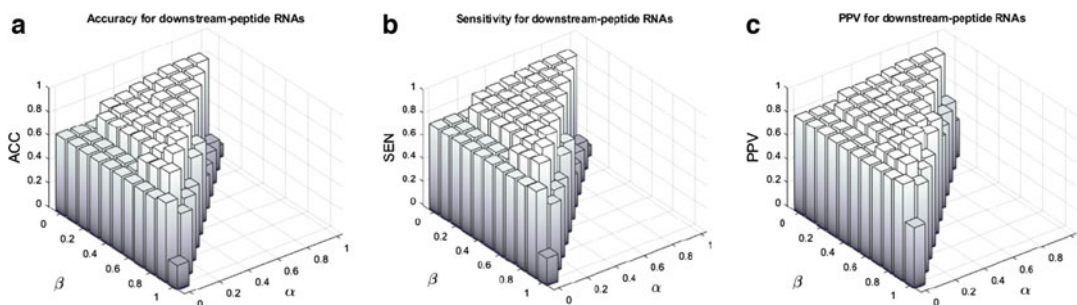


Fig. 7 Illustration of the structural alignment performance for downstream-peptide RNA pair AACY021152154.1/3-79 and AACY023856379.1/52-113. **(a)** ACC for the downstream-peptide RNA pair. **(b)** SEN for the downstream-peptide RNA pair. **(c)** PPV for the downstream-peptide RNA pair

pseudoknots) without any restriction. To the best of our knowledge, TOPAS is the first RNA structural alignment algorithm that represents RNA sequences as topological networks and directly takes advantage of network alignment techniques for constructing

RNA sequence alignments that are structurally sound. We expect that the performance of TOPAS may be further improved in the future by enhancing the network construction scheme – i.e., how the topological network is constructed from a given RNA sequence – and by exploring other potential network alignment strategies. The original TOPAS adopted a network alignment scheme similar to the one proposed in IsoRank [22], which is essentially based on a random walk with restart. TOPAS may potentially benefit from more recent random walk models [23–25], which have been shown to enhance network alignment performance. Moreover, customized random walk schemes that are specifically designed and optimized for the comparative analysis of topological networks representing RNA sequences and their underlying structures could improve the accuracy and efficiency of network-based RNA structural alignment algorithms even further.

References

1. Sankoff D (1985) Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J Appl Math* 45:810–825
2. Mathews DH, Turner DH (2002) Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. *J Mol Biol* 317:191–203
3. Hofacker IL et al (2004) Alignment of RNA base pairing probability matrices. *Bioinformatics* 20:2222–2227
4. Havgaard JH et al (2005) Pairwise local structural alignment of RNA sequences with sequence similarity less than 40%. *Bioinformatics* 21:1815–1824
5. Gardner PP et al (2005) A benchmark of multiple sequence alignment programs upon structural RNAs. *Nucleic Acids Res* 33:2433–2439
6. Will S et al (2007) Inferring noncoding RNA families and classes by means of genome-scale structure-based clustering. *PLoS Comput Biol* 3:e65
7. Chuong BD et al (2008) A max-margin model for efficient simultaneous alignment and folding of RNA sequences. *Bioinformatics* 24:i68–i76
8. Harmanci AO et al (2008) PARTS: probabilistic alignment for RNA joint secondary structure prediction. *Nucleic Acids Res* 36:2406–2417
9. Fu Y et al (2014) Dynalign II: common secondary structure prediction for RNA homologs with domain insertions. *Nucleic Acids Res* 42:13939–13948
10. Will S et al (2015) SPARSE: quadratic time simultaneous alignment and folding of RNAs without sequence-based heuristics. *Bioinformatics* 31:2489–2496
11. Sundfeld D et al (2016) Foldalign 2.5: multi-threaded implementation for pairwise structural RNA alignment. *Bioinformatics* 32:1238–1240
12. Sharan R, Ideker T (2006) Modeling cellular machinery through biological network comparison. *Nat Biotechnol* 24(4):427–433
13. Yoon BJ, Qian X, Sahraeian SME (2011) Comparative analysis of biological networks: hidden markov model and markov chain-based approach. *IEEE Signal Process Mag* 29(1):22–34
14. Chen CC, Jeong H, Qian X, Yoon BJ (2019) TOPAS: network-based structural alignment of RNA sequences. *Bioinformatics* 35(17):2941–2948
15. Palla G et al (2005) Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435(7043):814–818
16. Backstrom L, Leskovec J (2011) Supervised random walks: predicting and recommending links in social networks. In: *Proceedings of the fourth ACM international conference on web search and data mining*
17. Duchenne O et al (2011) A tensor-based algorithm for high-order graph matching. *IEEE Trans Pattern Anal Mach Intell* 33(12):2383–2395
18. Shi J, Malik J (2000) Normalized cuts and image segmentation. *IEEE Trans Pattern Anal Mach Intell* 22(8):888–905

19. Qian X, Yoon BJ (2010) Shape matching based on graph alignment using hidden Markov models. In: IEEE international conference on acoustics, speech and signal processing. IEEE
20. Jeong H, Liu Z (2020) PRIME: a probabilistic imputation method to reduce dropout effects in single cell RNA sequencing. *Bioinformatics* 36(13):4021–4029
21. Jeong H, Khunlertgit N (2020) Effective single-cell clustering through ensemble feature selection and similarity measurements. *Comput Biol Chem* 87:107283
22. Singh R, Jinbo X, Berger B (2008) Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proc Natl Acad Sci* 105(35):12763–12768
23. Sahraeian SME, Yoon BJ (2013) SMETANA: accurate and scalable algorithm for probabilistic alignment of large-scale biological networks. *PLoS One* 8(7):e67995
24. Jeong H, Yoon BJ (2015) Accurate multiple network alignment through context-sensitive random walk. *BMC Syst Biol* 9(1):1–12
25. Jeong H, Qian X, Yoon BJ (2015) Effective comparative analysis of protein-protein interaction networks by measuring the steady-state network flow using a Markov model. *BMC Bioinformatics* 17(13):15–27
26. Huang Q, Wu LY, Zhang XS (2011) An efficient network querying method based on conditional random fields. *Bioinformatics* 27(22):3173–3178
27. Kuchaiev O et al (2010) Topological network alignment uncovers biological function and phylogeny. *J R Soc Interface* 7(50):1341–1354
28. Malod-Dognin N, Pržulj N (2015) L-GRAAL: Lagrangian graphlet-based network aligner. *Bioinformatics* 31(13):2182–2189
29. Memišević V, Pržulj N (2012) C-GRAAL: common-neighbors-based global GRAPh ALignment of biological networks. *Integr Biol* 4(7):734–743
30. Kuchaiev O, Pržulj N (2011) Integrative network alignment reveals large regions of global network similarity in yeast and human. *Bioinformatics* 27(10):1390–1396
31. Patro R, Kingsford C (2012) Global network alignment using multiscale spectral signatures. *Bioinformatics* 28(23):3105–3114
32. Hashemifar S, Xu J (2014) Hubalign: an accurate and efficient method for global alignment of protein–protein interaction networks. *Bioinformatics* 30(17):i438–i444
33. Alkan F, Erten C (2014) BEAMS: backbone extraction and merge strategy for the global many-to-many alignment of multiple PPI networks. *Bioinformatics* 30(4):531–539
34. Mamano N, Hayes WB (2017) SANA: simulated annealing far outperforms many other search algorithms for biological network alignment. *Bioinformatics* 33(14):2156–2164
35. Saraph V, Milenković T (2014) MAGNA: maximizing accuracy in global network alignment. *Bioinformatics* 30(20):2931–2940
36. Vijayan V, Saraph V, Milenković T (2015) MAGNA++: maximizing accuracy in global network alignment via both node and edge conservation. *Bioinformatics* 31(14):2409–2411
37. Griffiths-Jones S et al (2003) Rfam: an RNA family database. *Nucleic Acids Res* 31:439–441
38. Reuter JS, Mathews DH (2010) RNAstructure: software for RNA secondary structure prediction and analysis. *BMC Bioinformatics* 11(1):1–9
39. Darty K et al (2009) VARNA: interactive drawing and editing of the RNA secondary structure. *Bioinformatics* 25:1974–1975



Fast RNA-RNA Interaction Prediction Methods for Interaction Analysis of Transcriptome-Scale Large Datasets

Tsukasa Fukunaga and Michiaki Hamada

Abstract

The computational prediction of RNA-RNA interactions has long been studied in RNA informatics. Most of the existing approaches focused on the interaction prediction of short RNAs in small datasets. However, in recent years, two fast prediction methods, RIsearch2 and RIBlast, have been developed to predict transcriptome-scale interactions or long RNA interactions. The key idea of the software acceleration of these tools was the integration of a seed-and-extend method, which is used in fast sequence alignment tools, into RNA-RNA interaction prediction. As a result, the two software programs were ten to a thousand times faster than the existing tools; because of this acceleration, detection of genome-wide microRNA target sites or interaction partners of function-unknown long noncoding RNAs has become possible. In this review, we describe the basic concept of the algorithm, its applications, and the future perspectives of the fast RNA-RNA interaction prediction tools.

Key words RNA-RNA interaction, lncRNA, miRNA target prediction, Seed-and-extension method, RNA accessibility

1 Introduction

Many functional noncoding RNAs (ncRNAs) exert their functions through complementary base-pairing interaction with other RNA molecules. For example, microRNAs (miRNAs) regulate gene expression by binding their seed regions to 3'-untranslated regions (3'-UTRs) of the target mRNAs [1]. As another example, 7SL, which is a human long noncoding RNA (lncRNA), suppresses the translation of TP53 by interacting with the 3'-UTR of the mRNA [2]. These examples demonstrate that the identification of RNA-RNA interactions is an effective approach for estimating the function of ncRNAs.

Recent advances in high-throughput sequencing technologies have enabled the comprehensive experimental detection of in vivo RNA-RNA interactions [3–7]. However, because many transcripts,

especially lncRNAs, show tissue-specific or developmental stage-specific expression patterns [8], performing these experiments on various tissues or developmental stages is necessary to understand the overall picture of RNA-RNA interactions. Because these experiments are labor-intensive, the computational prediction of comprehensive RNA-RNA interaction is an essential method.

To date, various software programs have been developed for predicting RNA-RNA interactions with high accuracy [9–12]. However, most of these programs have focused on predicting the interactions of short RNAs in small datasets, and they cannot be used for comprehensive prediction in transcriptome-scale large datasets because of the prohibitive computation time. Therefore, in recent years, two fast RNA-RNA interaction prediction methods have been developed, RIssearch2 [13] and RIBlast [14]. These tools enable us to conduct biological studies based on large-scale prediction results.

In this chapter, we review the basic concept of the algorithms and the applications of the fast RNA-RNA interaction prediction tools. This chapter is organized as follows: In Subheading 2, we briefly explain the basic algorithms of RIssearch2 and RIBlast. In Subheading 3, we introduce the recent updates of RIBlast. In Subheading 4, we present some examples of biological studies that have used these software programs. Finally, we give a conclusion and discuss future perspectives in Subheading 5.

2 Basic Algorithms for Fast RNA-RNA Interaction Prediction

We first briefly describe the concept of the RNA-RNA interaction prediction algorithms for small datasets. Although a variety of approaches have been developed for this purpose [15], several benchmarking studies have demonstrated that the methods based on the following two energies can achieve high accuracy: the hybridization energy and the accessibility energy [16–17]. Hybridization energy is the stabilizing energy of the RNA-RNA interaction, which is derived from hybridization between the complementary regions of two RNAs (*see* Fig. 1a). Programs such as IntaRNA [9] accurately calculate the energy using nearest-neighbor energy models [18]. However, this method requires relatively large computation times. Therefore, some methods, such as RIssearch [11], use the approximate scores for a less accurate but faster calculation of the hybridization energy. Accessibility energy is the energy that represents the ease of formation of RNA-RNA interactions in the region, and it is calculated based on the internal secondary structure of each of the RNAs (*see* Fig. 1b). Briefly, a region that tends to form intramolecular base pairs has high accessibility energy, and such regions are unlikely to form interactions with other RNAs. Accessibility energy can be calculated based on partition function

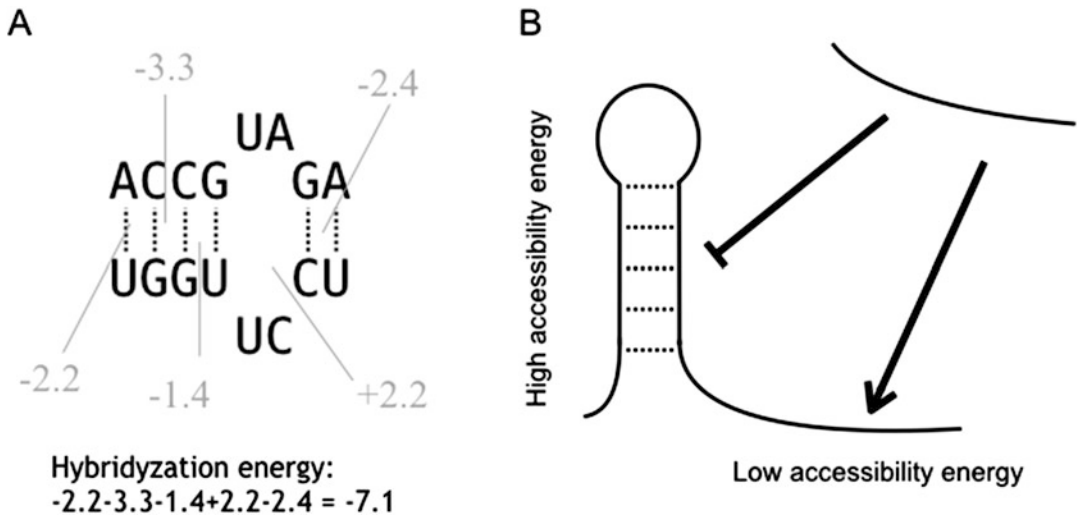


Fig. 1 (a) An example of hybridization energy calculation. Hybridization energy can be calculated as the summation of loop energies and stacking energies in the interaction region using the nearest-neighbor energy model. In this example, we used Turner 2004 parameters for the energy model [18]. (b) A schematic illustration of the influence of the accessibility energy on the RNA-RNA interactions. Regions with low accessibility energy are likely to interact with other RNAs

algorithms [19–20]. Highly accurate RNA-RNA prediction methods, such as IntaRNA [9], calculate the sum of these two energies and envisage regions with the lower values as the interaction regions.

However, these methods are too computationally time-consuming to be applied to large datasets. In particular, the calculation of hybridization energy needs extensive computational costs. As a result, faster RNA-RNA interaction prediction tools are required for the interaction analysis of the transcriptome-scale datasets. In recent years, two fast prediction tools, RIsearch2 [13] and RIblast [14], have been developed for this purpose. RIsearch2 and RIblast are freely available at <https://rth.dk/resources/risearch/> and <https://github.com/fukunagatsu/RIblast>, respectively. These methods focus on the similarity between the calculation of hybridization energy and that of the sequence alignment score. They achieve significant acceleration by utilizing the seed-and-extend method, which is an acceleration technique widely used in sequence alignment tools, such as BLAST [21] and LAST [22]. In these methods, the programs first rapidly search perfectly complementary short regions using a suffix array, which is an efficient text indexing data structure. The regions detected in this step are called the “seed” regions. Next, the programs “extend” the interaction regions from both ends of the seed regions based on the calculation method of the hybridization energy. Namely, the seed-and-extend method reduces the computation time by calculating the hybridization energy of only the vicinity of the seed region and

not that of the whole sequence region. Note that the seed-and-extend method is an approximate approach and thus may fail to detect the interactions that could be detected by the existing software.

Both RIssearch2 and RIblast achieved faster computations than their predecessor tools. However, there are three main differences between RIssearch2 and RIblast. The first difference is in the design of the seed regions. RIssearch2 uses fixed-length seeds, which are consecutive complementary regions with a user-defined parameter length k . This seed design is also used in BLAST for sequence alignment. On the other hand, RIblast adopts score-based seeds, which were proposed by Suzuki et al. in their work on the GHOSTX sequence alignment tool [23]. In this design, the seed regions were defined as the complementary regions whose hybridization energy is less than a user-defined parameter energy e . The second difference is in the calculation of the hybridization energy in the extension step. RIssearch2 rapidly calculates the energy based on the approximate score like RIssearch, whereas RIblast uses a nearest-neighbor energy model like IntaRNA to achieve accurate calculation. In this step, RIblast uses Andronescu's BL* energy model, which is estimated from thermodynamic melting data and experimentally determined RNA structure data using the Boltzmann likelihood method [24]. The third difference is in the usage of the accessibility energy. RIssearch2 does not calculate the accessibility energy and predicts the interactions based on only the hybridization energy, whereas RIblast uses the sum of the hybridization energy and the accessibility energy. As a result of these differences, although RIssearch2 can detect RNA-RNA interactions much faster than RIblast, RIblast can detect RNA-RNA interactions with higher accuracy than RIssearch2 [25].

3 Recent Updates of RIblast

In this section, we present the improvements of the RIblast software after the original RIblast study was published.

The first improvement is the development of a method for calculating the p -value in the RIblast evaluation results [26]. Although the interaction score calculated based on the hybridization and accessibility energies is a good indicator of the RNA-RNA interaction strength, the direct usage of the raw scores has no statistical guarantee and, thus, may result in the detection of false-positive interactions. Therefore, several previous studies calculated the p -values of the detected RNA-RNA interactions for the statistical assessment [27–28]. However, the p -value calculations focused only on small RNAs and could not be applied to long RNAs, which are the targets of RIblast. Accordingly, we developed a method for assessing the statistical significance of the RNA-RNA interactions between two long RNAs. We applied RIblast to

randomized pairs of human lncRNA sequences cut to specific long sequence lengths and validated that the minimum interaction scores between two RNAs follow a Gumbel distribution under the condition that repetitive sequences are masked. Next, we calculated the p -values by assuming that the null distribution of the interaction scores follows the Gumbel distribution.

The second improvement is the further acceleration from the original version of RIblast while retaining the prediction accuracy. For the acceleration, we introduced two techniques into RIblast. The first technique is the usage of the approximate calculation of the logarithmic and exponential functions. In calculating the accessibility energy in RIblast, we used the log-sum-exp method to avoid underflows in the numerical computation. However, because the method requires the execution of the logarithmic and the exponential functions with substantial calculation costs many times, the calculation of the accessibility energy needs a lot of computation time. To reduce the computation time, we used approximate but fast logarithmic and exponential functions provided in the “fast math” library instead of the default C++ library; we achieved the acceleration of the accessibility energy calculation [29]. The second technique is the usage of a stacked pairing constraint in the extension step of RIblast [12]. The idea is based on the following properties of the nearest-neighbor energy model: The regions that contribute to stabilizing the RNA-RNA interactions are stacked base pair regions, and thus, base pairs that do not form the stack or those that are too short-stacked are less likely to be included in the actual RNA-RNA interactions. Therefore, even if we restrict the candidate base pairs in the extension step to only the consecutive stacked base pairs with length l or more, we can still detect the actual RNA-RNA interactions. The speedup would be achieved by reducing the number of candidate base pairs when l is an appropriate value. Here, l is a user-defined parameter. Note that the RIblast extension step consists of the gapless extension step and the gapped extension step, and we applied this constraint to only the gapped extension step. Refer to the original RIblast study for details [14].

We compared the prediction accuracy and the computation times of the original version 1.0 with the newly developed version 1.2 of RIblast. In this comparison, we set the parameter l to 3. We conducted the same experiments as in the original RIblast study to evaluate their performances. Refer to the original study for more details on the experiments [14]. First, we evaluated the accuracy of bacterial small RNA target prediction based on 18 *E. coli* small RNAs and 4319 mRNAs. We used 64 sRNA-mRNA pairs with experimentally validated RNA-RNA interactions and all the other pairs in all the sRNA-mRNA combinations as the positive data and the negative data, respectively. Figure 2a shows the ROC-like curves of the two versions of RIblast. The results were almost similar, and this means that there was little difference in the prediction accuracy between the two versions of RIblast. Second, we

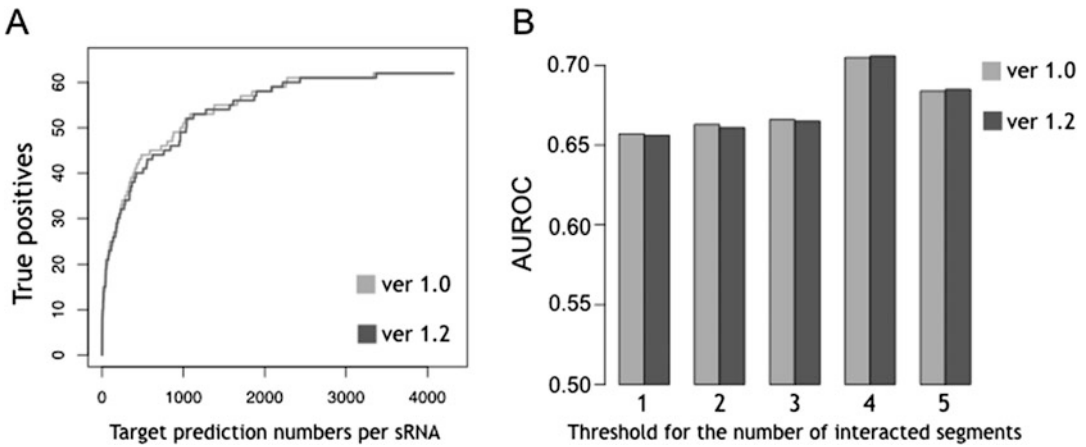


Fig. 2 (a) The prediction accuracy of bacterial small RNA target prediction. The x-axis and y-axis represent target prediction numbers per sRNA and true positives, respectively. The light gray and dark gray lines represent the Riblast versions 1.0 and 1.2, respectively. (b) The prediction accuracy of human lncRNA TINCR target prediction. The x- and y-axes represent the threshold number of interacted segments and the AUROC score, respectively. The light gray and dark gray bars represent the Riblast versions 1.0 and 1.2, respectively

Table 1

The computational times of the two versions of Riblast on the partial human lncRNA and mRNA datasets

	The number of lncRNAs and mRNAs				
Program	5 × 5	10 × 10	50 × 50	100 × 100	500 × 500
Version 1.0	33.16 s	61.18 s	375.31 s	1055.79 s	18205.53 s
Version 1.2	15.11 s	28.72 s	169.8 s	482.88 s	8235.23 s
The speedup ratio	2.19	2.13	2.21	2.19	2.21

Note: The columns represent the numbers of lncRNAs and mRNAs included in the datasets. We created the dataset to ensure that the small datasets were subsets of the large datasets

assessed the human lncRNA target prediction performances using an RIA-seq-based TINCR interaction dataset [3]. In this dataset, we regarded RNAs with more than a threshold number of interaction regions as the positive dataset, and we used a number from 1 to 5 as the threshold values. Here, the positive dataset with a large threshold value is a subset of the positive dataset with a smaller threshold value. Figure 2b shows the AUROC scores, and the results illustrate that the two versions of Riblast have comparable predictive performances. Finally, we evaluated the computational speed of the two versions of Riblast by predicting all-to-all interactions among randomly selected human lncRNAs and mRNAs. Table 1 shows the computational times of the two versions of Riblast for each dataset size. In all cases, version 1.2 was about two-times faster than version 1.0. In summary of the results, version 1.2 of Riblast succeeded in doubling the computational speed while retaining the prediction accuracy.

4 Applications of Fast RNA-RNA Interaction Prediction Methods

RIsearch2 and RIBlast enabled us to conduct biological studies based on the transcriptome-scale RNA-RNA interaction prediction results. In this section, we introduce some application studies using RIsearch2 or RIBlast.

The first is the prediction research of competing endogenous RNAs (ceRNAs) from the genome sequence based on the comprehensive detection of miRNA binding site clusters [30]. CeRNAs, which are also called miRNA sponges, regulate gene expression by binding to miRNAs and preventing the miRNAs from binding to mRNAs. Because ceRNAs with many binding sites have significant effects on gene expression regulation [31], this study hypothesized that transcripts derived from genomic regions with concentrated miRNA binding sites might act as ceRNAs. Here, this study applied RIsearch2 to 2578 human miRNAs and whole human genomes and predicted miRNA binding sites in the whole genome. Note that the genome-scale miRNA binding site prediction would not have been possible without ultrafast RNA-RNA interaction prediction methods. Next, the authors regarded the genomic regions with statistically significant concentrated predicted binding sites as novel ceRNA candidates and predicted 3673 ceRNA candidates, including ciRS-7, which is the experimentally verified ceRNA [32].

The second application study is the discovery of an lncRNA involved in the drug sensitivity of patients suffering from renal cell carcinoma (RCC) [33]. RCC is a type of urological tumor, and sunitinib is the first-line drug for the metastatic RCC patients. However, many patients exhibit resistance to the drug. Therefore, revealing the molecular mechanisms responsible for sunitinib administration is a crucial study for the treatment of this type of cancer. In this study, the authors first discovered that TR4, which is a ligand-activated transcription factor, influences the resistance to sunitinib by altering the expression of AXL, which is a receptor tyrosine kinase. However, as a result of the ChIP-binding assay, the authors revealed that TR4 does not directly regulate the transcription of AXL. Next, the authors hypothesized that TR4 regulates AXL expression via lncRNAs and searched lncRNAs binding to AXL mRNAs using RIBlast from a dataset of lncRNAs co-expressed with AXL. Consequently, the authors discovered lncTASR, which is an lncRNA regulated by TR4 and binds to AXL mRNA. Because the clinical data analysis also showed that patients with high lncTASR expression have significantly shorter survival rates, the authors concluded that TR4/lncTASR/AXL signaling is a vital mechanism of the sunitinib resistance to RCC. This study demonstrated that fast RNA-RNA interaction prediction software is useful in the elucidation of novel molecular regulatory mechanisms of lncRNAs.

Third, we introduce LncRRISearch, a web server for human and mouse lncRNA-RNA interaction predictions [34]. Although RIBlast is faster than the existing programs, a lot of computational costs are required to calculate the interactions between all transcripts exhaustively. Therefore, this study focused on lncRNAs and pre-calculated all human and mouse lncRNA-lncRNA and lncRNA-mRNA interactions using RIBlast. Next, this study constructed the LncRRISearch web server that can access all the prediction results. Additionally, the tissue-specific expression and subcellular localization data of transcripts were integrated with LncRRISearch to enable searching for tissue-specific or subcellular localized lncRNA interactions. This integration is based on studies demonstrating that the tissue-specific expression data enhances the prediction performances of lncRNA-RNA interactions [35]. LncRRISearch is freely available at <http://rtools.cbrc.jp/LncRRISearch/>.

In LncRRISearch, users can select one of the three methods for the analysis method: (1) the analysis of interactions of a specific transcript, (2) the analysis of tissue-specific interactions, and (3) the analysis of subcellular localized interactions. First, for the interaction analysis of a specific transcript, users select the target species and the energy threshold for the interaction prediction and input a Gencode gene/transcript name or ID on the top page (*see* Fig. 3a). If users specified a gene name or ID and not a transcript name or ID, they choose a transcript isoform of the gene on the next page. Next, because all the interacting RNAs for the query transcripts are listed, users select a pair of transcripts of interest. Finally, the details of the RNA-RNA interactions between the selected pair of transcripts are provided. Here, an image of the global interaction pattern and a graphical view for each local base-pairing interaction are provided for the visualization (*see* Fig. 3b, c). Second, for the analysis of tissue-specific interactions, on the top page, users select the target species, the energy threshold for the interaction prediction, the RNA-seq dataset for the analysis of the tissue specificity, the tissue of interest, and the patterns of the tissue specificity, for example, a pattern in which both expressions of two RNAs are specifically increased in the tissue. Next, users select an RNA-RNA pair of interest on the next page, after which the details of the RNA-RNA interactions are provided, like in the first analysis method. Third, for the subcellular localized interaction analysis, users select the energy threshold for the interaction prediction, the cell line, the relative concentration index (RCI), and the threshold of the RCI. Here, RCI was defined as the \log_2 -transformed ratio between the expression levels of two cellular compartments [36]. The subsequent steps are the same as in the second analysis method. Note that LncRRISearch supports only human lncRNA interactions for the analysis of subcellular localized interactions.

A

B



C

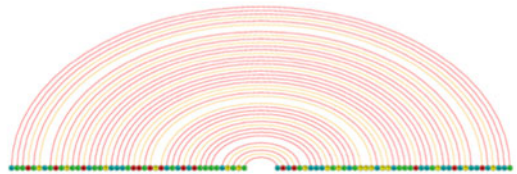


Fig. 3 (a) The top page of LncRRISearch for the analysis of interactions of a specific transcript. (b) A global interaction pattern of the predicted interaction results in LncRRISearch. In this example, the interaction between 7SL lncRNA and TP53 mRNA is shown. The blue and red lines represent the lncRNA and the mRNA, respectively. Additionally, the non-filled and filled red lines represent the UTR and the CDS region of the mRNA, respectively. Both the gray and black lines represent predicted RNA-RNA interactions; the darker the color, the stronger the interaction. (c) A local base-pairing interaction of predicted interaction results in LncRRISearch. The two lines at the bottom of the figure represent two RNA regions, and the elliptical arcs between the two lines represent the predicted base pairs. Different colors of the lines and the arcs represent different bases and base pairs, respectively

5 Conclusion and Future Perspectives

In this review, we focused on two ultrafast RNA-RNA interaction prediction tools, Rlsearch2 and Rlblast. The software acceleration based on the seed-and-extend method enabled us to carry out transcriptome-scale interaction target prediction, and the prediction results can help us in discovering novel functional RNAs or in revealing molecular regulatory mechanisms. In particular, these tools are expected to play a crucial role in the elucidation of the function of lncRNAs, most of which are unknown [37].

Although Rsearch2 and Rblast can predict RNA-RNA interactions with high speed and accuracy, the development of faster or more accurate software programs is an essential issue. One of the promising methods for software acceleration is the utilization of parallel computing. Currently, many sequence alignment tools based on various parallelization methods [38–39] have been developed. The application of these methods for predicting RNA-RNA interactions would be an interesting research topic. On the other hand, as a method for improving the prediction accuracy, the incorporation of a co-transcriptional folding model to the accessibility calculation [40–41] or the development of a machine learning-based prediction method using experimental detection results [3–7] may be a promising approach.

References

1. Gebert LFR, MacRae IJ (2019) Regulation of microRNA function in animals. *Nat Rev Mol Cell Biol* 20(1):21–37
2. Abdelmohsen K, Panda AC, Kang MJ et al (2014) 7SL RNA represses p53 translation by competing with HuR. *Nucleic Acids Res* 42(15):10099–10111
3. Kretz M, Sibrashvili Z, Chu C et al (2013) Control of somatic tissue differentiation by the long non-coding RNA TINCR. *Nature* 493(7431):231–235
4. Engreitz JM, Sirokman K, McDonel P et al (2014) RNA-RNA interactions enable specific targeting of noncoding RNAs to nascent pre-mRNAs and chromatin sites. *Cell* 159(1):188–199
5. Lu Z, Zhang QC, Lee B et al (2016) RNA duplex map in living cells reveals higher-order transcriptome structure. *Cell* 165(5):1267–1279
6. Nguyen TC, Cao X, Yu P et al (2016) Mapping RNA-RNA interactome and RNA structure in vivo by MARIO. *Nat Commun* 7:12023
7. Cai Z, Cao C, Ji L et al (2020) RIC-seq for global in situ profiling of RNA-RNA spatial interactions. *Nature* 582(7812):432–437
8. Iyer MK, Niknafs YS, Malik R et al (2015) The landscape of long noncoding RNAs in the human transcriptome. *Nat Genet* 47(3):199–208
9. Mann M, Wright PR, Backofen R (2017) IntaRNA 2.0: enhanced and customizable prediction of RNA-RNA interactions. *Nucleic Acids Res* 45(W1):W435–W439
10. Tafer H, Amman F, Eggenhofer F et al (2011) Fast accessibility-based prediction of RNA-RNA interactions. *Bioinformatics* 27(14):1934–1940
11. Wenzel A, Akbasli E, Gorodkin J (2012) Rsearch: fast RNA-RNA interaction search using a simplified nearest-neighbor energy model. *Bioinformatics* 28(21):2738–2746
12. Kato Y, Sato K, Hamada M et al (2010) RactIP: fast and accurate prediction of RNA-RNA interaction using integer programming. *Bioinformatics* 26(18):i460–i466
13. Alkan F, Wenzel A, Palasca O et al (2017) Rsearch2: suffix array-based large-scale prediction of RNA-RNA interactions and siRNA off-targets. *Nucleic Acids Res* 45(8):e60
14. Fukunaga T, Hamada M (2017) Rblast: an ultrafast RNA-RNA interaction prediction system based on a seed-and-extension approach. *Bioinformatics* 33(17):2666–2674
15. Backofen R (2014) Computational prediction of RNA-RNA interactions. *Methods Mol Biol* 1097:417–435
16. Lai D, Meyer IM (2016) A comprehensive comparison of general RNA-RNA interaction prediction methods. *Nucleic Acids Res* 44(7):e61
17. Umu SU, Gardner PP (2017) A comprehensive benchmark of RNA-RNA interaction prediction tools for all domains of life. *Bioinformatics* 33(7):988–996
18. Turner DH, Mathews DH (2010) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary

- structure. *Nucleic Acids Res* 38(Database: issue):D280–D282
19. Bernhart SH, Mückstein U, Hofacker IL (2011) RNA accessibility in cubic time. *Algorithms Mol Biol* 6(1):3
 20. Kiryu H, Terai G, Imamura O et al (2011) A detailed investigation of accessibilities around target sites of siRNAs and miRNAs. *Bioinformatics* 27(13):1788–1797
 21. Altschul SF, Gish W, Miller W et al (1990) Basic local alignment search tool. *J Mol Biol* 215(3):403–410
 22. Kielbasa SM, Wan R, Sato K et al (2011) Adaptive seeds tame genomic sequence comparison. *Genome Res* 21(3):487–493
 23. Suzuki S, Kakuta M, Ishida T et al (2014) GHOSTX: an improved sequence homology search algorithm using a query suffix array and a database suffix array. *PLoS One* 9(8): e103833
 24. Andronescu M, Condon A, Hoos HH et al (2010) Computational approaches for RNA energy parameter estimation. *RNA* 16(12): 2304–2318
 25. Antonov IV, Mazurov E, Borodovsky M, Medvedeva YA (2019) Prediction of lncRNAs and their interactions with nucleic acids: benchmarking bioinformatics tools. *Brief Bioinform* 20(2):551–564
 26. Fukunaga T, Hamada M (2018) A novel method for assessing the statistical significance of RNA-RNA interactions between two long RNAs. *J Comput Biol* 25(9):976–986
 27. Rehmsmeier M, Steffen P, Hochsmann M, Giegerich R (2004) Fast and effective prediction of microRNA/target duplexes. *RNA* 10(10):1507–1517
 28. Wright PR, Richter AS, Papenfort K et al (2013) Comparative genomics boosts target prediction for bacterial small RNAs. *Proc Natl Acad Sci U S A* 110(37):E3487–E3496
 29. <https://github.com/herumi/fmath/blob/master/fmath.hpp>
 30. Pan X, Wenzel A, Jensen LJ, Gorodkin J (2018) Genome-wide identification of clusters of predicted microRNA binding sites as microRNA sponge candidates. *PLoS One* 13(8): e0202369
 31. Denzler R, Agarwal V, Stefano J et al (2014) Assessing the ceRNA hypothesis with quantitative measurements of miRNA and target abundance. *Mol Cell* 54(5):766–776
 32. Hansen TB, Jensen TI, Clausen BH et al (2013) Natural RNA circles function as efficient microRNA sponges. *Nature* 495(7441): 384–388
 33. Shi H, Sun Y, He M et al (2020) Targeting the TR4 nuclear receptor-mediated lncTASR/AXL signaling with tretinoin increases the Sunitinib sensitivity to better suppress the RCC progression. *Oncogene* 39(3):530–545
 34. Fukunaga T, Iwakiri J, Ono Y, Hamada M (2019) LncRRISearch: a web server for lncRNA-RNA interaction prediction integrated with tissue-specific expression and subcellular localization data. *Front Genet* 10: 462
 35. Iwakiri J, Terai G, Hamada M (2017) Computational prediction of lncRNA-mRNA interactions by integrating tissue specificity in human transcriptome. *Biol Direct* 12(1):15
 36. Mas-Ponte D, Carlevaro-Fita J, Palumbo E et al (2017) LncATLAS database for subcellular localization of long noncoding RNAs. *RNA* 23(7):1080–1087
 37. de Hoon M, Shin JW, Carninci P (2015) Paradigm shifts in genomics through the FANTOM projects. *Mamm Genome* 26(9-10): 391–402
 38. Suzuki S, Kakuta M, Ishida T, Akiyama Y (2016) GPU-acceleration of sequence homology searches with database subsequence clustering. *PLoS One* 11(8):e0157338
 39. Suzuki H, Kasahara M (2018) Introducing difference recurrence relations for faster semi-global alignment of long sequences. *BMC Bioinformatics* 19(Suppl 1):45
 40. Proctor JR, Meyer IM (2013) COFOLD: an RNA secondary structure prediction method that takes co-transcriptional folding into account. *Nucleic Acids Res* 41(9):e102
 41. Fukunaga T, Hamada M (2018) Computational approaches for alternative and transient secondary structures of ribonucleic acids. *Brief Funct Genomics* 18(3):182–191



Chapter 11

Web Services for RNA-RNA Interaction Prediction

Tsukasa Fukunaga, Junichi Iwakiri, and Michiaki Hamada

Abstract

Non-coding RNAs have various biological functions such as translational regulation, and RNA-RNA interactions play essential roles in the mechanisms of action of these RNAs. Therefore, RNA-RNA interaction prediction is an important problem in bioinformatics, and many tools have been developed for the computational prediction of RNA-RNA interactions. In addition to the development of novel algorithms with high accuracy, the development and maintenance of web services is essential for enhancing usability by experimental biologists. In this review, we survey web services for RNA-RNA interaction predictions and introduce how to use primary web services. We present various prediction tools, including general interaction prediction tools, prediction tools for specific RNA classes, and RNA-RNA interaction-based RNA design tools. Additionally, we discuss the future perspectives of the development of RNA-RNA interaction prediction tools and the sustainability of web services.

Key words RNA-RNA interaction, Web service, Non-coding RNA, miRNA, RNA design

1 Introduction

Many recent studies have suggested that non-coding RNAs (ncRNAs) play a central role in various cellular functions such as epigenetic, transcriptional, and translational regulatory mechanisms. Functional biomolecules, including these ncRNAs, do not function independently but cooperate with other molecules to achieve their functions. To clarify the molecular mechanism of functional RNAs, the identification of interacting partners (e.g., DNA, RNA, and protein) of these RNAs is necessary. Several functionally important RNA-RNA interactions (RRIs), have been observed in translational regulation [1–4], liquid-liquid phase separation (LLPS) [5], and others [6–9]. Moreover, RRIs play an important role in RNA editing using CRISPR-Cas13 [10, 11].

A unique feature of RRIs compared with RNA-protein interactions is that it is easier to achieve specificity for the target RNA. This is because RRIs are capable of base-pair-based interactions

(cf. G:C and A:U for Watson-Crick base pairs, G:U for Wobble base-pair). Short ncRNAs, such as microRNAs (miRNAs) [1], piwi-interacting RNAs (piRNAs) [9], small nucleolar RNAs (snoRNAs) [9], and the CRISPR-Cas13 system, take advantage of this feature to serve as a rigorous guide for the target RNA.

Various experimental methods have been proposed to identify RRIs [12–14]; however, their detection sensitivity is still limited. In particular, ncRNAs have low expression levels and tissue specificity [15], and it is, therefore, difficult to identify ncRNA-RNA interactions experimentally. In contrast, computational approaches enable unbiased and comprehensive identification of RRIs (cf. [16]). In the computational prediction of RRIs, external base pairs as well as internal base pairs should be considered. Hence, similar to RNA secondary structure predictions, the prediction of RRIs is highly compatible with computational approaches. As a result, many studies related to RRIs have been performed, for example, [16–18].

Because of the biological importance of RRIs and the advent of relevant computational approaches, many web services for RRI predictions, which can be easily used by experimental biologists, have been developed. In this article, we survey web servers closely related to RRIs, including general RRI prediction, small RNA (sRNA) target prediction, miRNA target prediction, and RRI-based RNA design.

2 Web Services for RNA-RNA Interaction Prediction

Table 1 is a comprehensive list of RRI prediction web services that are currently available. There are four primary types of prediction tools. The first type is a general RRI prediction tool. These tools do not assume the input RNA class and predict interactions based on an RNA energy model or evolutionarily conserved pattern. These tools can be further divided into two types: joint structure prediction and local (global) interaction prediction. The former predicts both base pairs within individual RNAs and between two RNAs simultaneously, whereas the latter predicts only base pairs between two RNAs. These tools have been mainly used to predict the targets of microbial sRNAs. The second type is the miRNA target prediction tool. miRNAs are small ncRNAs that are widely conserved in eukaryotes and repress the expression of mRNAs by interacting with the 3'-UTR of the target mRNAs [19]. Therefore, miRNAs have attracted attention as important molecules for post-transcriptional regulation, and their physical and biochemical binding mechanisms have been experimentally revealed. The second type of tool predicts interactions by utilizing knowledge of the binding mechanisms. The third type is the RNA design tools for CRISPR gene editing, RNAi, and PCR. These biotechnologies require nucleic acids that complementarily bind to the target

Table 1
Comprehensive list of RNA-RNA interaction prediction web services

Methods	Keywords	Ref.
<i>General prediction/sRNA target prediction tools</i>		
PETcofold	Joint, Comparative	[39]
Rtips	Joint, Integer programming	[40]
CopraRNA	Local, Comparative, Accessibility	[18]
IntaRNA	Local, Accessibility	[22]
RNAcofold	Joint, Concatenation	[26]
RNAup	Local, Accessibility	[26]
RNApredator	Local, Accessibility	[41]
TargetRNA2	Local, Comparative, Accessibility	[42]
DuplexFold	Global	[43]
BiFold	Joint, Concatenation	[43]
VfoldCPX	Joint, Concatenation	[44]
<i>miRNA target prediction tools</i>		
metaMIR	Ensemble	[45]
TargetScan	Comparative, Accessibility	[30]
DIANA-tools	Comparative, Accessibility	[46]
RNA22	Pattern match	[47]
ComTAR	Plant miRNAs, Comparative	[48]
miRror	Emsemble	[49]
PACCMIT/PACCMIT-CDS	Comparative, Accessibility	[50]
ToppMiR	Ensemble	[51]
<i>RNA design tools</i>		
CRISPRdirect	gRNA design	[34]
CRISPOR	gRNA design	[52]
CHOPCHOP	gRNA design	[53]
CCTOP	gRNA design	[54]
siDirect	siRNA design	[55]
Oligowalk	siRNA design	[56]
Primer3	PCR primer design	[57]
MODENA web server	RRI switch design	[58]

(continued)

Table 1
(continued)

Methods	Keywords	Ref.
<i>The others</i>		
LncRRIssearch	LncRNAs	[36]
RILogo	Visualization of RRIs	[59]
psRNATarget	Plant small ncRNAs, Accessibility	[60]
Snoscan	C/D box snoRNAs	[32]
snoGPS	H/ACA snoRNAs	[33]
pirScan	<i>C. elegans</i> piRNAs	[61]

“Joint” means joint structure prediction tools, and “Local” (“Global”) means local (global) interaction prediction tools. “Comparative” means that the methods use the comparative genomic approach. “Ensemble” means ensemble tools of multiple prediction methods

nucleic acids, and the third type of tool helps users design nucleic acids that efficiently bind to the targets. We included design tools for a wide range of nucleic acid interactions not limited to RRIs. Note that we only introduced a few representative tools in this list because there are too many tools in this class. The fourth type comprises the remaining tools, which mainly include prediction tools for specific RNA classes other than miRNAs. In this review, we introduce only representative web services in detail.

2.1 *IntaRNA* and *CopraRNA*

Freiburg RNA tools are web services of RNA informatics software developed by Professor Backofen’s group at the University of Freiburg [20]. These web services include various analysis tools, such as for RNA structural alignments, RNA sequence design, and RNA-RNA interactions. We focused on two RRI prediction tools, *IntaRNA* [21, 22] and *CopraRNA* [18, 23].

IntaRNA is of the first type of local interaction prediction tools for general RRI prediction. *IntaRNA* predicts interactions based on two scores: hybridization energy and accessibility. The hybridization energy is the stabilization energy derived from the hybridization between two RNAs. Accessibility represents the difficulty of forming internal RNA secondary structures, that is, the ease of forming interactions with external RNAs. Recent benchmarking studies revealed that the methods based on these two scores have a high prediction performance, and *IntaRNA* is one of the state-of-the-art general RRI prediction tools [17, 24]. However, it should be noted that *IntaRNA* can only be applied to short RNAs because of its long computation time. *CopraRNA* predicts microbial sRNA targets by integrating the prediction results of *IntaRNA* for multiple genome sequences. Using the comparative genomic approach, *CopraRNA* shows an even higher prediction performance than *IntaRNA* [25].

The main inputs of IntaRNA are two types of RNAs: query RNAs and target RNAs. Queries should be short RNAs in the multi-FASTA format, whereas targets may be as long as several hundred bases. Users can specify the target RNAs using either the multi-FASTA format or the Refseq ID of a prokaryotic genome. In the latter case, as IntaRNA cannot handle the whole genome as input, RNA sequences around the start or stop codons are regarded as the target RNAs (the details are user-configurable). As optional inputs, the web service allows users to adjust various runtime parameters, such as the number of interactions per RNA pair, the maximal interaction length, the handling of G-U wobble base pairs, and the energy model for RNA secondary structure analysis.

IntaRNA outputs are provided in various forms. The most important result is a detailed description of the predicted interaction, which is composed of the interacting regions, base pairs, and score of the interaction (Fig. 1a). When users specify the target RNAs using the multi-FASTA format, figures of position-wise minimal energy profiles are provided that show the RNA regions likely to interact with each other. Figure 1b is an example figure and shows that the target RNA has two sites interacting with the query RNA. When users specify the target RNAs using the Refseq ID, a summarized figure of the interacting regions in the query and target RNAs is provided. Figure 1c shows the prediction results when the query is ChiX and the targets are RNAs around the start codon of *E. coli K-12 MG1655*. The figure suggests that ChiX has a specific interacting region and tends to bind regions slightly upstream of the start codon. Additionally, a heatmap of the functional enrichment analysis of the predicted targets is provided.

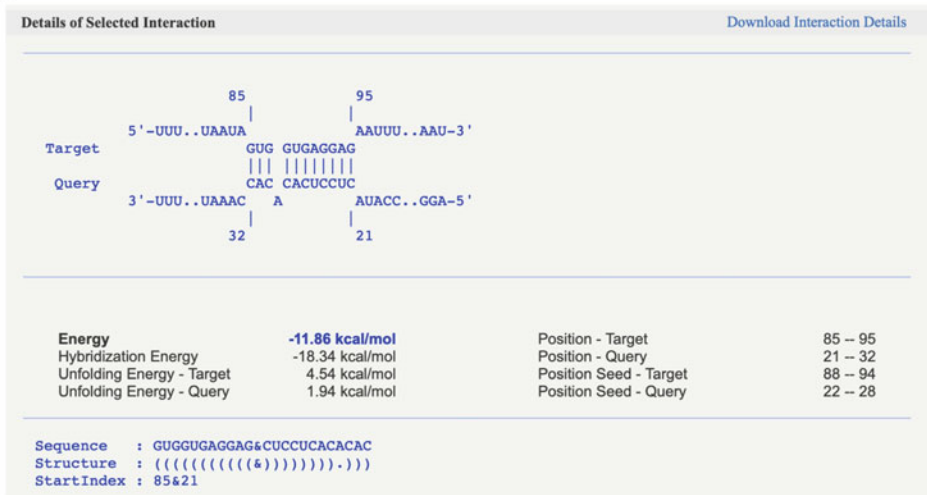
In CopraRNA, only queries in the multi-FASTA format (similar to IntaRNA) and targets as Refseq IDs are accepted as inputs. Users need to specify from three to eight Refseq IDs for comparative genome analysis and select one organism as the primary target. The outputs of the web service are similar to those of the IntaRNA.

2.2 RNAup and RNACofold

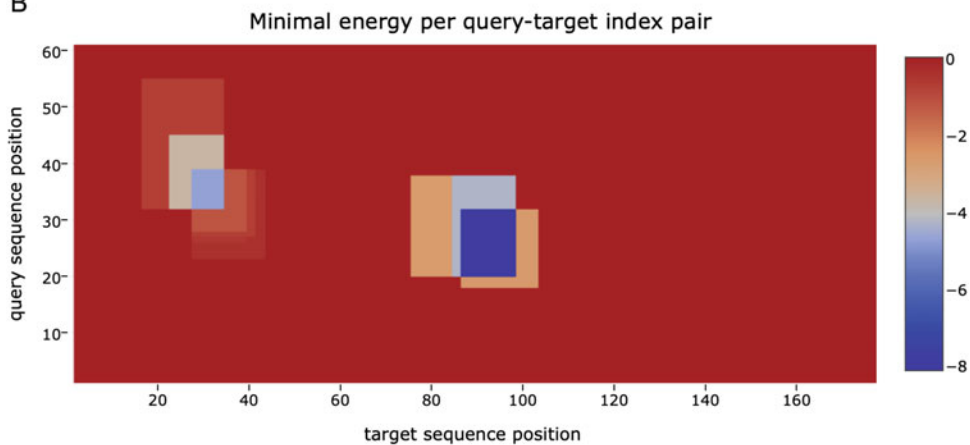
The Vienna RNA Websuite is a web service of RNA secondary structure analysis software developed by Professor Hofacker's group at the University of Vienna [26]. This web service also includes various software tools, such for as secondary structure prediction, RNA discovery, and folding kinetics analysis. Here, we focused on two RRI prediction tools, RNAup [27] and RNACofold [28].

RNAup is a local interaction prediction tool, whereas RNACofold is a joint structure prediction tool. RNAup utilizes hybridization energies and accessibilities for prediction, as with IntaRNA. Therefore, RNAup has a very high prediction performance comparable to that of IntaRNA but requires a much longer computation time [17, 24]. RNACofold predicts the interactions using a concatenation-based algorithm, which calculates a minimum free

A



B



C

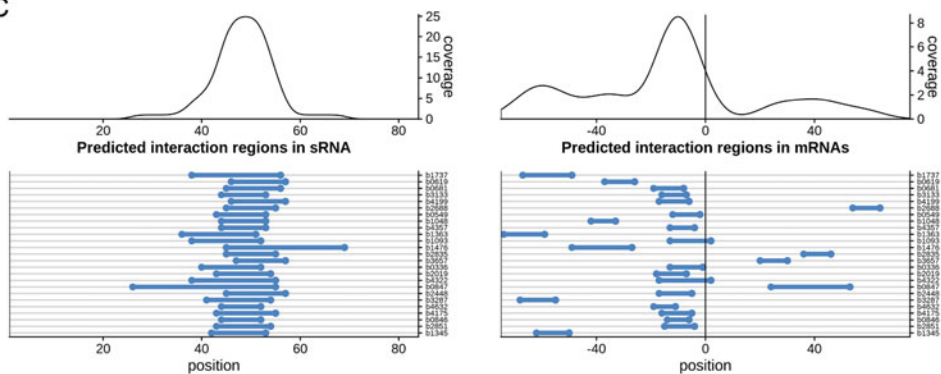


Fig. 1 Prediction results of IntaRNA. (a) The main result of IntaRNA prediction is composed of the interacting regions, base pairs, and interaction score. (b) Results of position-wise minimal energy profiles. Different colors indicate regions that are more likely to bind. (c) Prediction results for ChiX and RNAs around the start codon of *E. coli* K-12 MG1655. The left and right figures represent the sRNA and mRNA interaction regions, respectively. The lower figures represent interactions for each mRNA, and the horizontal and bold lines indicate each mRNA and predicted interaction, respectively. The upper figures represent a summary of all the predicted mRNA interactions. Position 0 to the right of figures indicates the start codon

A

RNAcofold WebServer 1 Enter Input Parameters 2 View Results

[\[Home|New job|Help\]](#)

The **RNAcofold web server** will predict secondary structures of single stranded RNA or DNA sequences upon dimer formation.

Simply paste or upload your sequences below and click *Proceed*. To get more information on the meaning of the options click the symbols. You can test the server using [these sample sequences](#).

Paste or type your **first sequence** here: [\[clear\]](#)

Or upload a file containing the **first sequence** in FASTA format: no file selected

Paste or type your **second sequence** here: [\[clear\]](#)

Or upload a file containing the **second sequence** in FASTA format: no file selected

B

RNAcofold WebServer 1 Enter Input Parameters 2 View Results

[\[Home|New job|Help\]](#)

Results for minimum free energy prediction

The optimal secondary structure of the hetero-dimer in dot-bracket notation with a minimum free energy of **-6.10 kcal/mol** is given below. "&" separates the two sequences and the two structures.

```
ACGAUCAGAGAUACAGAGCAUACGACAGCAG&ACGAAAAAAGAGCAUACGACAGCAG
..(((.....)))...(((.....(.....&.....))).....)).....
```

Results for thermodynamic ensemble prediction

The free energy of the thermodynamic ensemble is **-6.41 kcal/mol**.

The frequency of the MFE structure in the ensemble is **60.50 %**.

Delta G for heterodimer binding is **-2.29 kcal/mol**.

Fig. 2 Interaction prediction of two RNAs using RNAcofold. **(a)** User needs to input two RNA sequences in the single-FASTA format. **(b)** Prediction results of RNAcofold

energy structure for concatenated sequences of two input RNAs. RNAcofold shows a faster prediction speed and a lower prediction performance than IntaRNA and RNAup.

RNAup accepts two input RNA sequences in the single-FASTA format only. RNAup also allows users to set some optional parameters, such as the length of the unstructured region and the handling of isolated base pairs. The main result of RNAup is the predicted interaction with binding energy. Additionally, figures showing position-wise hybridization energies and position-wise accessibilities for the longer sequence are supplied.

RNAcofold also accepts two input RNAs in the single-FASTA format only and allows users to set some optional parameters (Fig. 2a). The characteristics of RNAcofold, including predicted interactions between RNAs and internal structures of two RNAs, are shown in Fig. 2b. Additionally, the base-pairing probabilities of two RNAs are provided as output [29].

2.3 TargetScan

TargetScan is an miRNA target prediction tool for several animal species [30]. Although miRNAs are conserved throughout eukaryotes, their mechanisms of action differ between plants and animals. For miRNAs to function in animals, the full lengths do not need to be complementary to the target regions, but 6–8 base seed regions must form completely complementary regions. Here, the seed region refers to the region starting from the second nucleotide position from the 5′-end of the miRNA. In addition to the interaction in the seed regions, TargetScan uses a conservation index as a useful feature for interaction prediction because miRNA-mRNA interactions are widely conserved among species. Furthermore, TargetScan uses 14 RNA features, such as accessibility, 3′-UTR lengths, and AU contents near the interaction site, for highly accurate prediction. These features were selected using a stepwise regression model based on 74 microarray datasets.

To use TargetScan, users must first select the target species. At present, it supports 13 animals, including human, mouse, rat, chimpanzee, rhesus, cow, dog, opossum, chicken, frog, worm, fly, and zebrafish. Next, the users specify a target RNA or miRNA. A target gene/transcript is specified using the gene symbol or the ENSEMBL gene/transcript ID. A target miRNA is specified by directly entering its name or selecting the name from a tab. These are all user inputs, making them simple and easy to use.

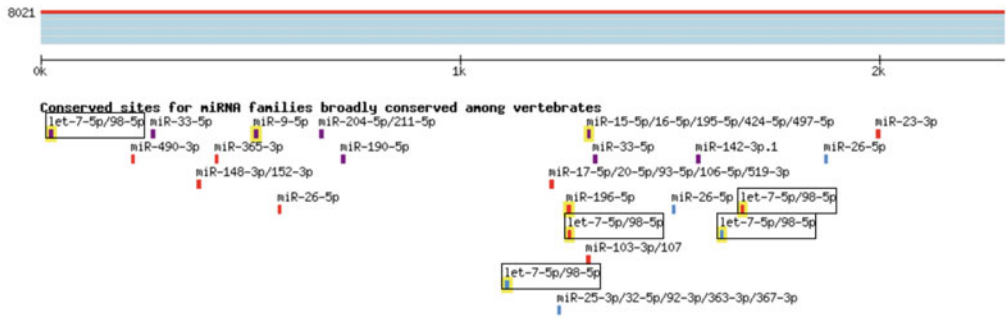
When users specify a target RNA as input, TargetScan outputs a list of miRNAs predicted to bind to the 3′-UTR of the transcript. Figure 3a shows an example result for ENST00000403681.2 (a transcript of the human HMGA2 gene). Additionally, multiple alignments of the transcripts among species are displayed to confirm the conservation of the interacting regions of the seed regions (Fig. 3b). If the seed region is conserved, the background color of the alignment changes, and users can quickly identify the species that have no conserved seed regions. For example, Fig. 3b shows that the target region of miR-26-5p in ENST00000403681.2 is not conserved in brown bats, lizards, and *Xenopus tropicalis* (western clawed frog). When users specify an miRNA as the input, TargetScan outputs a list of predicted target transcripts (Fig. 3c). Because the list includes hyperlinks to the Ensemble annotation page for the corresponding genes, users can readily investigate the target gene functions.

2.4 Snoscan/snoGPS

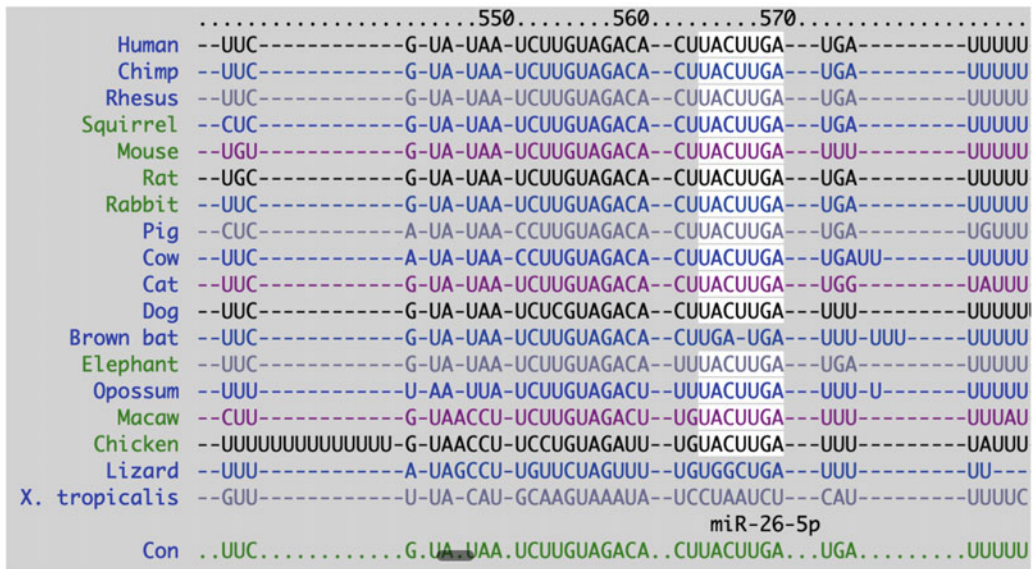
snoRNAs are a family of small ncRNAs that mediate several types of RNA modification for other non-coding/coding RNAs via base-pairing RRIs with their target RNAs. Two types of RNA modification, ribose 2′-O methylation for every nucleotide and pseudouridylation of uridine, have been investigated over the past decades [31]. snoRNAs interact with several proteins, including modification enzymes such as methyltransferases and pseudouridine synthetase, comprising snoRNPs (small nucleolar ribonucleoprotein

A

Human HMG2A2 ENST00000403681.2 3' UTR length: 3003



B



C

Target gene	Representative transcript	Gene name	Number of 3P-seq tags supporting UTR + 5	Link to sites in UTRs	Conserved sites				Poorly conserved sites			
					total	8mer	7mer-m8	7mer-A1	total	8mer	7mer-m8	7mer-A1
ONECUT2	ENST00000491143.2	one cut homeobox 2	847	Sites in UTR	13	8	2	3	7	0	4	3
YBX3	ENST00000279550.7	Y box binding protein 3	631	Sites in UTR	2	2	0	0	0	0	0	0
LYVE1	ENST00000256178.3	lymphatic vessel endothelial hyaluronan receptor 1	5	Sites in UTR	1	1	0	0	3	1	1	1
POU2F1	ENST00000367866.2	POU class 2 homeobox 1	83	Sites in UTR	6	4	2	0	3	0	2	1
ONECUT1	ENST00000560699.2	one cut homeobox 1	10	Sites in UTR	13	8	4	1	3	0	1	2
SLC50A1	ENST00000368404.4	solute carrier family 50 (sugar efflux transporter), member 1	3317	Sites in UTR	2	2	0	0	0	0	0	0
POU6F2	ENST00000518318.2	POU class 6 homeobox 2	5	Sites in UTR	4	0	3	1	0	0	0	0
TRPM7	ENST00000560955.1	transient receptor potential cation channel, subfamily M, member 7	281	Sites in UTR	2	0	1	1	0	0	0	0
AP1S2	ENST00000329235.2	adaptor-related protein complex 1, sigma 2 subunit	357	Sites in UTR	2	1	0	1	0	0	0	0
PDK4	ENST00000005178.5	pyruvate dehydrogenase kinase, isozyme 4	5549	Sites in UTR	1	1	0	0	0	0	0	0

Fig. 3 Prediction results of TargetScan. (a) A list of miRNAs predicted to bind to the 3'-UTR of ENST00000403681.2, a human HMG2A2 transcript. (b) Multiple alignment of transcripts among species. White indicates a conserved seed region. (c) List of predicted target transcripts of human miR-9-5p

complex). The major target RNAs for snoRNA-mediated modification are four ribosomal RNAs (rRNAs), in which more than 100 modification sites have been identified.

snoRNAs are categorized into the following two types: box C/D type (ribose 2'-O methylation) and box H/ACA type (pseudouridylation). Box C/D snoRNAs are characterized by their conserved sequence motifs and secondary structures. The secondary structure of box C/D snoRNA is a stem-loop with four conserved sequences (called box C, C', D, D') in the loop region as follows: 5'-(box C:AUGAUGA)-[guide sequence1]-(box D':CUGA)-(box C':UGAUGA)-[guide sequence2]-(box D:CUGA)-3'. The guide sequences are complementary to the target sites of rRNAs, in which the methylation sites are specified by a 5-nt upstream position from box D/D'. Box H/ACA snoRNAs consist of two hairpin structures connected with linker sequences containing two conserved sequences called (box H and ACA) as follows: 5'-(hairpin1 with guide sequence1)-[linker with box H:ANANNA]-(hairpin2 with guide sequence2)-[box ACA: ACA]-3'. The target sequences are located at the internal loop regions of hairpin structures and are complementary to the target site of rRNAs, guiding precise pseudouridylation.

Snoscan [32] is a web service for the prediction of box C/D type snoRNAs and modification of target sites in rRNAs based on the detection of box C and D sequences, secondary structure, and RRI between the snoRNA and rRNA. Users need to prepare candidate snoRNA sequences and target RNA sequences as input data for Snoscan. For the target RNA sequences, Snoscan also provides rRNA sequences of several species, such as human, mouse, fly, and worm, for the alternative input data. Thus, target RNA sequences are not necessary for these species. In addition, users need to specify the phylogenetic group (Mammalian, Yeast, Archaea) to apply an appropriate prediction model. Users can also specify several search options, such as the length of RRI between snoRNA and rRNA and distances between boxes C and D, to control the prediction results (Fig. 4a). The prediction results of Snoscan contain several kinds of information, such as RNA-RNA base-pairing interaction between the snoRNA and the target RNA, the position of box sequences, stem region of the snoRNA secondary structure, and the methylation site/position in the target RNA sequence (Fig. 4b).

snoGPS [33] is also a web service for predicting H/ACA snoRNAs and modification sites in target RNAs based on the secondary structures and RRI between these RNAs and the box sequences. For the input data of snoGPS, users need to prepare the candidate H/ACA snoRNA sequences and target RNA sequences. snoGPS also provides rRNA sequences only for human and yeast as alternative input data. In addition, users also need to specify the phylogenetic group (human, yeast, and archaea) of the input sequence. An important difference between snoGPS and Snoscan

A

Probabilistic Search Model:

Mammalian (good default) ▾

Query Sequence

Format:

- Raw Sequence
- Sequence name (optional): (no spaces)
- Other (FASTA, GenBank, EMBL, GCG, IG)

Paste your query sequence(s) here:

```
>snoRNA_candidate
sstcaatgatstgttscatattatctaatctattactatataataaacacitt
agctcfaaattactctaaacct
```

(The web-server may experience problems if submitted queries total more than 100K nucleotides at any one time)

[Sample yeast snoRNA sequences](#)

Clear Sequence

OR submit a file: ファイルが選択されていません。

- Show results in this browser, OR
- Receive results by e-mail instead:

Run Snoscan!

Target RNA Sequence (e.g. rRNA)

Format:

- Raw Sequence
- Sequence name (optional): (no spaces)
- Other (FASTA, GenBank, EMBL, GCG, IG)

Paste your target RNA sequence(s) here:

(The web-server may experience problems if submitted targets total more than 10K nucleotides at any one time)

Clear Sequence

OR submit a file: ファイルが選択されていません。

OR use a provided target RNA sequence:

Optional: Specify positions of known or predicted sites of methylation

Methylation file: ファイルが選択されていません。

[Methylation file format](#) | [Yeast methylation file](#) | [Human methylation file](#)

B

```
snoscan Results

Please cite:
Lowe, T.M. & Eddy, S.R. (1999) "A computational screen for
methylation guide snoRNAs in yeast", Science 283: 1168-1171.

-----

Snoscan (v.1.0) search results:
=====
Cutoff: (bits)
C Box: 6.49 D'Box: -2.02 Compl score: 8.0 Final score: 12.0
Min Match: 9 bp Max mismatch: 2 bp
Max C-D Box Dist: 110 bp Min D box-match dist w/D' Box present: 10 bp
-----

snoRNA Hit Information (Jump to Sequence Listing)

>> snoRNA_candidate1 20.34 (2-80) Compl: Hu-185-Am159 (-) 16/0 bp Gs-D box: 58 (57) Len: 79 TS
No known meth site found Guide Seq Sc: 5.25 (29.07 -7.14 -15.69 -1.00)

+
D' seq: 5'- GUAUUCUAGAACUAA -3' Hu-185 (156-173)
Qry seq: 3'- AUCUUAUAUAUCUUAUU -5' snoRNA_candidate1 (73-88)
Terminal stem:
+---[C Box] -W- CUGG - 5' Stem Sc: 2.80 (4 bp)
| | | |
+---[D Box] - GACCU - 3' Stem Transit Sc: -0.63

>>Summary [ C Box ] -- -- [ Compl/ Mism ] X [ D Bx ] Length
>Meth Am 159 [AUGAUU] -- 49 bp -- [ 16 / 0 ] 1 [CUGA] 79 bp
>Sc 20.34 [ 9.81 ] -- -2.60 -- [ 29.07 bits ] [3.05]

Candidate sequence:
>snoRNA_candidate1 20.34 (2-80) Compl: Hu-185-Am159 Len: 79
GTCAATGCTGTGTGTCATATATCTGAATCTATTCTGATGTTGTAATAACACTTAG
CTUTAAAGTAATCTGAGA
```

Fig. 4 Prediction of box C/D snoRNA using Snoscan. (a) The user needs to input candidate snoRNA sequences (as query sequences) and target RNA sequences. For the target RNA sequences, the user can select the rRNA sequences of several species provided by Snoscan. (b) Prediction results for Snoscan

is that users need to specify the search mode (one or two stems) for snoGPS. The “one stem” mode is useful for several archaeal snoRNAs that comprise a single hairpin, whereas “two stem” mode is useful for predicting eukaryotic snoRNAs consisting of two hairpins (Fig. 5a). The prediction results of snoGPS provide the following information: pseudouridylation sites in the target RNAs,

A

Source:

Scanner:
 Two stem
 One stem

Query Sequence

Format:

Raw Sequence
 Sequence name (optional): (no spaces)
 Other (FASTA, GenBank, EMBL, GCG, IG)

Paste your query sequence(s) here:

```
>SNORA21
cccccttttaaaagcactcaatggccctgtggcctaatgaacctattgagccgtcaagaag
ggagaaatgaaacatcacttttggatgaatgaacacatattgtttctcaatcag
tcatatcaagaa
```

(The web-server may experience problems if submitted queries total more than 100K nucleotides at any one time)

or submit a file: ファイルが選択されていません.

To try out the server, you can use this [sample human H/ACA snoRNA sequence.](#)

Target Sequence

Choose a target sequence:

...OR input your own sequence.

Format:

Raw Sequence
 Sequence name (optional): (no spaces)
 Other (FASTA, GenBank, EMBL, GCG, IG)

Paste your target sequence(s) here:

(The web-server may experience problems if submitted queries total more than 100K nucleotides at any one time)

or submit a file: ファイルが選択されていません.

Enter your own [pseudouridine file](#): ファイルが選択されていません.

Show results in this browser.
 Receive results by e-mail instead:

Output Sorting Options:

Sorting method: Output only top hits per pseudoU site Remove lower-scoring overlapping candidates Check positive strand only Guide region score cutoff:

Final Score cutoff:

B

Results

snoRNA Hits

```
>SNORA21.12 48.83 (0-131) Cmp: H_sapiens_LSU.U4460 Nd Pairs: 11/0/6/8/C 132 bp (H)
>SNORA21.10 43.15 (0-131) Cmp: H_sapiens_LSU.U4391 Nd Pairs: 10/0/9/8/H 132 bp (H)
```

snoRNA Hits' Sequences

[Go back to snoRNA Hits](#)



Fig. 5 Prediction of box H/ACA snoRNA using snoGPS. (a) The user needs to input candidate snoRNA sequences (as query sequences) and target RNA sequences. For the target RNA sequences, the user can select the rRNA sequences of several species provided by snoGPS. (b) Prediction results for snoGPS

RRI between the snoRNA and the target RNA (indicated as LLLLL or RRRRR sequences shown in the bottom/next line of the snoRNA sequence), hairpin structures, and box sequences (Fig. 5b).

2.5 CRISPRdirect

The clustered regularly interspaced short palindromic repeat (CRISPR)-CRISPR associated protein 9 (Cas9) system is one of the most popular tools for genome DNA editing in biological research. In particular, the CRISPR-Cas9 system is frequently used in gene knockout experiments. In the CRISPR-Cas9 system, a single guide RNA (gRNA) is designed for site-specific DNA editing based on RNA-DNA base-pairing interactions, which leads to precise DNA cleavage by the Cas9 endonuclease. The length of the gRNA is approximately 100 nt, in which the first 20 nt sequence (5'-end) needs to be complementary to the target DNA sequence. The target DNA sequences should be adjacent to the 5'-NGG motif (NGG for Cas9 derived from *Streptococcus pyogenes*, NNGRRT for Cas9 derived from *Staphylococcus aureus*) called protospacer adjacent motif (PAM), which is recognized and cleaved by Cas9 at a site 3 nt upstream of the sequence.

However, one of the major concerns of the CRISPR-Cas9 system is the off-target effect that occurs when arbitrary target DNA sequences or similar DNA sequences are present in the genome, which causes unintended DNA cleavage. Thus, the target DNA sequence within the gene of interest should not be similar to other part of the genome sequence to reduce off-target effects.

CRISPRdirect [34] is a web service that assists users in designing the gRNA sequence by searching the PAM with 20-mer DNA sequences in a user-defined sequence (e.g., a knockout target gene). For the input data of CRISPRdirect, users need to prepare the DNA sequence or specify its accession number (RefseqID) or genomic coordinate, which is targeted by CRISPR-Cas9. In addition, users also specify the PAM sequence to adapt to the type of Cas9 used in their experiment. To check the specificity of the target DNA sequences in the reference genome, the genome sequence needs to be specified. Currently, CRISPRdirect supports more than 600 reference genomes, including several assemblies/versions. Figure 6 shows an example of the search result of CRISPRdirect against human genome (hg38) using NGG PAM sequence and NEAT1 genes (NR_028272.1) as input data. The result list includes the position of candidate target sequences (20-mer) with PAM and the number of target sites detected in reference genome sequence. The numbers of 12-mer and 8-mer subsequences of the candidate target sequences are also shown for the estimation of the off-target effect. The “show highly specific target only” checkbox would be useful to find a target sequence with small off-target effects. The graphical view provides visualization of the positions and strands of candidate target sites.

A

B

Results:

Sequence name: NR_131012.1 Homo sapiens nuclear paraspeckle assembly transcript 1 (NEAT1), transcript variant MENbeta, long non-coding RNA
PAM sequence: NGG
Specificity check: Human (Homo sapiens) genome, GRCh38/hg38 (Dec. 2013)
Time: 2021-02-27 15:07:36

- Highlighted target positions (e.g., 45 - 67) indicate sequences that are highly specific and have fewer off-target hits.
- Target sequences with '0' in '20mer+PAM' (in number of target sites column) are shown in gray. Such sequences may possibly span over exon-exon junctions, so avoid using these.
- Target sequences with TTTTs are also shown in gray. Avoid TTTTs in gRNA vectors with pol III promoter.

show **highly specific** target only

Show **20** entries Search:

position	target sequence	sequence information				number of target sites		
start - end	20mer+PAM (total 23mer)	GC% of 20mer	Tm of 20mer	TTTT in 20mer	restriction sites	20mer +PAM	12mer +PAM	8mer +PAM
313 - 335	GGTGGGCTGGTGGTGGTGGG [gRNA]	65.00 %	79.64 °C	-	-	1 [detail]	1 [detail]	1148 [detail]
499 - 521	CGAGGGGGTGGGAGGGGG [gRNA]	70.00 %	82.58 °C	-	-	1 [detail]	1 [detail]	787 [detail]
541 - 563	GGACTGGACTGGACTGGG [gRNA]	50.00 %	70.66 °C	-	-	1 [detail]	1 [detail]	346 [detail]
542 - 564	GGACTGGACTGGACTGGG [gRNA]	50.00 %	69.70 °C	-	-	1 [detail]	1 [detail]	172 [detail]

Showing 1 to 4 of 4 entries (filtered from 247 total entries) First Previous 1 Next Last

Graphical View:

Fig. 6 Search of target DNA sequences for the CRISPR-Cas9 system using CRISPRdirect. (a) Sequence/ID input form for CRISPRdirect search. (b) Search results for CRISPRdirect

2.6 LncRRISearch

LncRRISearch is a web service/database that searches/stores *predicted* long non-coding RNA (lncRNA)-RNA interactions in humans and mice, where the interactions are based on *locally stable* interactions predicted by RIBlast [35], an ultrafast RRI prediction tool that enables us to derive comprehensive local RRIs from the entire Human/Mouse transcriptome [36]. Although RIBlast predicts RRIs based on interaction energy that is computed using internal and external base pairs and is expected to generate unbiased predictions, a relatively large number of interactions are provided. To narrow down the candidate RRIs, LncRRISearch incorporates two kinds of experimental information: expression and subcellular localization. Note that the server accepts only lncRNA as query RNA and accepts both mRNA and lncRNA as target RNAs in the middle of the search.

In LncRRISearch, users can search for local RRIs using one of the following three modes:

Search by name or ID.

Users can search RRIs by gene/transcript name or ID, where both the query and target RNAs are provided by the users. In this mode, the parameters that users can specify are Species (“Human” and “Mouse”) and the Energy thresholds (-16 and -12 kcal/mol) of the interaction energy.

For example, if users select “Human” as Species and “ -16 , kcal/mol” as Energy threshold, and set “NEAT1” as Query, the server returns RRIs between NEAT1 and other transcripts that have an interaction energy smaller than -16 kcal/mol. Note that users can also select a transcript variant as query RNA.

Search by expression pattern.

Expression patterns of query/target transcripts are useful indicators for narrowing down RRIs. In this mode, users can search for RRIs using expression patterns for query and target RNAs. The parameters users can specify in this mode are “Species” (Human or Mouse), “Energy threshold,” “RNA-seq study” (five studies), “Tissue,” and “Expression pattern”; “Expression pattern” includes three patterns, “lncRNA UP Target UP,”

“lncRNA UP Target DOWN,” and “lncRNA DOWN Target UP,” where UP and DOWN mean specifically up- and downregulated in the selected tissue, respectively.

For example, if users select “brain” as Tissue and “lncRNA UP, Target UP” as expression pattern, LncRRIsearch returns a list of lncRNA-RNA interactions, where the lncRNA is specifically upregulated in brain and its target RNA is specifically upregulated.

Search by lncATLAS RCI.

Subcellular localization is another indicator for narrowing down RRI, because only RNAs in the same location have a chance to interact with each other [15]. In this mode, users can search RRIs while considering the localization of the query and target transcripts.

The parameters that users can specify are “Energy threshold,” “Cell line” (among 15 cell lines), “RCI” (Relative Concentration Index [37]) type (cytosol/nucleus or nucleus/cytosol), and “RCI threshold.”

For example, if a user selects “CN-RCI” as RCI, “A549” as Cell line, and “2” as RCI threshold, the server returns a list of RRIs where CN-RCI of both target and query transcripts is greater than 2 in the A549 cell line. Here, the CN-RCI is defined by $CN-RCI = \log_2(FPKM_{cyto}/FPKM_{nuc})$ where FPKM_{cyto} and FPKM_{nuc} indicate FPKM of cytosol and nucleus, respectively. The larger CN-RCI values indicate that the transcripts tend to localize to the cytoplasm.

After searching in any of the three modes described above, the user obtains the following results.

- Genomic location of query and target RNAs/transcripts with the link to UCSC genome browser [38].
- Expression pattern and subcellular localization of query/target sequences among several tissues (Fig. 7a, b).

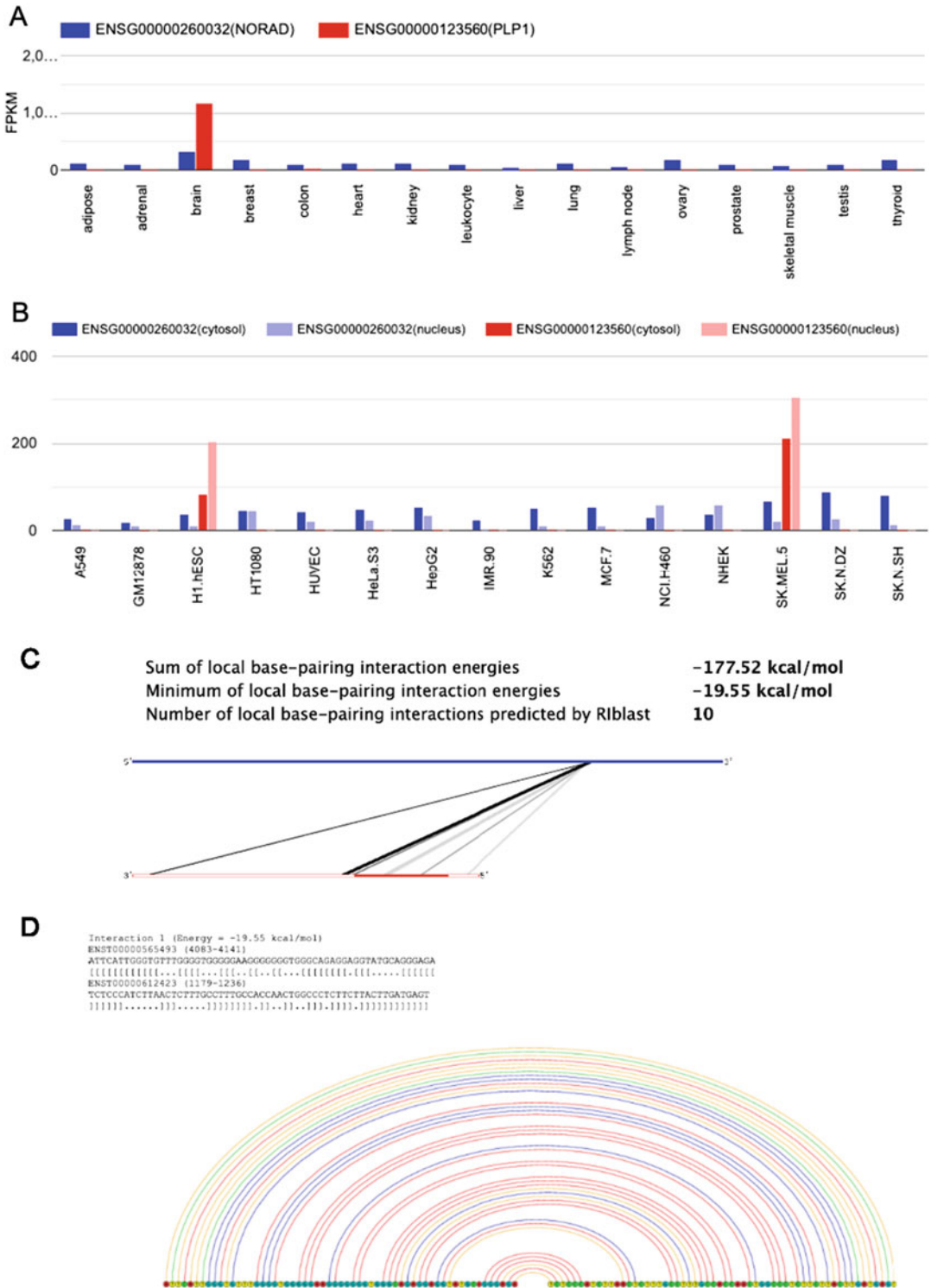


Fig. 7 An example of results by LncRRIssearch: RRI between ENST00000565493 (NORAD, chr20(-) 36045622–36050960) and ENST00000612423 (PLP1, chrX(+)) 103776511–103792616 (a) Expression pattern. (b) Subcellular localization (cytoplasm and nucleus). (c) Global view of RRI, including many local RRIs. (d) Example of a local RRI

- Visualization of the set of local interactions in the whole query and target sequences, the sum/minimum and the number of local interaction energies [16] (Fig. 7c)
- Detailed base pairs of local interactions between query and target sequences (predicted by RIblast) (Fig. 7d).

3 Conclusion

In this review, we introduced various web services for RRI prediction. For bioinformatics software to be widely used, it must have both good performance and high usability. Web service development is an effective way to increase the number of users; thus, webserver papers are cited more often than method development papers in some cases. In the future, along with the development of novel algorithms, the development of user-friendly web services will continue.

Although many web services for CRISPR gRNA design have been developed in recent years, few web services for miRNA target prediction, which were very popular a decade ago, have been developed. Additionally, web services for interaction prediction for specific ncRNA classes, such as snoRNAs and piRNAs, have not been well developed. This underdevelopment probably does not result from existing methods being so accurate that there is no room for improvement but rather that the corresponding research fields are not receiving sufficient attention. Recently, progress has been made in developing high-accuracy machine learning techniques such as deep learning and experimental methods for large-scale RRI detection based on high-throughput sequencers. The development of novel web services for predicting RRIs for specific ncRNA classes using these technologies is an essential research direction.

One of the most difficult problems in web service development is the sustainability of services. A recent study revealed that only about half of the web services developed ten years ago are still available today [62]. When we surveyed web servers for this review, we found that many RRI prediction web services are no longer unavailable. This unsustainability causes not only inconvenience for users performing data analysis but also non-reproducibility of experiments. Web service developers should ensure the availability of web services even after publishing the paper. Additionally, the research community should continue to ensure that the number of unavailable tools does not increase, for example, by providing web service repository maintenance.

References

1. Hsu PW, Huang HD, Hsu SD, Lin LZ, Tsou AP, Tseng CP, Stadler PF, Washietl S, Hofacker IL (2006) miRNAMap: genomic maps of microRNA genes and their target genes in mammalian genomes. *Nucl Acids Res* 34 (Database issue):D135–139
2. Kretz M, Siprashvili Z, Chu C, Webster DE, Zehnder A, Qu K, Lee CS, Flockhart RJ, Groff AF, Chow J, Johnston D, Kim GE, Spitale RC, Flynn RA, Zheng GX, Aiyer S, Raj A, Rinn JL, Chang HY, and Khavari PA (2013) Control of somatic tissue differentiation by the long non-coding RNA TINCR. *Nature* 493(7431):231–235
3. Shi H, Sun Y, He M, Yang X, Hamada M, Fukunaga T, Zhang X, and Chang C (2020) Targeting the TR4 nuclear receptor-mediated lncTASR/AXL signaling with tretinoin increases the sunitinib sensitivity to better suppress the RCC progression. *Oncogene*, 39(3): 530–545
4. Gong C, Maquat LE (2011) lncRNAs transactivate STAU1-mediated mRNA decay by duplexing with 3' UTRs via Alu elements. *Nature* 470(7333):284–288
5. Langdon EM, Qiu Y, Ghanbari Niaki A, McLaughlin GA, Weidmann CA, Gerbich TM, Smith JA, Crutchley JM, Termini CM, Weeks KM, Myong S, Gladfelter AS (2018) mRNA structure determines specificity of a polyQ-driven phase separation. *Science* 360 (6391):922–927
6. Jacq A, Becquet D, Guillen S, Boyer B, Bello-Goutierrez MM, Franc JL, François-Bellan AM (2021) Direct RNA-RNA interaction between Neat1 and RNA targets, as a mechanism for RNAs paraspeckle retention. *RNA Biol* 18(11):2016–2027
7. Lee B, Sahoo A, Marchica J, Holzhauser E, Chen X, Li JL, Seki T, Govindarajan SS, Markey FB, Batish M, Lokhande SJ, Zhang S, Ray A, Perera RJ (2017) The long noncoding RNA SPRIGHTLY acts as an intranuclear organizing hub for pre-mRNA molecules. *Sci Adv* 3(5):e1602505
8. Matheny T, Van Treeck B, Huynh TN, Parker R (2021) RNA partitioning into stress granules is based on the summation of multiple interactions. *RNA* 27(2):174–189
9. Siomi MC, Sato K, Pezic D, Aravin AA (2011) PIWI-interacting small RNAs: the vanguard of genome defence. *Nat Rev Mol Cell Biol* 12(4): 246–258
10. Cox DBT, Gootenberg JS, Abudayyeh OO, Franklin B, Kellner MJ, Joung J, Zhang F (2017) RNA editing with CRISPR-Cas13. *Science* 358(6366):1019–1027
11. Abudayyeh OO, Gootenberg JS, Essletzbichler P, Han S, Joung J, Belanto JJ, Verdine V, Cox D. BT, Kellner MJ, Regev A, Lander ES, Voytas DF, Ting AY, Zhang F (2017) RNA targeting with CRISPR-Cas13. *Nature* 550(7675):280–284
12. Sharma E, Sterne-Weiler T, O'Hanlon D, Blencowe BJ (2016) Global mapping of human RNA-RNA interactions. *Mol Cell* 62(4): 618–626
13. Lu Z, Zhang QC, Lee B, Flynn RA, Smith MA, Robinson JT, Davidovich C, Gooding AR, Goodrich KJ, Mattick JS, Mesirov JP, Cech TR, Chang HY (2016) RNA duplex map in living cells reveals higher-order transcriptome structure. *Cell* 165(5):1267–1279
14. Nguyen TC, Cao X, Yu P, Xiao S, Lu J, Biase FH, Sridhar B, Huang N, Zhang K, Zhong S (2016) Mapping RNA-RNA interactome and RNA structure in vivo by MARIO. *Nat Commun* 7:12023
15. Iwakiri J, Terai G, Hamada M (2017) Computational prediction of lncRNA-mRNA interactions by integrating tissue specificity in human transcriptome. *Biol Direct* 12(1):15
16. Terai G, Iwakiri J, Kameda T, Hamada M, Asai K (2016) Comprehensive prediction of lncRNA-RNA interactions in human transcriptome. *BMC Genomics* 17(Suppl 1):12
17. Umu SU, Gardner PP (2017) A comprehensive benchmark of RNA-RNA interaction prediction tools for all domains of life. *Bioinformatics* 33(7):988–996
18. Wright PR, Georg J, Mann M, Sorescu DA, Richter AS, Lott S, Kleinkauf R, Hess WR, Backofen R (2017) CopraRNA and IntaRNA: predicting small RNA targets, networks and interaction domains. *Nucl Acids Res* 42(Web Server issue):W119–W123
19. Ameres SL, Zamore PD (2013) Diversifying microRNA sequence and function. *Nat Rev Mol Cell Biol* 14(8):475–488
20. Raden M, Ali SM, Alkhnabashi OS, Busch A, Costa F, Davis JA, Eggenhofer F, Gelhausen R, Georg J, Heyne S, Hiller M, Kundu K, Kleinkauf R, Lott SC, Mohamed MM, Mattheis A, Miladi M, Richter AS, Will S, Wolff J, Wright PR, Backofen R (2018) Freiburg RNA tools: a central online resource for RNA-focused research and teaching. *Nucleic Acids Res* 46(W1):W25–W29
21. Busch A, Richter AS, Backofen R (2008) IntaRNA: efficient prediction of bacterial

- sRNA targets incorporating target site accessibility and seed regions. *Bioinformatics* 24(24): 2849–2856
22. Mann M, Wright PR, Backofen R (2017) IntaRNA 2.0: enhanced and customizable prediction of RNA-RNA interactions. *Nucleic Acids Res* 45(W1):W435–W439
 23. Wright PR, Richter AS, Papenfort K, Mann M, Vogel J, Hess WR, Backofen R, Georg J (2013) Comparative genomics boosts target prediction for bacterial small RNAs. *Proc Natl Acad Sci USA* 110(37):E3487–E3496
 24. Lai D, Meyer IM (2016) A comprehensive comparison of general RNA-RNA interaction prediction methods. *Nucleic Acids Research* 44(7):e61–e61
 25. Pain A, Ott A, Amine H, Rochat T, Bouloc P, Gautheret D (2015) An assessment of bacterial small RNA target prediction programs. *RNA Biol* 12(5):509–513
 26. Gruber AR, Lorenz R, Bernhart SH, Neuböck R, Hofacker IL (2008) The Vienna RNA websuite. *Nucleic Acids Res* 36(Web Server issue):W70–W74
 27. Mückstein U, Tafer H, Hackermüller J, Bernhart SH, Stadler PF, Hofacker IL (2006) Thermodynamics of RNA-RNA binding. *Bioinformatics* 22(10):1177–1182
 28. Bernhart SH, Tafer H, Mückstein U, Flamm C, Stadler PF, Hofacker IL (2006) Partition function and base pairing probabilities of RNA heterodimers. *Algorithms Mol Biol* 1(1):1–10
 29. McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for rna secondary structure. *Biopolymers: Original Res Biomol* 29(6–7):1105–1119
 30. Agarwal V, Bell GW, Nam JW, Bartel DP (2015) Predicting effective microRNA target sites in mammalian mRNAs. *Elife* 4:e05005
 31. Watkins NJ, Bohnsack MT (2012) The box C/D and H/ACA snoRNPs: key players in the modification, processing and the dynamic folding of ribosomal RNA. *Wiley Interdiscip Rev RNA* 3(3):397–414
 32. Lowe TM, Eddy SR (1999) A computational screen for methylation guide snoRNAs in yeast. *Science* 283(5405):1168–1171
 33. Schattner P, Decatur WA, Davis CA, Ares M, Fournier MJ, Lowe TM (2004) Genome-wide searching for pseudouridylation guide snoRNAs: analysis of the *Saccharomyces cerevisiae* genome. *Nucleic Acids Res* 32(14): 4281–4296
 34. Naito Y, Hino K, Bono H, Ui-Tei K (2015) CRISPRdirect: software for designing CRISPR/Cas guide RNA with reduced off-target sites. *Bioinformatics* 31(7): 1120–1123
 35. Fukunaga T, Hamada M (2017) RIBlast: an ultrafast RNA-RNA interaction prediction system based on a seed-and-extension approach. *Bioinformatics* 33(17):2666–2674
 36. Fukunaga T, Iwakiri J, Ono Y, Hamada M (2019) LncRRISearch: a web server for lncRNA-RNA interaction prediction integrated with tissue-specific expression and subcellular localization data. *Front Genet* 10: 462
 37. Mas-Ponte D, Carlevaro-Fita J, Palumbo E, Hermoso Pulido T, Guigo R, Johnson R (2017) LncATLAS database for subcellular localization of long noncoding RNAs. *RNA* 23(7):1080–1087
 38. Navarro Gonzalez J, Zweig AS, Speir ML, Schmelter D, Rosenbloom KR, Raney BJ, Powell CC, Nassar LR, Maulding ND, Lee CM, Lee BT, Hinrichs AS, Fyfe AC, Fernandes JD, Diekhans M, Clawson H, Casper J, Benet-Pagès A, Barber GP, Haussler D, Kuhn RM, Haussler M, Kent WJ (2021) The UCSC Genome Browser database: 2021 update. *Nucleic Acids Res* 49(D1):D1046–D1057
 39. Seemann S. E, Menzel P, Backofen R, Gorodkin J (2011) The PETfold and PETcofold web servers for intra- and intermolecular structures of multiple RNA sequences. *Nucleic Acids Res* 39(Web Server issue):W107–W111
 40. Kato Y, Sato K, Asai K, Akutsu T (2012) Rtips: fast and accurate tools for RNA 2D structure prediction using integer programming. *Nucleic Acids Res* 40(Web Server issue):29–34
 41. Eggenhofer F, Tafer H, Stadler PF, Hofacker IL (2011) RNApredator: fast accessibility-based prediction of sRNA targets. *Nucleic Acids Res* 39(Web Server issue):W149–W154
 42. Kery MB, Feldman M, Livny J, Tjaden B (2014) TargetRNA2: identifying targets of small regulatory RNAs in bacteria. *Nucleic Acids Res* 42(Web Server issue):W124–W129
 43. Bellaousov S, Reuter JS, Sectin MG, Mathews DH (2013) RNAstructure: web servers for RNA secondary structure prediction and analysis. *Nucleic Acids Res* 41(Web Server issue): W471–W474
 44. Xu X, Chen SJ (2016) VfoldCPX Server: predicting RNA-RNA complex structure and stability. *PLoS One* 11(9):e0163454
 45. Davis JA, Saunders SJ, Mann M, Backofen R (2017) Combinatorial ensemble miRNA target prediction of co-regulation networks with non-prediction data. *Nucleic Acids Res* 45(15):8745–8757

46. Paraskevopoulou MD, Georgakilas G, Kostoulas N, Vlachos IS, Vergoulis T, Reczko M, Filippidis C, Dalamagas T, Hatzigeorgiou AG (2013) DIANA-microT web server v5.0: service integration into miRNA functional analysis workflows. *Nucleic Acids Res* 41(Web Server issue):W169–W173
47. Miranda KC, Huynh T, Tay Y, Ang YS, Tam WL, Thomson AM, Lim B, Rigoutsos I (2006) A pattern-based method for the identification of MicroRNA binding sites and their corresponding heteroduplexes. *Cell* 126(6):1203–1217
48. Chorostecki U, Palatnik JF (2014) comTAR: a web tool for the prediction and characterization of conserved microRNA targets in plants. *Bioinformatics* 30(14):2066–2067
49. Friedman Y, Karsenty S, Linial M (2014) miRror-Suite: decoding coordinated regulation by microRNAs. *Database* (Oxford) 2014
50. Šulc M, Marín RM, Robins HS, Vaníček J (2015) PACCMIT/PACCMIT-CDS: identifying microRNA targets in 3' UTRs and coding sequences. *Nucleic Acids Res* 43(W1):W474–W479
51. Wu C, Bardes EE, Jegga AG, Aronow BJ (2014) ToppMiR: ranking microRNAs and their mRNA targets based on biological functions and context. *Nucleic Acids Res* 42(Web Server issue):W107–W113
52. Concordet JP, Haeussler M (2018) CRISPOR: intuitive guide selection for CRISPR/Cas9 genome editing experiments and screens. *Nucleic Acids Res* 46(W1):W242–W245
53. Labun K, Montague TG, Krause M, Torres Cleuren YN, Tjeldnes H, Valen E (2019) CHOPCHOP v3: expanding the CRISPR web toolbox beyond genome editing. *Nucleic Acids Res* 47(W1):W171–W174
54. Stemmer M, Thumberger T, Del Sol Keyer M, Wittbrodt J, Mateo JL (2015) CCTop: An Intuitive, Flexible and Reliable CRISPR/Cas9 Target Prediction Tool. *PLoS One* 10(4):e0124633
55. Naito Y, Yoshimura J, Morishita S, Ui-Tei K (2009) siDirect 2.0: updated software for designing functional siRNA with reduced seed-dependent off-target effect. *BMC Bioinformatics* 10:392
56. Lu ZJ, Mathews DH (2008) OligoWalk: an online siRNA design tool utilizing hybridization thermodynamics. *Nucleic Acids Res* 36 (Web Server issue):W104–108
57. Untergasser A, Cutcutache I, Koressaar T, Ye J, Faircloth BC, Remm M, Rozen SG (2012) Primer3–new capabilities and interfaces. *Nucleic Acids Res* 40(15):e115
58. Taneda A, Sato K (2021) A web server for designing molecular switches composed of two interacting RNAs. *Int J Mol Sci* 22(5):2720
59. Menzel P, Seemann SE, Gorodkin J (2012) RILogo: visualizing RNA-RNA interactions. *Bioinformatics* 28(19):2523–2526
60. Dai X, Zhuang Z, Zhao PX (2018) psRNATarget: a plant small RNA target analysis server (2017 release). *Nucleic Acids Res* 46(W1):W49–W54
61. Wu WS, Huang WC, Brown JS, Zhang D, Song X, Chen H, Tu S, Weng Z, Lee HC (2018) pirScan: a webservice to predict piRNA targeting sites and to avoid transgene silencing in *C. elegans*. *Nucleic Acids Res* 46(W1):W43–W48
62. Kern F, Fehlmann T, Keller A (2020) On the lifetime of bioinformatics web services. *Nucleic Acids Res* 48(22):12523–12533



ResidualBind: Uncovering Sequence-Structure Preferences of RNA-Binding Proteins with Deep Neural Networks

Peter K. Koo, Matt Ploenzke, Praveen Anand, Steffan Paul,
and Antonio Majdandzic

Abstract

Deep neural networks have demonstrated improved performance at predicting sequence specificities of DNA- and RNA-binding proteins. However, it remains unclear why they perform better than previous methods that rely on k -mers and position weight matrices. Here, we highlight a recent deep learning-based software package, called ResidualBind, that analyzes RNA-protein interactions using only RNA sequence as an input feature and performs global importance analysis for model interpretability. We discuss practical considerations for model interpretability to uncover learned sequence motifs and their secondary structure preferences.

Key words Deep learning, RNA-binding proteins, Motif analysis, Model interpretability, RNACompete, RNA-protein interactions, Global importance analysis

1 Introduction

Deep neural networks (DNNs) are a powerful class of models that can learn a functional mapping between input genomic sequences and experimentally measured labels, requiring minimal feature engineering [1, 2]. Recently, DNNs have been employed to model RNA-protein interactions [3–10]. Consisting of 244 in vitro affinity selection experiments that span across many RNA-binding protein (RBP) families, the 2013-RNACompete dataset serves as a standard benchmark dataset to compare computational methods [11]. DeepBind was the first deep learning approach to analyze RBP-RNA interactions [3]. At the time, it demonstrated improved performance over position weight matrix (PWM)- and k -mer-based methods [12–15]. Since then, other deep learning-based methods have emerged [4–6], further improving prediction performance on this dataset and other CLIP-seq-based datasets [8–10, 15]. To validate that DNNs have learned biologically meaningful features, previous methods have visualized

representations from first convolutional layer filters and attribution methods [3–5, 8]. First layer filters have been shown to capture motif-like representations, but their efficacy depends highly on model architecture [16, 17]. First-order attribution methods, including in silico mutagenesis [3, 18] and other gradient-based methods [19–22], are interpretability methods that identify the independent importance of single nucleotide variants in a given sequence toward model predictions. The utility of these interpretability methods has mainly been limited to showing that DNNs learn motif representations that resemble *known* motifs, previously identified by PWM- and *k*-mer-based methods, which raises the question, “Why are DNNs performing better?”

Here we highlight ResidualBind, a convolutional neural network with a dilated residual block that has demonstrated improved performance compared to previous methods [7]. Unlike previous DNNs designed for this task, ResidualBind employs a residual block consisting of dilated convolutions, which allows it to fit the residual variance not captured by previous layers while considering a larger sequence context [23]. Moreover, the skipped connection in residual blocks fosters gradient flow to lower layers, improving training of deeper networks [24]. To understand why ResidualBind performs better than previous methods, we employ *global importance analysis* (GIA). Unlike previous attribution methods, which only identify the importance of individual variants within features *local* to an individual sequence, GIA quantifies the *global* importance of features across a population of sequences. Using GIA, we show that in addition to sequence motifs, ResidualBind learns a model that includes the number of motifs, their spacing, and sequence context, such as positive and negative effects of RNA structure context and GC-bias.

2 Methods

2.1 ResidualBind Package Overview

The ResidualBind is a Python package that uses TensorFlow for DNN training and evaluation and is hosted on GitHub: <http://github.com/koo-lab/residualbind>.

2.1.1 Dependencies

- Python 3.5 or greater.
- Pandas, NumPy, SciPy, Matplotlib, H5py.
- TensorFlow 1.15 or greater.
- Logomaker (Tareen and Kinney)

2.1.2 Source Files

- residualbind.py – contains class for ResidualBind and GlobalImportance,
- helper.py – functions for file handling, loading data, and data preprocessing,

2.1.3 Example Files

- `explain.py` – functions for in silico mutagenesis and k -mer alignment-based motif visualization,
- `E_RNAplfold`, `H_RNAplfold`, `I_RNAplfold`, `M_RNAplfold` - `RNAplfold` – script to calculate probability of external loop, hairpin loop, internal loop, and multi-loop, respectively.
- `generate_rnacompete_2013_dataset.py` – scripts to process the RNAcompete dataset,
- `train_rnacompete_2013.py` – trains ResidualBind models on all RNAcompete experiments,
- `test_rnacompete_2013.py` – tests ResidualBind models on all RNAcompete experiments,
- `global_importance_analysis.py` – runs GIA experiments systematically across all RNAcompete,
- `Figure1_performance_analysis.ipynb` – Jupyter notebook that generates Fig. 1 in [7].
- `Figure2_RBFOX1_analysis.ipynb` – Jupyter notebook that generates Fig. 2 in [7].
- `Figure3_VTS1_analysis.ipynb` – Jupyter notebook that generates Fig. 3 in [7].
- `Figure4_GC-bias_analysis.ipynb` – Jupyter notebook that generates Fig. 4 in [7].

2.2 Data

The 2013-RNAcompete dataset can be obtained from [11]. The 2013-RNAcompete experiments consist of ~241,000 RNA sequences each 38–41 nucleotides in length, split into two sets “set A” (120,326 sequences) and “set B” (121,031 sequences). Each sequence from “set A” and “set B” was converted to a one-hot representation. We then performed either clip-transformation or log-transformation. Clip-transformation is performed by clipping the extreme binding scores to the 99.9th percentile. Log-transformation processes the binding scores according to the function: $\log(S - S^{MIN} + 1)$, where S is the raw binding score and S^{MIN} is the minimum value across all raw binding scores. This monotonically reduces extreme binding scores while maintaining their rank order and also yields a distribution that is closer to a normal distribution. The processed binding scores of either clip-transformation or log-transformation were converted to a z-score. We randomly split set A sequences to fractions 0.9 and 0.1 for the training and validation set, respectively. Set B data was held out and used for testing. RNA sequences were converted to a one-hot representation with zero-padding added as needed to ensure all sequences had the same length of 41 nucleotides.

The script to generate the 2013-RNAcompete dataset is `generate_rnacompete_2013_dataset.py`. The output is a file `rnacompete2013.h5`, which contains datasets: `X_train`, `Y_train`, `X_valid`,

Y_valid, X_test, Y_test, and experiment (id number of each RNA-compete experiment). To load and preprocess the data, run the command:

```
train, valid, test = helper.load_rnacompete_data(data_path, ss_type,
normalization, rbp_index)
```

- data_path – path to rnacompete2013.h5.
- ss_type – input data features: “seq” (sequence only), “pu” (sequence + PU), and “struct” (sequence + PHIME).
- normalization – type of normalization to perform, log_norm, or clip_norm.
- rbp_index – index of the RNAcompete experiment ranging from [0, 243].
- train, valid, and test are dictionaries with keys “inputs” for input data and “targets” for labels.

2.3 Secondary Structure Features

Two types of secondary structure profiles can be incorporated into ResidualBind as additional input features: paired-unpaired (PU) structural profiles calculated using RNAplfold [25] and paired, hairpin-loop, internal-loop, multi-loop, and external-loop (PHIME) calculated using a modified RNAplfold script [13]. By default, the window length (-W parameter) and the maximum spanning base-pair distance (-L parameter) are set to the full length of the sequence (script to generate data: generate_rnacompete_2013_dataset.py).

2.4 ResidualBind

ResidualBind takes RNA sequence (and optional secondary structure profiles) as input and outputs a single binding score prediction for a given RBP. ResidualBind consists of (1) convolutional layer (96 filters, filter size 11), (2) dilated residual module [23], (3) mean-pooling layer (pool size 10), (4) fully connected hidden layer (256 units), and (5) fully connected output layer to a single output. The dilated residual module consists of three convolutional layers with a dilation rate of 1, 2, and 4, each with a filter size of 3. Each convolutional layer employs batch normalization prior to a rectified linear unit (ReLU) activation and dropout probabilities according to layers (1) 0.1, (2) 0.2, (4) 0.5. The pre-activated output of the third convolutional layer is added to the inputs of the dilated residual module, a so-called skipped connection [24], the output of which is then activated with a ReLU. The stride of all convolutions is 1 and set to the pool size for the mean-pooling layer.

For regression tasks like the RNAcompete dataset, the output activation should be set to linear (default setting), and the loss function is given by the mean squared error function. For binary classification tasks, such as CLIP-seq datasets, the output activation should be set to sigmoid, and the loss function is given by the binary cross-entropy function. For classification-based methods, users should set the classification flag to True. The ResidualBind class is contained in residualbind.py. To instantiate a ResidualBind model, run command:

```
from residualbind import ResidualBind

resnet = ResidualBind(input_shape, output_shape, num_class,
weights_path, classification=False)
```

- *input_shape* – shape of the input data (length of sequence, # channels – 4 for just RNA, 6 for RNA + PU, and 9 for RNA + PHIME).
- *output_shape* – shape of the output predictions (should be the same dimensions as labels). For a single-task problem, the output shape is 1.
- *num_class* – number of class labels, i.e., dimensionality of output predictions.
- *weights_path* – path to store the model weights – it should have a .hdf5 extension (e.g., /path/to/results/model_weights.hdf5).
- *classification* – specifies whether to perform a binary classification. By default, it is set to False, which means it is set for a regression task (for RNAcompete data).

2.5 Training

ResidualBind is trained by minimizing the loss function with mini-batch stochastic gradient descent (mini-batch of 100 sequences) with Adam updates [26] with a decaying learning rate – the initial learning rate is set to 0.001 and will decay by a factor of 0.3 if the model performance on a validation set (as measured by the Pearson correlation) does not improve for seven epochs. Training stops when the model performance on the validation set does not improve for 25 epochs. Optimal parameters are selected by the epoch which yields the highest performance metric on the validation set – Pearson correlation or area under the receiver operating characteristic curve for regression and classification tasks, respectively. The parameters of each model are initialized according to Glorot initialization [27].

To train a ResidualBind model, run the command:

```
resnet.fit(train, valid, num_epochs=300, batch_size=100, patience=20,  
lr=0.001, lr_decay=0.3, decay_patience=7)
```

- *train* – dictionary with keys “inputs” for input data and “targets” for labels.
- *valid* – dictionary with same keys as train.
- *num_epochs* – maximum number of epochs to train – early stopping may truncate this time.
- *batch_size* – number of sequences to include for gradient updates.
- *patience* – number of epochs to wait for validation metric to improve before stopping training.
- *lr* – initial learning rate.
- *decay_patience* – number of epochs to wait for validation metric to improve before decaying the learning rate by a factor *lr_decay*.
- *lr_decay* – factor to decay the learning rate when *decay_patience* is not satisfied.

To evaluate the model after training, run the command:

```
metrics = resnet.test_model(test, batch_size=100, load_weights=False)
```

- *test* – dictionary with keys “inputs” for input data and “targets” for labels.
- *batch_size* – number of sequences to load onto GPU for each batch.
- *load_weights* – a Boolean that when set to True loads the saved weights from training. This option is important when employing a previously trained ResidualBind model.
- *metrics* – performance on the test set. For regression, metrics represent the Pearson correlation. For classification, metrics give [accuracy, AUROC, AUPR], where AUROC is the area under the receiver operating characteristic curve and AUPR is the area under the precision recall curve.

3 Model Interpretability with Global Importance Analysis

3.1 *In Silico* Mutagenesis

In silico mutagenesis employs a trained model to predict the effects of all possible single nucleotide mutations in a given sequence. The predictions can be ordered as a nucleotide-resolution map ($4 \times L$, where 4 is for each nucleotide and L is the length of the sequence). Each prediction is subtracted by the wild-type prediction, effectively giving zeros at positions where the variant matches the wild-type sequence. To visualize an in silico mutagenesis map, the L2-norm across variants for each position can be calculated and plotted as a sequence logo for the wild-type sequence, where heights correspond to the sensitivity of each position via the L2-norm.

To perform in silico mutagenesis, run the command:

```
import explain

attr_scores = explain.mutagenesis(resnet, X, class_index=0, layer=-1)

scores = np.sum(attr_scores**2, axis=2, keepdims=True)*X
```

- *model* – keras model within ResidualBind class, i.e., model.model.
- *X* – one-hot sequences to perform mutagenesis analysis.
- *class_index* – index for the class to investigate; default is 0 but can be changed for multi-class problems.
- *layer* – layer to get neuron activations from (default is -1) the final output layer.
- *attr_scores* – mutagenesis scores (same dimensions as X).
- *scores* – L2-norm of mutagenesis scores in a one-hot-like representation.

Logomaker [28] is used to plot a sequence logo from an in silico mutagenesis map.

```
import logomaker

import pandas as pd

index = 0 # sequence index

N, L, A = X.shape

counts_df = pd.DataFrame(data=0.0, columns=list('ACGU'),
index=list(range(L)))

for a in range(A):
    for l in range(L):
        counts_df.iloc[l,a] = scores[index,l,a]

plt.figure(); logomaker.Logo(counts_df)
```

3.2 Global Importance Analysis

While in silico mutagenesis, which is the gold standard of model interpretability of DNNs in genomics, is a powerful approach to highlight learned representations that resemble known motifs, it cannot inform the effect size of larger patterns, such as motifs or combinations of motifs. Global importance analysis measures the population-level effect size that a putative feature, like a motif, has on model predictions. Given a sequence-function relationship, i.e., $\mathcal{F}: \mathbf{x} \rightarrow \mathbf{y}$, where \mathbf{x} is a sequence of length L ($\mathbf{x} \in \mathbb{R}^L$) and \mathbf{y} represents a corresponding function measurement ($\mathbf{y} \in \mathbb{R}$), the global importance of pattern ϕ embedded starting at position i in sequences under the data distribution \mathcal{D} is given by:

$$\mathcal{J}^{\text{global}} = \mathbb{E}_{\mathbf{x}^{\phi_i} \sim \mathcal{D}}[\mathbf{y}|\mathbf{x}] - \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[\mathbf{y}|\mathbf{x}], \quad (1)$$

where \mathbb{E} is an expectation.

Equation 1 quantifies the global importance of an embedded feature (ϕ) while marginalizing out the contribution from other positions across a population of sequences. Important to this approach is the randomization of other positions, which is necessary to mitigate the influence of background noise and extraneous confounding signals that may exist in a given individual sequence. If the dataset is sufficiently large and randomized, then Eq. 1 can be calculated directly from the data. Alternatively, a trained DNN can be employed as a surrogate model for experimental measurements

by generating synthetic sequences necessary to calculate Eq. 1 and using model predictions as a proxy for experimental measurements. Given a DNN that maps input sequence to output predictions, i.e., $f: \mathbf{x} \rightarrow \mathbf{y}^*$, where \mathbf{y}^* represents model predictions, the estimated global importance of pattern ϕ embedded starting at position i under the synthetic data distribution \mathcal{D}^* is given by:

$$\widehat{\mathcal{J}}^{\text{global}} \approx \frac{1}{N} \sum_n^N f(\mathbf{x}_n^{\phi_i}) - f(\mathbf{x}_n), \quad (2)$$

where $\widehat{\mathcal{J}}^{\text{global}}$ represents an estimate of $\mathcal{J}^{\text{global}}$ and the expectation is approximated with an average of N samples from a synthetic data distribution $\mathcal{D}^* \sim \mathcal{D}$. By embedding pattern ϕ in position i of the n th sample \mathbf{x}_n , i.e., $\mathbf{x}_n^{\phi_i}$, the difference in the summation of Eq. 2 estimates the *local* effect size of the pattern in a given sequence, while the average estimates the *global*, i.e., population-level, effect size.

To demonstrate how to employ GIA, we generate 1000 synthetic RNA sequences, each 41 nucleotides long, by sampling from a profile-based sequence model, which is estimated by the observed nucleotide frequency at each position of the training data. Patterns under investigation were embedded in positions specified in each GIA experiment. A trained ResidualBind model can be queried with these sequences with and without the embedded pattern. We refer to the difference between the predictions with and without the pattern for each sequence as the “local” importance (the value inside the summation of Eq. 2) and the average across the population as the “global” importance. Below, we detail how to employ GIA using ResidualBind and provide additional considerations in Notes.

GIA is a class in `residualbind.py` and can be instantiated with this command:

```
from residualbind import GlobalImportance

alphabet = 'ACGU'

gia = GlobalImportance(resnet, alphabet)
```

- *resnet* – a ResidualBind model.
- *alphabet* – the order of the nucleotides in the dataset; by default, it is “ACGU.”

To generate a profile sequence model, run:

```
profile_model = np.mean(np.squeeze(train['inputs']), axis=0)
profile_model /= np.sum(profile_model, axis=1, keepdims=True)
gia.set_null_model(null_seq_model, num_sim=1000)
```

- *train* – dictionary with keys “inputs” for input data and “targets” for labels,
- *profile_model* – observed nucleotide frequencies at each position (shape: $L \times 4$),

Experiment: *k*-mer Motif Discovery GIA can be used for ab initio motif discovery by embedding all possible *k*-mers in the null sequences. To perform this analysis, run:

```
kmers, kmer_scores = gia.optimal_kmer(kmer_size=6, position=17,
class_index=0)
```

- *kmer_size* – the size of the kmer to embed in the null sequences.
- *position* – the start position to embed the kmers.
- *class_index* – index of the output class – nonzero values are relevant for multitask problems.
- *kmers* – list of all possible kmers that were embedded.
- *kmer_scores* – global importance corresponding to the list of kmers.

Experiment: Mutagenesis of *k*-mer Using the top scoring *k*-mer as a base binding site, we can determine the importance of each nucleotide variant by calculating the global importance for all possible single nucleotide mutations. To perform this analysis, run:

```
motif = kmers[0]
mean_scores = gia.kmer_mutagenesis(motif, position=17, class_index=0)
```

- *motif* – pattern to be investigated, e.g., *k*-mer with the highest global importance score,
- *position* – the start position to embed the kmer along the sequence.

Visualizing Motif Representations A motif representation can be generated from the global in silico mutagenesis analysis in two ways: (1) by plotting the top k -mer with heights scaled by the L2-norm of the GIA-based in silico mutagenesis scores at each position and (2) by creating an alignment of the top k -mers and calculating a weighted average according to their global importance, which provides a position frequency matrix representation that can be used to generate a sequence logo.

1. Top k -mer motif with heights scaled according to the L2-norm of the global importance of nucleotide variants at each position, which is measured by an in silico mutagenesis of the top k -mer, can be computed with the command:

```
kmer_motif = np.sqrt(np.sum(mean_scores**2, axis=1, keepdims=True))
kmer_onehot = np.array([self.alphabet.index(i) for i in pattern])
logo = kmer_onehot * kmer_motif
```

- *mean_scores* – results from k -mer mutagenesis analysis.

2. The alignment-based k -mer motif can be calculated with the command:

```
kmer_motif = explain.kmer_alignment_motif(kmers, kmer_scores,
alphabet)
I = np.log2(4) + np.sum(kmer_motif * np.log2(kmer_motif+1e-
7),axis=1,keepdims=True)
logo = I * kmer_motif
```

Both logos from top k -mer motif and the alignment-based k -mer motif can be visualized as a sequence logo using Logomaker, according to:

```

import logomaker

import pandas as pd

L = len(kmer_motif)

counts_df = pd.DataFrame(data=0.0, columns=list('ACGU'),
index=list(range(L)))

for l in range(L):
    for a in range(len(alphabet)):
        counts_df.iloc[l,a] = kmer_motif[l,a]

logomaker.Logo(counts_df)

```

Experiment: Multiple Sites To test how ResidualBind considers multiple binding sites, a GIA experiment can be performed by progressively embedding the putative motif (or top k -mer) in synthetic sequences at various positions. For instance, Fig. 1a shows a ResidualBind model that learns each additional RBFOX1 motif (UGCAUG, which was identified as the top 6-mer) is additive. To perform this analysis, run:

```

all_scores = gia.multiple_sites(motif, positions=[4, 12, 20],
class_index=0)

```

- positions – *start positions to successively add additional motifs.*

Experiment: Binding Site Spacing To test whether ResidualBind considers the spacing between motifs, a GIA experiment can be performed by varying the spacing between two binding sites. For example, Fig. 1b shows a ResidualBind model trained on an RNA-compete experiment for RBFOX1 learns that binding scores can decrease when two motifs are too close, which manifests biophysically through steric hindrance. To perform this analysis, run:

```

motif = 'UGCAUG'

positions = [[17,17], [16, 20], [15, 21], [13, 22]]

all_scores = []

for position in positions:

    interventions = []

    for pos in position:

        interventions.append((motif, pos))

    all_scores.append(gia.embed_predict_effect(interventions,
class_index))

```

- positions – *start positions to add motifs – each pair of positions are independent GIA experiment.*

Experiment: Positional Bias To test whether ResidualBind learns a positional bias for a given binding site, a GIA experiment can be performed by systematically embedding the pattern in different locations of the synthetic sequences. To perform this analysis, run:

```

positions = [3, 10, 16, 22, 28, 34]

all_scores = gia.positional_bias(motif, positions, class_index=0)

```

- positions – *start positions to place motifs – each position is an independent GIA experiment.*

Experiment: Secondary Structure Preference To test whether ResidualBind learns secondary structure context, a GIA experiment can be performed by embedding a motif in either the loop or stem region of synthetic sequences designed to have a stem-loop structure – enforcing Watson-Crick base pairs. Figure 1c shows a ResidualBind model trained on the 2009-RNAcompete dataset for VTS1, a well-studied RBP with a sterile-alpha motif (SAM) domain that has a high affinity toward RNA hairpins that contain “CNGG” [29, 30], learns that the canonical VTS1 motif embedded in the context of a hairpin loop leads to higher binding scores compared to when it is placed in other secondary structure contexts. To perform this analysis, run:

```
positions = [17, 9]
all_scores = []

# embed patterns in random RNA (as a control)
for position in positions:
    all_scores.append(gia.embed_predict_effect((motif, position),
class_index=0))

# embed patterns in sequences with stem-loop structure
for position in positions:
    pattern = (motif, position)
    one_hot = gia.embed_pattern_hairpin(pattern, stem_left=8,
stem_right=24, stem_size=9)
    all_scores.append(gia.predict_effect(one_hot))
```

- positions – *start positions to place motifs – each position is an independent GIA experiment.*
- stem_left – *start index of the stem on the left side.*
- stem_right – *start index of the stem on right side – complimentary to left side.*
- stem_size – *length of stem – number of base pairs.*

Experiment: GC-bias To test whether ResidualBind learns positional sequence context that may influence predictions, a GIA experiment can be performed by embedding both the motif in one position and placing the sequence context in another position. In the 2013-RNAcompete dataset, we noticed that *in silico* mutagenesis plots for top scoring sequences exhibited importance scores for known motifs along with GC content toward the 3' end [7]. We did not observe any consistent secondary structure preference for the 3' GC-bias using structure predictions given by RNAplfold. Using GIA, we can test the effect size of the GC-bias for sequences

with a top 6-mer motif embedded at the center (Fig. 1d). To perform this analysis, run:

```
motif_position = 17

gc_motif = 'GCGCGC'

gc_positions = [34, 2]

all_scores = gia.gc_bias(motif, motif_position, gc_motif,
gc_positions, class_index=0)
```

- *motif_position* – start positions to place motifs – each position is an independent GIA experiment.
- *gc_motif* – GC-content pattern.
- *gc_position* – position of the *gc_motif* – each position experiment is independent.

4 Notes

1. *Data processing* – we noticed that the preprocessing step employed by previous DNN methods for the RNAcompete dataset, which clips large experimental binding scores to their 99.9th percentile value and normalizing to a z-score, a technique we refer to as clip-transformation, adversely affects the fidelity of ResidualBind’s predictions for higher binding scores, the most biologically relevant regime. Instead, we prefer preprocessing experimental binding scores with a log-transformation similar to a Box-Cox transformation so that its distribution approaches a normal distribution while also maintaining their rank order.
2. *Visualization of filters* – visualizing first layer convolutional filters can sometimes resemble known motifs. Filter representations are sensitive to network design choices [16, 17]; ResidualBind is not designed with the intention of learning interpretable filters. Hence, visualization of the filters may not be informative.
3. In silico mutagenesis is an informative approach to uncover features without any prior knowledge. Since it only informs the effect size of single nucleotide variants on an individual sequence basis, insights have to be gleaned by observing patterns that generalize across multiple sequences. Without ground truth, interpreting in silico mutagenesis plots can be

challenging. Global importance analysis provides a framework to quantitatively test hypotheses of the importance of features and explore their specific functional relationships.

4. In silico mutagenesis analysis is a first-order interpretability approach, and hence it cannot specify which nucleotides are interacting. Second-order in silico mutagenesis analysis, which scores all 16 possible nucleotide pairs for every pair of positions, can reveal detailed base-pairing interactions [31].
5. *Null sequence selection* – the synthetic data distribution must be chosen carefully to minimize a distribution shift between the synthetic sequences and the data distribution. Here we highlight how GIA can be employed with a profile model – average nucleotide frequency at each position – and a random sequence model with a stem-loop structure. A profile model captures position-dependent biases while averaging down position-independent patterns, like motifs. Alternative sequence models include random shuffling and dinucleotide shuffling of the real sequences in the dataset. If there exist high-order dependencies in the observed sequences, such as RNA secondary structure or motif interactions, a distribution shift between the synthetic sequences and the data distribution may arise, which can lead to misleading results. A synthetic sequence model can also sample real sequences directly from the dataset, although this requires careful selection such that unaccounted patterns do not persist systematically, which can act as a potential confounder. Prior knowledge can help design a suitable synthetic sequence model.
6. Ideally, a computational model trained on an in vitro dataset would learn principles that generalize to other datasets, including in vivo datasets. However, models trained on one dataset typically perform worse when tested on other datasets derived from different sequencing technologies/protocols [32], which have different technical biases [33, 34]. While ResidualBind learned that GC-bias helps performance on test sequences derived from the same technology, we do not expect this feature to generalize to other datasets. Nevertheless, GIA highlights a path forward to tease out sequencing biases, which can inform downstream analysis to either remove/debias unwanted features from the dataset.
7. Versions of TensorFlow – the ResidualBind package is compatible with TensorFlow version 1.15 and version 2.0. Different versions of TensorFlow are not necessarily compatible, and hence it may be beneficial to deploy it in its own Python environment.
8. The global importance of features can be estimated experimentally with sequences designed to contain a pattern under

investigation and randomizing the other positions. Calculating this through experimental measurements can be time-consuming and costly due to the large number of sequences required to calculate Eq. 1. Here we demonstrate how a well-trained neural network can be employed as a proxy for these wet lab experiments, generating predictions (instead of experimental measurements) for sequences necessary to calculate Eq. 1. However, *in silico* experiments are not a true replacement for experiments as predictions are based on the model's fit of the data. Treated as a model interpretability tool, GIA enables the quantification of the effect size of patterns that are causally linked to model predictions. Calculating GIA with wet lab experiments also has its limitations due to measurement noise, which can skew biological processes with technical biases.

9. Using model predictions as a proxy for experimental measurements means that GIA quantifies the effect size for a pattern under investigation through the lens of a DNN, and hence results should be taken from the perspective of model interpretability. While Eqs. 1 and 2 describe the global importance of a single pattern, GIA supports embedding more than one pattern. GIA can also be extended to multitask problems by treating each class independently.

Acknowledgments

The authors would like to acknowledge the support from the Developmental Funds from the Cancer Center Support Grant 5P30CA045508 and the Simons Center for Quantitative Biology at Cold Spring Harbor Laboratory. MP was supported by NIH NCI RFA-CA-19-002. We would also like to thank Justin Kinney, David McCandlish, Ammar Tareen, Nick Lee, Eduardo Esteva, and the rest of the Koo Lab for helpful discussions.

References

1. Eraslan G et al (2019) Deep learning: new computational modelling techniques for genomics. *Nat Rev Genet* 20(7):389–403
2. Koo PK, Ploenzke M (2020) Deep learning for inferring transcription factor binding sites. *Curr Opin Syst Biol* 19:16–23. <https://doi.org/10.1016/j.coisb.2020.04.001>
3. Alipanahi B et al (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat Biotechnol* 33(8):831–838
4. Gandhi S et al (2018) Deepbind: a context sensitive deep learning model of RNA-protein binding. *bioRxiv* 345140. <https://doi.org/10.1101/345140>
5. Ben-Bassat I et al (2018) A deep neural network approach for learning intrinsic protein-RNA binding preferences. *Bioinformatics* 34(17):i638–i646
6. Su Y et al (2019) Integrating thermodynamic and sequence contexts improves protein-RNA binding prediction. *PLoS Comput Biol* 15(9): e1007283

7. Koo PK et al (2021) Global importance analysis: an interpretability method to quantify importance of genomic features in deep neural networks. *PLoS Comput Biol* 17(5):e1008925
8. Ghanbari M and Ohler U. Deep neural networks for interpreting RNA-binding protein target preferences. *Genome Res* 30(2): 214–226
9. Pan X, Shen HB (2017) RNA-protein binding motifs mining with a new hybrid deep learning based cross-domain knowledge integration approach. *BMC Bioinformatics* 18(1):136
10. Grønning AGB et al (2020) DeepCLIP: predicting the effect of mutations on protein–RNA binding with deep learning. *Nucleic Acids Res* 48(13):7099–7118
11. Ray D et al (2013) A compendium of RNA-binding motifs for decoding gene regulation. *Nature* 499(7457):172–177
12. Foat BC et al (2006) Statistical mechanical Modeling of genome-wide transcription factor occupancy data by MatrixREDUCE. *Bioinformatics* 22(14):e141–e149
13. Kazan H et al (2010) RNAcontext: a new method for learning the sequence and structure binding preferences of RNA-binding proteins. *PLoS Comput Biol* 6:e1000832
14. Orenstein Y et al (2016) RCK: accurate and efficient inference of sequence- and structure-based protein–RNA binding models from RNAcompete data. *Bioinformatics* 32(12): i351–i359
15. Maticzka D et al (2014) GraphProt: modeling binding preferences of RNA-binding proteins. *Genome Biol* 15(1):R17
16. Koo PK, Eddy SR (2019) Representation learning of genomic sequence motifs with convolutional neural networks. *PLoS Comput Biol* 15(12):e1007560
17. Koo PK, Ploenzke M (2021) Improving representations of genomic sequence motifs in convolutional networks with exponential activations. *Nat Mach Intell* 3:258–266.
18. Kelley DR et al (2016) Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Res* 26(7):990–999
19. Simonyan K et al (2013) Deep inside convolutional networks: visualising image classification models and saliency maps. arXiv [cs.CV]. <http://arxiv.org/abs/1312.6034>.arXiv
20. Sundararajan M et al (2017) Axiomatic attribution for deep networks. arXiv [cs.LG]. <http://arxiv.org/abs/1703.01365>.arXiv
21. Shrikumar A et al (2017) Learning important features through propagating activation differences. arXiv [cs.CV]. <http://arxiv.org/abs/1704.02685>.arXiv
22. Lundberg SM, Lee S (2017) A unified approach to interpreting model predictions. *Adv Neural Inf Proces Syst* 30:4765–4774
23. Yu F et al (2017) Dilated residual networks. *Proc IEEE Conf Comput Vis Pattern Recognit*:472–480
24. He K et al (2016) Deep residual learning for image recognition. *Proc IEEE Conf Comput Vis Pattern Recognit*:770–778
25. Lorenz R et al (2011) ViennaRNA package 2.0. *Algorithms for molecular biology*. *AMB* 6:26
26. Kingma DP, Ba JL (2014) Adam: a method for stochastic optimization. arXiv [cs.LG]. <http://arxiv.org/abs/1412.6980>.arXiv
27. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*:249–256
28. Tareen A, Kinney JB (2020) Logomaker: beautiful sequence logos in python. *Bioinformatics* 36(7):2272–2274
29. Aviv T et al (2006) Sequence specific recognition of RNA hairpins by the SAM domain of Vts1. *Nat Struct Mol Biol* 13(2):168–176
30. Aviv T et al (2006) The NMR and X-Ray structures of the *saccharomyces cerevisiae* Vts1 SAM domain define a surface for the recognition of RNA hairpins. *J Mol Biol* 356(2):274–279
31. Koo PK et al (2018) Inferring sequence-structure preferences of RNA-binding proteins with convolutional residual networks. *bioRxiv* 418459
32. Weirauch MT et al Evaluation of methods for modeling transcription factor sequence specificity. *Nat Biotechnol* 31(2):126–134
33. Wheeler EC et al (2018) Advances and challenges in the detection of transcriptome-wide protein–RNA interactions. *Wiley Interdiscipl Rev RNA* 9(1):e1436
34. Friedersdorf MB, Keene JD (2014) Advancing the functional utility of PAR-CLIP by quantifying background binding to mRNAs and lncRNAs. *Genome Biol* 15(1):R2



RNA Structure Determination by High-Throughput Structural Analysis

Naoki Takizawa

Abstract

RNA functions are closely linked with their structures. Therefore, elucidating the secondary structure of RNAs provides crucial information regarding their function. The chemical modification or RNase-mediated digestion of single-stranded RNA has been utilized to experimentally reveal RNA secondary structures. Owing to advances in high-throughput sequencing technology and chemical analysis, RNA structural analyses that enable structural profiling at the transcriptomic scale in living cells have been developed. Here, we provide an overview of the high-throughput RNA structural (HTS) analyses and describe the computational processing steps of recent HTS analysis pipelines: PROBer, BUMHMM, and reactIDR.

Key words RNA structure, Secondary structure, High-throughput analysis, Genome-wide analysis, Large-scale sequencing

1 Introduction

1.1 Background

RNA molecules play pivotal roles in diverse cellular functions, including, but not limited to, mediating the transfer of information from genes to proteins as well as the regulation of transcriptional and posttranscriptional processes [1]. Most RNAs are structured molecules composed of single-stranded and double-stranded regions. RNA functions are closely linked with their structure [2]. Specific RNA structures recruit corresponding binding proteins, thereby regulating the function(s) of these proteins. RNA molecules such as ribozymes exhibit enzymatic activity. In fact, elucidating RNA structure has become a vital step for understanding RNA functions [3, 4]. Accurate structural models have been obtained using X-ray crystallography and nuclear magnetic resonance. However, these methods require a high amount of processing time and manual labor. It is difficult to examine the massive scale of RNA structures owing to technological limitations. Computational structural predictions from RNA sequences and

sequence alignment data have been widely used for elucidating RNA secondary structures [5]. However, the prediction accuracy needs to be improved, and the time required for such processing is still too high for application in genome-wide analysis.

To evaluate the massive scale of RNA structures in a cost-effective and automated manner, novel high-throughput experimental methods using large-scale sequencing and chemical or enzymatic reagents have been developed [6]. In such experiments, local structural information is collected using structure-sensitive reagents that modify or cleave RNAs with different reactivities depending on the presence of base pairing. Chemical reagents such as dimethyl sulfate (DMS), 2-methylnicotinic acid imidazolide, 2-(azidomethyl)nicotinic acid acyl imidazole, and nucleases such as RNase S1 and RNase V1 are utilized for sequencing methods including DMS-seq [7–9], selective 2'-hydroxyl acylation and profiling experiment (SHAPE)-seq [10, 11], in vivo click SHAPE (icSHAPE) [12], SHAPE and mutational profiling (SHAPE-MaP) [13], and parallel analysis of RNA structure (PARS) [14]. Despite the experimental differences, this principle remains the same for all methods [15]. A common workflow of the high-throughput structural (HTS) analyses of RNAs is as follows: (1) reaction of the structure-sensitive reagent with RNA molecules, (2) reverse transcription (RT), (3) library prep and large-scale sequencing, and (4) mapping and calculation of reactivity. The probing reagents react with nucleotides depending on the local stereochemistry that is affected by its base-pairing state. The degree of modification at each nucleotide is detected by RT. The RT stops at modified nucleotides or introduces a mutation in the specific buffer condition. RT stops and mutations are counted for each nucleotide from high-throughput sequencing reads. To measure the background noise in RT stops or mutations, a control experiment is performed in which RNAs are not treated with the reagent. Sequence readouts from these experiments are analyzed to extract structural parameters for each nucleotide in terms of its reactivity with the probing reagents. Reactivity profiles of all nucleotides are evaluated based on the counts obtained from the experiment and control assays.

Reactivity estimation is a key step in the determination of the massive scale of RNA structures. The estimation methods differ between studies but share a conceptual framework. To calculate raw reactivity, RT stop or mutation counts are adjusted by variations in coverage. One core assumption in calculating raw reactivity is that the modifications introduced by structure-sensitive reagents additively contribute to RT stops or mutations in control assays. Therefore, raw reactivities are often calculated by the differences or ratios of read levels between the experiment and control assays. For allowing comparison between different reactivity profiles, raw reactivity profiles are normalized, and unusually reactive nucleotides that are considered outliers are excluded from the raw reactivity

profiles by methods such as 2–8% normalization [16] and 90% winsorization [8, 17]. For the determination of a more precise RNA structure from HTS analysis, pipelines including probabilistic models such as Spats [18], ProbrRNA [19], and PROBer [20], and those including the information obtained from the replicated HTS data, such as Mod-seeker [21], BUMHMM [22], and reactIDR [23], have been developed. In principle, HTS data contains stochastic noise, thereby emphasizing the need for statistical analysis to identify true signals.

Here, we summarize the characteristic features of recent reactivity estimation pipelines PROBer, BUMHMM, and reactIDR from a biological perspective and describe the procedure for using these pipelines to analyze the massive scale of RNA structure. The reactivity profiles evaluated from the HTS analysis improve with the use of the appropriate pipeline. However, choosing a method of analysis may prove difficult for a non-bioinformatician. This chapter attempts to present the available solutions from the perspective of a non-bioinformatician for determining precise RNA structure via HTS analysis.

1.2 Characteristic Features of PROBer, BUMHMM, and reactIDR

In transcriptome-wide HTS analysis, some reads align ambiguously with multiple transcripts, such as splicing isoforms and paralogous genes. The multi-mapping problem has an adverse effect on the estimation of the probability of chemical modification in RNA molecules. PROBer estimates transcript abundances and modification profiles based on a statistically rigorous approach. This approach achieves the same performance as the conventional approach for structural estimates by utilizing less amount of data. This pipeline is especially useful in transcriptome-wide analysis with numerous multi-mapping reads and is applicable in a wide range of diverse experiments involving toeprinting assays in high-throughput sequencing such as iCLIP that explores RNA-protein interactions [24] and Pseudo-seq that detects RNA pseudouridylation [25]. It is important to note that PROBer cannot be used for “-MaP” (SHAPE-MaP [13], DMS-MaPseq [26], and PAIR-MaP [27]) data because these assays encode chemical marks as mismatches rather than RT drop-off. A caveat of using PROBer is that PROBer does not consider replicate information. Therefore, the reproducibility of structural analyses needs to be evaluated by methods such as Pearson correlation coefficient. PROBer outputs can be used to estimate raw reactivity profiles. Therefore, the elimination of outliers and the normalization of reactivity are performed separately.

As continuous base-pairing is necessary for generating double-stranded RNA structure, the modification states are assumed to not switch randomly from the unreactive state to the reactive state and vice versa. This assumption is implemented in BUMHMM by hidden Markov model (HMM) algorithm. BUMHMM quantifies

the variability of the drop-off rate in control conditions by the computation of null distribution and determines empirical P-values by comparing each treatment-control drop-off rate with the null distribution. BUMHMM calculates modification probabilities from these empirical P-values by applying HMM. Depending on the unique method for calculating probability, BUMHMM outputs the probabilities of modifications of each nucleotide displaying an almost binary output of 0 (not modified) or 1 (modified). Due to this characteristic feature, one of the advantages of BUMHMM is detecting large structured regions such as pseudoknot structures. However, this feature leads to the difficulty in the discussion of structural heterogeneity such as protein binding with RNAs and multiple configurations of RNAs that form different secondary structures *in vivo*.

Most HTS analytical methods do not consider the information of obtained from replicated experiments. To improve the accuracy of structural prediction, reactIDR uses the irreproducible discovery rate (IDR) [28] with the HMM algorithm to discriminate between true signal and noise generated by replicated HTS experiments. IDR is a statistical method used for detecting chromatin immunoprecipitation-seq peak from replicated experiments. reactIDR outputs loop probability at each nucleotide from input data, which comprises repeatedly measured 5'-end read-depths obtained from icSHAPE, PARS, or other HTS toeprinting assay. Another feature of reactIDR is that reactIDR can incorporate supervised learning to estimate the optimal model parameters. Moreover, reactIDR can be used in various HTS comparisons and achieves the highest accuracy to estimate human rRNA structure from icSHAPE and PARS data in comparison with the accuracy of BUMHMM and PROBER.

2 Materials

2.1 Installation

Installations and methods are confirmed with macOS and can be run with some modifications on Linux.

1. PROBER can be built on Linux and MacOS. CMake and zlib are required for installation. Zlib is installed by default on the MacOS, and CMake can be installed on the MacOS by Homebrew (<https://brew.sh/>):

```
$ brew install cmake
```

2. The source code can be obtained from GitHub. For compiling the source code, type the following lines:

```
$ git clone https://github.com/pachterlab/PROBer.git
$ cd PROBer
$ mkdir build
$ cd build
$ cmake ..
$ make
$ make install
```

3. BUMHMM is developed using the R software. Installation of R (<https://www.r-project.org/>) is a prerequisite, and RStudio (<https://rstudio.com/>) is the recommended integrated development environment for running R scripts. To install the BUMHMM package, start R or RStudio and enter:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("BUMHMM")
```

4. The BUMHMM package is dependent on the Bioconductor packages, SummarizedExperiment, Biostrings, and IRanges, and CRAN package, devtools. To install these packages, enter:

```
BiocManager::install(c("SummarizedExperiment", "Biostrings", "IRanges"))
install.packages('devtools')
```

5. reactIDR is developed in Python 3. Python 3 can be installed via Homebrew and pyenv, etc. (*see Note 1*). To install Python 3 by Homebrew, enter:

```
$ brew install python
```

6. Python libraries numpy and scikit-learn are required for reactIDR, and cython is required for installation (*see Note 2*). These packages are installed by pip:

```
$ pip install numpy scikit-learn cython
```

7. The source code of reactIDR can be obtained from GitHub. To install reactIDR, enter:

```
$ git clone https://github.com/carushi/reactIDR
$ cd reactIDR
$ python setup.py build_ext -b reactIDR
```

8. Docker image for RT stop counter that converts bam to read count data is available. Installation of Docker (<https://www.docker.com/>) is a prerequisite for using this converter. After installing and running Docker, the docker image of RT stop counter is downloaded:

```
$ docker pull carushi/rt_end_counter
```

2.2 Input Data

Detailed protocols for chemical modification and sequencing library preparation of genome-wide HTS analysis has been published [29–31]. If the experiment is designed to examine genome-wide RNA structures, sufficient amount and quality of starting RNA and a high number of sequencing reads are required to achieve sufficient coverage of the RNA molecules. For each replicate, approximately 200 to 600 million raw reads were obtained in the genome-wide icSHAPE experiment [12]. Paired-end sequencing is not necessary for determining reverse transcriptase stop positions but is recommended to calculate coverages in PROBer. If possible, PCR duplicates are removed by the barcode sequence in the adapter, and the adapter sequences are trimmed using tools such as Trimmomatic [32]. Sequence reads are aligned by alignment tools such as Bowtie2 [33] (*see Note 3*), and the sam files are converted to bam files by samtools [34]. Samtools can be installed by Homebrew.

The reference fasta file and aligned bam file (or sequence fastq file) are necessary for running PROBer and reactIDR. The dot-bracket format of the predicted RNA secondary structure file is required as an option for running reactIDR. Before running BUMHMM, a reference sequence file and three files described as coverage, drop-off count, and drop-off rate for each nucleotide need to be created (*see Note 4*).

3 Methods

3.1 PROBer

1. Create reference files for PROBer by PROBer prepare:

```
$ PROBer prepare (reference fasta).fa (reference name)
```

2. After creating reference files, perform estimation of probabilities by PROBer estimate:

```
$ PROBer estimate (reference name) (sample name) --alignments
(experiment).bam (control).bam --size-selection-min 100 --
size-selection-max 500
```

--size-selection-min and --size-selection-max set to minimal and maximal size of the fragments. The sequencing fastq files are available with --bowtie2 and --reads options instead of --alignments. In this case, reference files for bowtie2 need to be created by PROBer prepare with --bowtie2 option. If the input reads are paired-end reads, --paired-end option is added to the proper estimate. If the primer length is not 6, it is set by --primer-length <int> option. After running PROBer, three output files, (sample name).expr, (sample name).beta, and (sample name).gamma, are created. The .expr file contains transcript id, length, the amount of the transcript by TPM (transcript per million), and

FPKM (fragment per kilobase per million reads) information. The .beta and .gamma files contain the reactivity profiles and transcriptase noise of each nucleotide, respectively. A normalized reactivity profile can be calculated from .beta file (*see Note 5*).

The figures below demonstrate an example of the normalized probability score processed by PROBER. SHAPE-seq data of influenza virus that has eight-segmented viral RNA (vRNA) as a genome are processed by PROBER, and the probabilities are normalized by 90% winsorization. A pseudoknot structure is predicted in segment 5 vRNA, and the length of segment 5 vRNA is approximately 1500 nt. Therefore, we chose segment 5 vRNA as an example of HTS analysis. Figure 1a shows the probability of segment 5 vRNA, and Fig. 1b shows the probability of nucleotide positions 60–160 of segment 5 vRNA that includes a predicted pseudoknot structure at nucleotide positions 87–130.

3.2 BUMHMM

The coverage, drop-off count, and drop-off rate of each nucleotide are calculated and saved in a tab-separated file.

C1	C2	T1	T2
50258	10914	22580	10548
61362	13558	24949	11371

“C1” and “C2” denote replicate 1 and 2 of control experiments, and “T1” and “T2” denote replicate 1 and 2 of treatment experiments.

1. Before starting BUMHMM, these data are stored in a SummarizedExperiment object in R. Start RStudio or R and enter:

```
library(BUMHMM)
library(Biostrings)
library(SummarizedExperiment)
library(IRanges)

counts <- read.delim("coverage.txt")
counts <- as.matrix(counts)
dropoff <- read.delim("dropoff.txt")
dropoff <- as.matrix(dropoff)
rate <- read.delim("dropoff_rate.txt")
rate <- as.matrix(rate)
colData <- DataFrame(replicate=c("control", "control", "treatment", "treatment"), row.names=c("C1", "C2", "T1", "T2"))
se_exp <- SummarizedExperiment (assays=list(coverage=counts, dropoff_count=dropoff, dropoff_rate=rate), colData=colData)
```

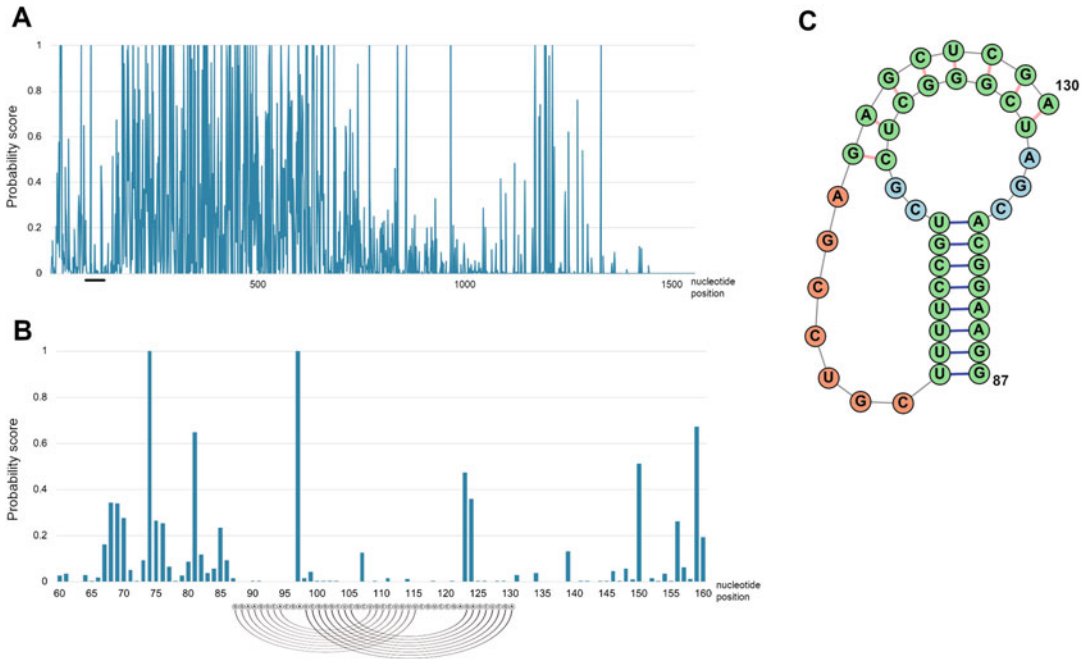



Fig. 1 The reactivity score of influenza virus genome segment calculated by PROBER. Probabilities from the SHAPE-seq of influenza virus virion were calculated by PROBER and normalized by 90% winsorization. (a) The reactivity score of viral genome segment 5 and (b) reactivity score of nucleotide positions 60–160 that contains predicted pseudoknot structure. Black line indicates the predicted pseudoknot region. Sequence and lines indicate predicted base pairs in the pseudoknot structure. (c) Predicted pseudoknot structure at nucleotide positions 87–130 in segment 5. A pseudoknot structure at nucleotide positions 87–130 predicted by IPknot (<http://tips.dna.bio.keio.ac.jp/ipknot/>) is shown

2. Steps involving the selection of nucleotide pairs to compute the log-ratios, scaling the drop-off rates across replicates, and computing stretches of nucleotide positions for HMM analysis are performed. N_c and N_t refer to the numbers of control and treatment experimental replicates, respectively, and t denotes coverage threshold:

```
Nc <- Nt <- 2
t <- 1
nuclSelection <- selectNuclPos(se_exp, Nc, Nt, t)
assay(se_exp, "dropoff_rate") <- scaleDOR(se_exp, nuclSelection, Nc, Nt)
stretches <- computeStretches(se_exp, t)
```

3. The correction of coverage bias and sequence bias is performed:

```
varStab <- stabiliseVariance(se_exp, nuclSelection, Nc, Nt)
LDR_C <- varStab$LDR_C
LDR_CT <- varStab$LDR_CT
```

```

hist(LDR_C, breaks = 30)

nuclNum <- 3
patterns <- nuclPerm(nuclNum)
sequence <- DNASTring(scan("(referece sequence).txt",
what=""))
nuclPosition <- findPatternPos(patterns, sequence, '+')

```

4. In this step, posterior probabilities with HMM are calculated and output files are saved:

```

nuclPosition <- list()
nuclPosition[[1]] <- 1:nchar(sequence)
posteriors <- computeProbs(LDR_C, LDR_CT, Nc, Nt, '+', nucl-
Position, nuclSelection$analysedC, nuclSelection$analysedCT,
stretches)
head(posteriors)
shifted_posteriors <- matrix(, nrow=dim(posteriors)[1],
ncol=1)
shifted_posteriors[1:(length(shifted_posteriors) - 1)] <-
posteriors[2:dim(posteriors)[1],2]
plot(shifted_posteriors)
write.csv(shifted_posteriors,"(output probability file name).
txt", quote=F, row.names = F)

```

The output file contains probabilities ranging from 0 to 1. Figure 2 shows an example of probabilities processed by BUMHMM. The duplicated SHAPE-seq data of the influenza virus were processed, and probabilities of segment 5 vRNA were shown.

3.3 *reactDR*

1. RT stops and the coverage of each nucleotide are counted by `rt_end_counter` of the docker image. In this example, all bam files were stored in `/bam` directory. After running docker, enter:

```

$ docker run --name rtecount -it carushi/rt_end_counter -i
$ mkdir bam
$ exit
$ docker start rtecount
$ docker cp (local path of bam files)/. rtecount:/bam/
$ docker exec -it rtecount /bin/bash
$ bash count_and_cov.sh bam/(filename).bam
(convert all bam files to bed)
$ cd bam
$ mkdir bed
$ mv *.bed bed
$ exit
$ docker cp rtecount:/bam/bed (local path to save bed files)
$ docker stop rtecount
$ docker rm rtecount

```

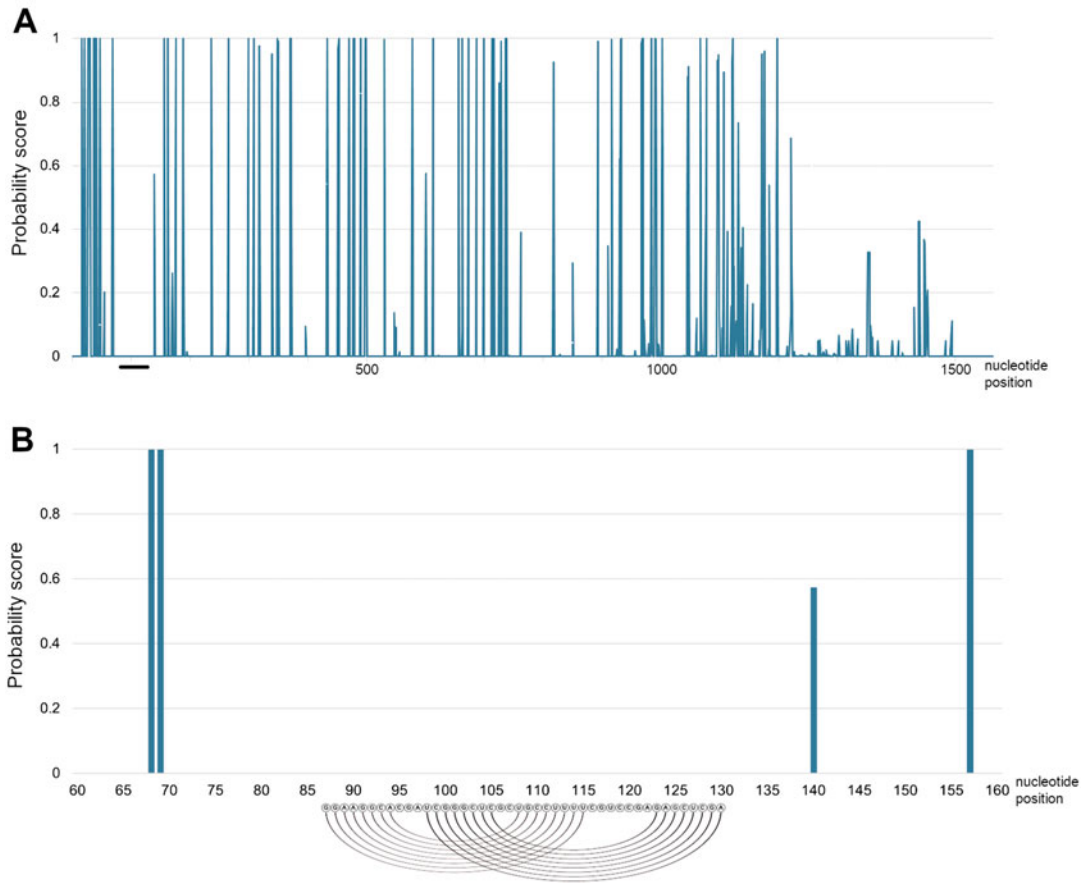


Fig. 2 The reactivity score of influenza virus genome segment calculated by BUMHMM. Probabilities from the SHAPE-seq of influenza virus virion were calculated by BUMHMM. (a) The reactivity score of viral genome segment 5. (b) The reactivity score of nucleotide positions 60–160. Black line indicates the predicted pseudoknot region. Sequence and lines indicate predicted base pairs in the pseudoknot structure

All bam files including the duplicated bam files from the treated sample and from control sample are converted to bed files.

2. The bed files are converted and merged. RT stop count and coverage files from duplicate treated and control samples are converted in this step. In this example, all bed files are moved to the bed directory created in the installed reactIDR directory. Change directory to the reactIDR/bed directory:

```
$ python ../script/bed_to_pars_format.py --offset -1 --fasta
(reference fasta file).fa (filename)_l15q0filt_ctss.bed
(convert all ctss bed files)
```

```
$ python ../script/bed_to_pars_format.py --offset -0 --fasta
(reference fasta file).fa (filename)_l15q0filt_cov.bed
(convert all cov bed files)
$ python ../reactIDR/score_converter.py --merge --output
(name)_ctss (filename treatment rep1)_l15q0filt_ctss.bed.tab,
(filename treatment rep2)_l15q0filt_ctss.bed.tab (filename
control rep1)_l15q0filt_ctss.bed.tab,(filename control rep2)
_l15q0filt_ctss.bed.tab
$ python ../reactIDR/score_converter.py --merge --output
(name)_cov (filename treatment rep1)_l15q0filt_cov.bed.tab,
(filename treatment rep2)_l15q0filt_cov.bed.tab (filename con-
trol rep1)_l15q0filt_cov.bed.tab,(filename control rep2)
_l15q0filt_cov.bed.tab
```

3. This step is optional. In this step, the parameters are evaluated from the predicted secondary structure. The RNA secondary structures can be predicted, and the dot-bracket format of RNA secondary structure file can be obtained using RNA secondary structure prediction programs such as RNA fold and CentroidFold. The dot-bracket format file should contain all of predicted secondary structures in the reference transcript. The dot-bracket format file is stored in the same directory of bed files. The following script is entered to perform this step:

```
$ python ../reactIDR/IDR_hmm.py --train --time 10 --core 4 --
grid --output_param (filename)_train.param.txt --output (fi-
lename).csv --ref (reference predicted secondary structure
file).fa --case (filename)_ctss_case.tab --cont (filename)_-
ctss_cont.tab
```

4. In this step, the probabilities are calculated. If the trained parameter file is not created in one step before, `--param` and `--ref` options are not necessary for calculation.

```
$ python ../reactIDR/IDR_hmm.py --test --time 10 --core 4 --
param (filename)_train.param.txt --output (filename).csv --ref
(reference predicted secondary structure file).fa --case
(filename)_ctss_case.tab --cont (filename)_ctss_cont.tab
```

`test_(filename).csv` is the output file. “IDR-HMM-final (transcript name)+case” items represent posterior probabilities after IDR and HMM. Figure 3 shows an example of probabilities processed by reactIDR. Duplicated SHAPE-seq data of the influenza virus were processed, and probabilities of segment 5 vRNA were shown.

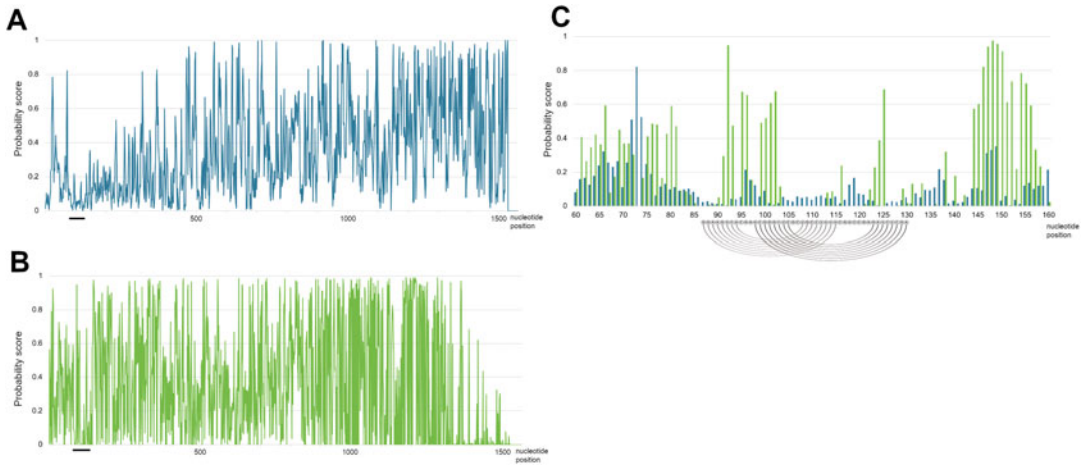


Fig. 3 The reactivity score of influenza virus genome segment calculated by reactIDR. Probabilities from the SHAPE-seq of influenza virus virion were calculated by reactIDR. The figure shows the reactivity score of viral genome segment 5 from `-test` option (**a**), `-test` option without learning parameters (**b**), and nucleotide positions 60–160 (**c**). Blue indicates probabilities from `-test` option, and green indicates those without learning parameters. Black line indicates the predicted pseudoknot region. Sequence and lines indicate predicted base pairs in the pseudoknot structure

5. reactIDR package contains python programs to visualize the output csv files (*see* **Note 6**). To plot csv files of reactIDR, type:

```
$ python ../reactIDR/plot_bargraph.py --window 10 --ignore --
output (output file name) (input csv file 1).csv (input csv
file 2).csv
```

One or two csv files are necessary for plotting. `--window` option means window size of moving average. The script outputs PDF files for each transcript.

4 Notes

1. For non-bioinformatician, python version control is complicated. I simply recommend non-bioinformaticians to use Homebrew for the installation of Python 3. It is important to note that Python 2 is installed by default in macOS, so that the path description in `.bash_profile` needs to change. Path of Python 3 (`/usr/local/opt/python/libexec/bin`) is added to the first of “`export PATH=`” place in `.bash_profile`.
2. If you get an error with `scipy`, check the version of `scipy` which is installed on the system. I did a test run on `scipy` version 1.2.2. To install `scipy` version 1.2.2, enter:

```
$ pip install scipy==1.2.2
```

3. When reads are mapped in the transcriptome-wide analysis, you need to use different reference sequences depending on the software you will use. PROBER and reactIDR can calculate probabilities of each transcript at a run. BUMHMM reads and outputs one sequence at a run. Thus, when BUMHMM is used for transcriptome-wide analysis, the reads are mapped to the chromosome.
4. As mentioned in Methods 3.2, the coverage, drop-off count, and drop-off rate of each nucleotide need to be counted separately for running BUMHMM. The coverage can be counted by using igvtools. The drop-off count can be counted from sam file. The `rt_end_counter` used in reactIDR counts the coverage and drop-off count, and therefore, the output bed files can be used for input in BUMHMM after modifications. The drop-off rate can be calculated from the coverage and drop-off count.
5. R can be used for normalization. When 90% winsorization is used for normalization, the fifth and 95th percentile are calculated in R and the probabilities are normalized based on the values.
6. The Python packages, `seaborn` and `pandas`, are necessary for plotting. These packages can be installed by the following command:

```
$ pip install seaborn pandas
```

References

1. Sharp PA (2009) The centrality of RNA. *Cell* 136:577–580. <https://doi.org/10.1016/j.cell.2009.02.007>
2. Mortimer SA, Kidwell MA, Doudna JA (2014) Insights into RNA structure and function from genome-wide studies. *Nat Rev Genet* 15: 469–479. <https://doi.org/10.1038/nrg3681>
3. Strobel EJ, Watters KE, Loughrey D, Lucks JB (2016) RNA systems biology: uniting functional discoveries and structural tools to understand global roles of RNAs. *Curr Opin Biotechnol* 39:182–191. <https://doi.org/10.1016/j.copbio.2016.03.019>
4. Wan Y, Kertesz M, Spitale RC et al (2011) Understanding the transcriptome through RNA structure. *Nat Rev Genet* 12:641–655. <https://doi.org/10.1038/nrg3049>
5. Mathews DH, Moss WN, Turner DH (2010) Folding and finding RNA secondary structure. *Cold Spring Harb Perspect Biol* 2. <https://doi.org/10.1101/cshperspect.a003665>
6. Spitale RC, Flynn RA, Torre EA et al (2014) RNA structural analysis by evolving SHAPE chemistry. *Wiley Interdiscip Rev RNA* 5: 867–881. <https://doi.org/10.1002/wrna.1253>
7. Ding Y, Tang Y, Kwok CK et al (2014) In vivo genome-wide profiling of RNA secondary structure reveals novel regulatory features. *Nature* 505:696–700. <https://doi.org/10.1038/nature12756>
8. Rouskin S, Zubradt M, Washietl S et al (2014) Genome-wide probing of RNA structure reveals active unfolding of mRNA structures in vivo. *Nature* 505:701–705. <https://doi.org/10.1038/nature12894>
9. Wan Y, Qu K, Zhang QC et al (2014) Landscape and variation of RNA secondary structure across the human transcriptome. *Nature* 505:706–709. <https://doi.org/10.1038/nature12946>

10. Lucks JB, Mortimer SA, Trapnell C et al (2011) Multiplexed RNA structure characterization with selective 2'-hydroxyl acylation analyzed by primer extension sequencing (SHAPE-Seq). *Proc Natl Acad Sci U S A* 108: 11063–11068. <https://doi.org/10.1073/pnas.1106501108>
11. Spitale RC, Crisalli P, Flynn RA et al (2013) RNA SHAPE analysis in living cells. *Nat Chem Biol* 9:18–20. <https://doi.org/10.1038/nchembio.1131>
12. Spitale RC, Flynn RA, Zhang QC et al (2015) Structural imprints in vivo decode RNA regulatory mechanisms. *Nature* 519:486–490. <https://doi.org/10.1038/nature14263>
13. Siegfried NA, Busan S, Rice GM et al (2014) RNA motif discovery by SHAPE and mutational profiling (SHAPE-MaP). *Nat Methods* 11:959–965. <https://doi.org/10.1038/nmeth.3029>
14. Kertesz M, Wan Y, Mazor E et al (2010) Genome-wide measurement of RNA secondary structure in yeast. *Nature* 467:103–107. <https://doi.org/10.1038/nature09322>
15. Choudhary K, Deng F, Aviran S (2017) Comparative and integrative analysis of RNA structural profiling data: current practices and emerging questions. *Quant Biol* 5:3–24. <https://doi.org/10.1007/s40484-017-0093-6>
16. Low JT, Weeks KM (2010) SHAPE-directed RNA secondary structure prediction. *Methods* 52:150–158. <https://doi.org/10.1016/j.ymeth.2010.06.007>
17. Incarnato D, Neri F, Anselmi F, Oliviero S (2014) Genome-wide profiling of mouse RNA secondary structures reveals key features of the mammalian transcriptome. *Genome Biol* 15:491. <https://doi.org/10.1186/s13059-014-0491-2>
18. Aviran S, Trapnell C, Lucks JB et al (2011) Modeling and automation of sequencing-based characterization of RNA structure. *Proc Natl Acad Sci U S A* 108:11069–11074. <https://doi.org/10.1073/pnas.1106541108>
19. Hu X, Wong TKF, Lu ZJ et al (2014) Computational identification of protein binding sites on RNAs using high-throughput RNA structure-probing data. *Bioinformatics* 30: 1049–1055. <https://doi.org/10.1093/bioinformatics/btt757>
20. Li B, Tambe A, Aviran S, Pachter L (2017) PROber provides a general toolkit for Analyzing sequencing-based Toeprinting assays. *Cell Syst* 4:568–574.e7. <https://doi.org/10.1016/j.cels.2017.04.007>
21. Talkish J, May G, Lin Y et al (2014) Mod-seq: high-throughput sequencing for chemical probing of RNA structure. *RNA* 20:713–720. <https://doi.org/10.1261/rna.042218.113>
22. Selega A, Sirocchi C, Iosub I et al (2017) Robust statistical modeling improves sensitivity of high-throughput RNA structure probing experiments. *Nat Methods* 14:83–89. <https://doi.org/10.1038/nmeth.4068>
23. Kawaguchi R, Kiryu H, Iwakiri J, Sese J (2019) reactIDR: evaluation of the statistical reproducibility of high-throughput structural analyses towards a robust RNA structure prediction. *BMC Bioinformatics* 20:130. <https://doi.org/10.1186/s12859-019-2645-4>
24. König J, Zarnack K, Rot G et al (2010) iCLIP reveals the function of hnRNP particles in splicing at individual nucleotide resolution. *Nat Struct Mol Biol* 17:909–915. <https://doi.org/10.1038/nsmb.1838>
25. Carlile TM, Rojas-Duran MF, Zinshteyn B et al (2014) Pseudouridine profiling reveals regulated mRNA pseudouridylation in yeast and human cells. *Nature* 515:143–146. <https://doi.org/10.1038/nature13802>
26. Zubradt M, Gupta P, Persad S et al (2016) DMS-MaPseq for genome-wide or targeted RNA structure probing in vivo. *Nat Methods* 14:75–82. <https://doi.org/10.1038/nmeth.4057>
27. Li C, Hatta M, Nidom CA et al (2010) Reassortment between avian H5N1 and human H3N2 influenza viruses creates hybrid viruses with substantial virulence. *Proc Natl Acad Sci* 107:4687–4692. <https://doi.org/10.1073/pnas.0912807107>
28. Li Q, Brown JB, Huang H, Bickel PJ (2011) Measuring reproducibility of high-throughput experiments. *Ann Appl Stat* 5:1752–1779. <https://doi.org/10.1214/11-AOAS466>
29. Flynn RA, Zhang QC, Spitale RC et al (2016) Transcriptome-wide interrogation of RNA secondary structure in living cells with icSHAPE. *Nat Protoc* 11:273–290. <https://doi.org/10.1038/nprot.2016.011>
30. Smola MJ, Rice GM, Busan S et al (2015) Selective 2'-hydroxyl acylation analyzed by primer extension and mutational profiling (SHAPE-MaP) for direct, versatile and accurate RNA structure analysis. *Nat Protoc* 10: 1643–1669. <https://doi.org/10.1038/nprot.2015.103>
31. Ding Y, Kwok CK, Tang Y et al (2015) Genome-wide profiling of in vivo RNA structure at single-nucleotide resolution using

- structure-seq. *Nat Protoc* 10:1050–1066. <https://doi.org/10.1038/nprot.2015.064>
32. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics* 30:2114–2120. <https://doi.org/10.1093/bioinformatics/btu170>
33. Langmead B, Salzberg SL (2012) Fast gapped-read alignment with bowtie 2. *Nat Methods* 9: 357–359. <https://doi.org/10.1038/nmeth.1923>
34. Li H, Handsaker B, Wysoker A et al (2009) The sequence alignment/map format and SAMtools. *Bioinformatics* 25:2078–2079. <https://doi.org/10.1093/bioinformatics/btp352>



RNA 3D Modeling with FARFAR2, Online

Andrew M. Watkins and Rhiju Das

Abstract

Understanding the three-dimensional structure of an RNA molecule is often essential to understanding its function. Sampling algorithms and energy functions for RNA structure prediction are improving, due to the increasing diversity of structural data available for training statistical potentials and testing structural data, along with a steady supply of blind challenges through the RNA-Puzzles initiative. The recent FARFAR2 algorithm enables near-native structure predictions on fairly complex RNA structures, including automated selection of final candidate models and estimation of model accuracy. Here, we describe the use of a publicly available webserver for RNA modeling for realistic scenarios using FARFAR2, available at <https://rosie.rosettacommons.org/farfar2>. We walk through two cases in some detail: a simple model pseudoknot from the frameshifting element of beet western yellows virus modeled using the “basic interface” to the webserver and a replication of RNA-Puzzle 20, a metagenomic twister sister ribozyme, using the “advanced interface.” We also describe example runs of FARFAR2 modeling including two kinds of experimental data: a c-di-GMP riboswitch modeled with low-resolution restraints from MOHCA-seq experiments and a tandem GA motif modeled with ^1H NMR chemical shifts.

Key words RNA, 3D structure modeling, Rosetta

1 Introduction

Noncoding RNA molecules exhibit diverse cellular functions, from catalysis to the detection of small molecules to translation itself [1], and they execute those functions by adopting intricate three-dimensional folds. In such well-defined structures, an RNA’s secondary structure elements are fixed in defined orientations by junctions and tertiary contacts. To keep pace with the acceleration in sequencing technology furnishing new RNA molecules for study, experimental methods for 3D structure determination are being successfully supplemented with structure prediction methods, from physical modeling to knowledge-based techniques [2–5].

Increasingly, new methods for biomolecular modeling are released as webservers, to ensure scientific reproducibility and to mitigate challenges in installation or the availability of computational resources for scientists. This trend includes the ROSIE

platform [6, 7], which provides a simplified interface for nonexpert users to access computationally intensive protocols developed in the Rosetta framework [8]. The ROSIE server for FARFAR2 enables researchers to model their RNA of interest using a Rosetta algorithm with excellent performance on RNA-Puzzles and other blind challenges [9]. This chapter provides an overview of the two interfaces to this webserver and illustrates how to apply each one to real RNA modeling cases. Because FARFAR2 requires significant computational expense to sample a modeling problem thoroughly, this server, available at <https://rosie.rosettacommons.org/farf2>, provides users with a few thousand CPU-hours for their modeling problem, free of charge.

2 Method

Here, we illustrate how to use the FARFAR2 ROSIE server in detail for two example problems using the server’s two available interfaces. The “basic” interface allows users to provide nothing more than the two most common pieces of data available for RNA structure prediction tasks: the sequence and dot-bracket secondary structure. These data are also the standard inputs provided to RNA 3D modeling webservers from SimRNAweb [10] and RNAComposer [11] to iFoldRNA v2 [12] and MC-FOLD | MC-SYM [13]. No files need to be prepared. The “advanced” interface permits users to specify significantly more options; every parameter that can affect command-line executions of Rosetta’s FARFAR2 algorithm may be specified through this interface. Users may create an account to receive higher priority and email notifications, or they may submit as guests. Whether or not they create an account, users may make their jobs private if they involve sensitive data. All files needed to run these examples are available from the Appendix.

2.1 The “Basic” Interface to the FARFAR2 ROSIE Server

First, we examine the basic interface (Fig. 1a, b) through interrogation of a pseudoknotted –1 frameshifting element from beet western yellows virus (BWYV; PDB code: 1L2X) [14].

2.1.1 Sequence Specification

The user may specify the sequence of the RNA of interest, either as lowercase or uppercase. Chain boundaries ought to be specified via commas. Rosetta’s internal representation of RNA sequence uses lowercase letters, to permit compatibility with uppercase protein sequences in other applications; user specification of capital letters A, C, G, and U will be converted to lowercase. For the BWYV frameshifting element segment crystallized and deposited as 1L2X, the sequence input is:

```
gcgcggcaccguccgcggaacaaacgg
```

A Submit a new FARFAR2 job

Job short description (visible in queue)

Just specify sequence and dot-bracket secondary structure.

Use [almost] any options available to command line FARFAR2, including local template structures, "general" secondary structure (allowing noncanonical pairs), chemical mapping data, and more.

Sequence

RNA sequence as a single line of text:

Secondary structure

Dot-bracket secondary structure of the target. Must be the same length, in total number of residues, as the target fasta. May use []<> for pseudoknots. Use . for any regions specified explicitly as starting PDBs, even if those regions are helical.

Structures

B FARFAR2 Job 112x_test 「№86872」 Details

Inputs	Status
Job ID	86872
Job Name	112x_test
Visibility	PUBLIC
Protocol	FARFAR2
CPU hours used	3787.6
user	Developer
Status	Finished
Daemon	GrayLab.Rosetta-4
Description	

Results

Ten lowest-energy cluster centers created in the FARFAR2 run:

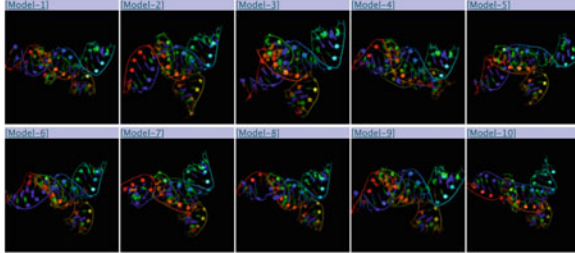
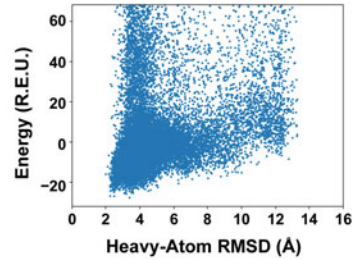
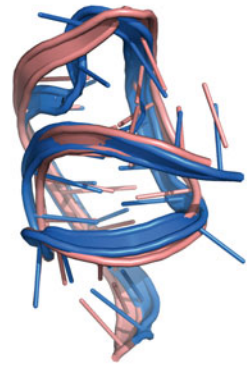
**C****D**

Fig. 1 (a) Entering inputs necessary to model the –1 frameshifting element from BWYV to the FARFAR2 ROSIE server’s “basic interface.” Analysis of 20,000 models resulting from application of FARFAR2 to a viral pseudoknot from beet western yellows virus. **(b)** Excerpts from the page on the FARFAR2 ROSIE server reporting final results. **(c)** Plotting the all-atom RMSD to the lowest-energy (in Rosetta energy units, or REU) FARFAR2 structure suggests a scoring function that favors a single conformation and numerous models within 3 Å of this structure. **(d)** The second-lowest-energy cluster (in pink) has an RMSD of only 3.24 Å to the experimental structure (in salmon); in contrast, several other clusters in the top 10 are 7–10.5 Å away

2.1.2 Secondary Structure Specification

The user should provide the RNA secondary structure in dot-bracket notation. Rosetta uses a common extension to dot-bracket notation that uses brackets other than parentheses to specify pseudoknots. Pseudoknots through third order may be expressed using matching square [], curly {}, and angled <> brackets. For the frameshifting element of 1L2X, the dot-bracket secondary structure is:

```
.((((((...[[[.))))).....]]]
```

Pseudoknots of higher orders are rare but are found in a handful of structures, such as the eight-stranded nanosquare (PDB code: 3P59) [15]. In these cases, it may be necessary to specify higher order pseudoknots using *matched lowercase letters* from a to z. Critically, because the pairing partners of these letters are ambiguous, each distinct fourth-order pseudoknot must be specified with a distinct letter. As above, any chain boundaries ought to be specified via commas.

2.1.3 Specification of the Number of Structures Generated

There is currently no hard and fast rule for how many structures to generate, but we suggest initial runs start with 1000 models and increase beyond that if convergence is not achieved (as evaluated in the next Sect. 2.1.4). The optimal number of models depends on the size and complexity of the modeling problem at hand, the computational resources available, and the RMSD accuracy required for whatever downstream application requires structure modeling. It is possible that there is no way to achieve confident 5.0 Å RMSD predictions on the user's modeling problem of interest even using a million CPU-hours, and it is possible that their modeling problem is simple enough that the entire space of plausible structures itself barely spans more than 5.0 Å RMSD (e.g., a simple stem-loop). As a baseline heuristic, we generally see significant RMSD convergence over the first 1000 structures generated, even for structures of some complexity, if pseudoknot interactions (as in this example) or other information (tertiary contact templates, experimental data; see below) are available. It is unlikely that FARFAR2 will sample significantly closer-to-native structures past that point. That said, problems as small as the viral pseudoknot RNA in 1L2X are relatively inexpensive in computational cost, so for the purpose of illustrating this example thoroughly, we elect to generate 20,000 structures.

2.1.4 Analysis of the Resulting Structural Ensemble

The FARFAR2 ROSIE server generates several useful analyses from the resulting structures. First, it takes the lowest-energy structure from the ensemble and computes the all-heavy-atom RMSD of each structure to this model. Plotting the resulting ensemble (Fig. 1a) helps indicate how cleanly the modeling has converged on a single answer. If there are energetic minima far from the global minimum, this undermines confidence in the modeling and suggests two possibilities. First, sampling may be incomplete, and the true global minimum, yet to be identified, is significantly lower in energy than any minima so far identified. Second, the true minimum may have been identified, but the energy function does not adequately distinguish it from the other minimum structures already identified.

The server also clusters the resulting ensemble and finds the average RMSD among the top 10 cluster centers by energy. This

value is known to predict the RMSD to native of the best cluster center by the equation $y = 0.81x + 3.69 \text{ \AA}$, where y is the predicted RMSD to native for the closest cluster and x is the average pairwise RMSD of the top 10 cluster centers, with an R^2 of 0.84 [9]. There is a similar relation predicting not just the RMSD error but the uncertainty on this prediction: $y_{err} = 0.91x_{err} + 0.09 \text{ \AA}$. In this case, the predicted best RMSD for the BWYV cluster centers is $9.8 \pm 1.85 \text{ \AA}$. Most of the top 10 clusters indeed have RMSD to the crystal structure 1L2X of 7–10.5 \AA , consistent with the predicted accuracy. In this favorable case, the second-best energy cluster turns out to be quite close to the experimental structure, achieving an actual all-heavy-atom RMSD of 3.24 \AA (Fig. 1b).

2.2 The “Advanced” Interface to the FARFAR2 ROSIE Server

Next, we illustrate the use of the advanced interface (Fig. 2a, b) on the twister sister ribozyme structure 5Y87 [16], the experimental structure corresponding to RNA-Puzzle 20. During our modeling of this problem for RNA-Puzzles, we made use of a template structure for a hypothesized tertiary contact (a T-loop/intercalation interaction), and the advanced interface is necessary to supply this information.

2.2.1 Sequence Specification Through a Specially Formatted FASTA File

We employ a specially formatted FASTA file designed to encode the desired sequence numbering as well as the sequence. The benefits of this FASTA file are significant. First, the specification of custom numbering and chain codes allows the FARFAR2 code to understand desired residue-residue correspondences between models and a provided experimental structure, allowing for the direct computation of native structure RMSD during the simulation. Second, that same correspondence allows for the specification of template structures (see Sect. 2.2.6) that give fixed coordinates for particular nucleotides. The FASTA for the 5Y87 modeling challenge is:

```
>rna_puzzle_20_t_loop A:1-18
accgcaaggcgcgacggc
>rna_puzzle_20_t_loop B:1-50
gccgcccugggucaaguccagccacgcucggcgugggcgcucaugggu
```

The FASTA specification allows the study of sequences including chemically modified nucleotides, which must be indicated using special Rosetta nomenclature: indicating the one-letter code as X and specifying a modified base using the PDB three-letter code, enclosed in brackets. (Thus, the nucleotide dihydrouridine, common in tRNAs, which is found in the PDB as H2U, is indicated in a sequence as x[H2U].) The brackets eliminate potential ambiguity between three-letter codes and one-letter codes. The twister sister ribozyme includes no chemically modified nucleotides, so this specific capability is unnecessary.

2.2.2 *Secondary Structure Specification Through an Uploaded File*

In the advanced interface, we supply the secondary structure through a file, rather than a secondary structure string. Unlike when specifying a secondary structure string, the user should not use commas or other characters to separate chains; the FASTA already indicates where chains begin and end.

The secondary structure for our twister sister modeling problem is:

```
(((((...((((((...(((...)))...)))...)))...)))...)))
```

2.2.3 *Specification of Noncanonical Pairs*

The “ordinary” secondary structure, as specified above, can contain only Watson-Crick base pairs and G-U wobble pairs. Any base pair indicated above will be assumed to have that standard geometry, and base pairs incapable of a canonical Watson-Crick pairing will prevent job submission. Many noncanonical base pairs nonetheless exhibit highly stereotyped configurations that engage the Hoogsteen or sugar edges of one or both bases or that engage the Watson-Crick edges in a parallel/trans orientation [17]. For structures known to contain such base pairs, from local motifs like kink turns [18] to tertiary contacts like tetraloop/receptors [19–21], supplementing the secondary structure with this information can be helpful but is rarely available in de novo modeling scenarios, so it has not been widely explored with FARFAR2.

Nevertheless, the advanced interface provides two ways to provide noncanonical pair information. First, the user may provide a “general” secondary structure file formatted just the same as above. Any pairs provided in the “general” secondary structure may be satisfied using any combination of nucleobase edges in any orientation, drawn from a database of validated base pairing orientations.

Alternatively, if aspects of the correct noncanonical base pair is known for sure, they may be specified individually in text, formatted like so:

```
A:18 A:55 W H A A:24 A:72 X S C
```

The string above stipulates that base 18 of chain A and base 55 of chain A must make an antiparallel pair between the Watson-Crick edge of A:18 and the Hoogsteen edge of A:55, and A:24 and A:72 must make a cis pair between any edge of A:24 and the sugar edge of A:72. The permissible base edges are Watson-Crick (W), Hoogsteen (H), sugar (S), and “any” (X), while orientations may be specified as parallel (P)/antiparallel (A) orientation of base normals or through the cis (C)/trans (T) nomenclature of Leontis and Westhof [17], as well as any (X).

For the RNA-Puzzle 20 twister sister modeling problem 5Y87, there was such a hypothesized set of noncanonical interactions involving a T-loop motif, but this set is actually captured by a local template structure (*see* Subheading 2.2.6), and so specification of noncanonical pairs is unnecessary to reproduce it.

2.2.4 Chain Connections

Sometimes there are ambiguities in experimental data intended to guide structure determination. For example, some datasets employing cross-linking or long-distance cleavage information indicate the general proximity of two sets of nucleotides, but no indication as to what nucleotides those could be. “Chain connections” allow users to indicate that there should be a base pair of *some* type – potentially noncanonical – between two sets of nucleotides, without making any assumptions about what the nature or identity of the base pair should be. This is potentially useful for systems with multiple chains, where the “register” of base pairing or tertiary contacts between two chains may be ambiguous. For 5Y87, the secondary structure is known unambiguously, thanks to previously published analysis of twister sister ribozymes.

2.2.5 Constraints

Rosetta’s concept of constraints is equivalent to the idea of energetic *restraints* in molecular dynamics. It is common to express certain types of experimental data as energetic restraints that reward a pair of atoms for being a certain distance apart. The FARFAR2 ROSIE server supports the specification of constraint files using Rosetta’s documented constraint file syntax. A specific example related to multiplexed •OH cleavage analysis (MOHCA) experiments is provided in Sect. 2.3.1.

The server also accepts two additional parameters governing how constraints are applied. First, users may specify the weight applied to constraints, which permits constraints to influence or outright dominate the energy function. Second, users may alter the way that constraints are applied throughout the course of the low-resolution phase of FARFAR2. Choosing “staged constraints” ensures that constraints between residues that are close to each other in primary sequence are applied earlier in the simulation. Prioritizing local interactions in this way appears to help FARFAR2 discover more solutions that satisfy the constraints.

2.2.6 Input Template PDB Files

Taking advantage of any known homology of the RNA modeling problem to previously known structural templates accelerates the process significantly and permits a smaller computational expenditure to deliver superior results [22]. The homology does not have to extend over the entire modeled RNA to aid modeling – homology arising from the “modularity” of many RNA motifs, many of which fold to highly similar structures in different contexts [23], is also valuable and illustrated below. FARFAR2 permits the

specification of template structures, whose coordinates are kept absolutely fixed during fragment assembly and moved only through energy minimization if desired.

While there can be significant benefits to supplying a template for any junction of an RNA, there are two cases that are especially worth highlighting and have recurred in RNA-puzzles and real-world problems in our lab. First, it can be difficult to model small molecule binding sites with algorithms like FARFAR2 and scoring functions not specialized to that task. But binding sites are often well conserved between RNAs of the same or similar families, and so the junction surrounding, e.g., the S-adenosylmethionine binding site of the SAM-I riboswitch, may guide modeling of SAM-IV [9]. Second, tertiary contacts often require precise orientations to form correctly, and during the low-resolution fragment assembly stage, the energetic minima are not particularly deep, so even if a tertiary contact is sampled during fragment assembly, only a fraction of models will retain the contact at the end. Thus, local templates of tertiary contacts can enormously focus sampling.

The RNA-Puzzle 20 twister sister modeling problem includes a local template for the intercalated T-loop formed by an adenosine from the catalytic two-way junction and an apical loop distant in the secondary structure. We were able to infer this interaction by a simple analogy to a previously studied twister sister ribozyme, RNA-Puzzle 19, deposited under PDB code 5T5A [24].

We have made additional webservers available for tasks that are useful for manipulating local templates, such as threading on a new sequence (frequently a perfectly valid template will have differences in helix sequence, e.g., that do not affect its quality) and renumbering to match the modeling problem at hand. These webservers may be found at https://rosie.rosettacommons.org/rna_thread and https://rosie.rosettacommons.org/renumber_pdb.

To thread a new RNA sequence onto a PDB (Fig. 2a), supply the starting PDB and the new desired sequence as “acgu” to the server. The resulting PDB file will have its numbering “reset” to A:1- N , where N is the total number of nucleotides in your PDB structure. For the twister sister modeling problem, no threading was needed as the sequences were identical in the intercalated T-loop between the template structure 5T5A and the new twister sister RNA.

To update the chains and numbers in a PDB (Fig. 2b), supply the starting PDB and the new numbering in the same sort of format used in FASTA files above. That is, chain and residue numbering is described as “A:1” or “B:5” – while multiple consecutive residues with the same chain may be summarized as “A:1–20” or equivalent. We do renumber the T-loop from 5T5A, as it represents an interaction between adenosine residue A:8 and a T-loop comprising A:22–26 in its original context, while in the new twister

A Submit a new *rna_thread* job

Job short description (visible in queue):

Start PDB

The PDB you want to thread.

No file chosen

Sequence

The sequence (as "acgu") desired.:

B Submit a new *renumber_pdb* job

Job short description (visible in queue):

Start PDB

The PDB you want to renumber.



rna_puzzle_20_t_loop_START1_5y87_RNA.pdb

New numbering

The desired numbering, formatted to indicate the chains and numbers required, e.g. "A:1-20 B:6 C:1-20":

Fig. 2 (a) Interface for the *rna_thread* server. **(b)** Interface for the *renumber_pdb* server, prepared to renumber the starting template as needed

sister ribozyme that we seek to model, the interaction is between A:7 and B:12–16, so our input numbering is:

A:7 B:12–16

The T-loop from 5T5A, renumbered to match the 5Y87 structural context, may be found in https://github.com/DasLab/FARFAR2_modeling_examples. This template turned out to have 0.36 Å RMSD from the experimental structure.

2.2.7 Alignment PDB Structure

There are many situations where the user might have multiple input template structures and an approximate understanding of where they might sit in space. To encode this expectation, the user can supply one “alignment PDB” encoding that understanding. FARFAR2 will impose energetic restraints on each atom in generated models corresponding to an atom in the alignment PDB, penalizing conformations where they stray more than 4.0 Å away from this ideal location. Thus, models will reliably have the desired orientation, but small deviations necessary to accommodate sequence context changes from the template may be permitted.

This feature is also useful for understanding the best possible models achievable by FARFAR2 that are close to a target structure like an experimental structure. Comparison of energies of such near-native models with unrestrained de novo modeling has been useful in understanding limitations in the FARFAR2 energy function [9].

2.2.8 Native PDB Structure

The sequence and numbering of any specified native structure must correspond exactly to the provided FASTA file. This structure will of course be unavailable for actual blind challenges approached using the webserver, but for benchmarking cases like the twister sister ribozyme, it is available (5Y87) and we use it here in this example. This file is also supplied in the DasLab/FARFAR2_modeling_examples repository (https://github.com/DasLab/FARFAR2_modeling_examples).

2.2.9 High-Resolution Minimization Settings

Following fragment assembly, FARFAR2 refines structures through continuous minimization of torsion angles in an all-atom scoring function. This is not strictly mandatory, but structures that have not been optimized in this way will possess nonphysical chain breaks and clashes. If minimization is desired (it is active by default), then users may select one of three high-resolution energy functions. The default is the best-performing setting and the standard for FARFAR2.

By default, residues drawn from template structures are not minimized. Users may additionally specify residues from input template structure that may move during minimization. The input format indicates each residue by residue number and chain, like “A: 1”; a series of residues may be specified as “A: 1-3 A: 5” and so on. This option is especially useful when the new modeling context for a template is somewhat different from its original context. Because we used an existing twister sister ribozyme structure for the intercalated T-loop template in this modeling, we did not use this option, but if we had used a different structure with a T-loop – say, a tRNA – it may have been helpful.

2.2.10 Low-Resolution Fragment Assembly Settings

The initial low-resolution fragment assembly stage also has several manipulable parameters. The user can raise or lower the simulation temperature, which affects how likely a fragment move is to be accepted. They may select the current, updated fragment database; the previous Rosetta default in effect from 2010 to 2019; or the original fragments that only used one structure of an *E. coli* 23S rRNA. They may enrich the existing fragment database by adding in torsional combinations drawn from a Gaussian centered at the torsions of each experimental fragment. Additionally, they may enforce an approximate symmetry condition: For example, if the user is interested in modeling a duplex, then they may wish to

ensure that any fragment move is applied concurrently to each strand. These non-default options have not been explored widely.

The final condition that modifies the low-resolution phase can be important for rigorous benchmarking: an option originally added to Rosetta when the Das lab needed a standard for comparison in stepwise Monte Carlo benchmarking [25]. The user may choose to *exclude homologous fragments from the fragment library* that resemble the native structure too closely. Using a structure of a twister sister ribozyme like 5Y87 as a benchmark case for testing FARFAR2 would be unfair if fragments from that same ribozyme were present in the library. The fragment exclusion algorithm looks at every 6-mer sequence in the native PDB and removes any fragments that match those 6-mers in sequence and whose substitution would result in a conformation closer than the provided RMSD radius to the native conformation. The sequence match for what might get removed defaults to matching purine/pyrimidine identity but could either ignore sequence entirely or require an exact sequence match. For RNA-Puzzle 20, PDB 5Y87, we supply an RMSD radius of 1.2 Å, following the FARFAR2 study [9].

2.2.11 Experimental Data

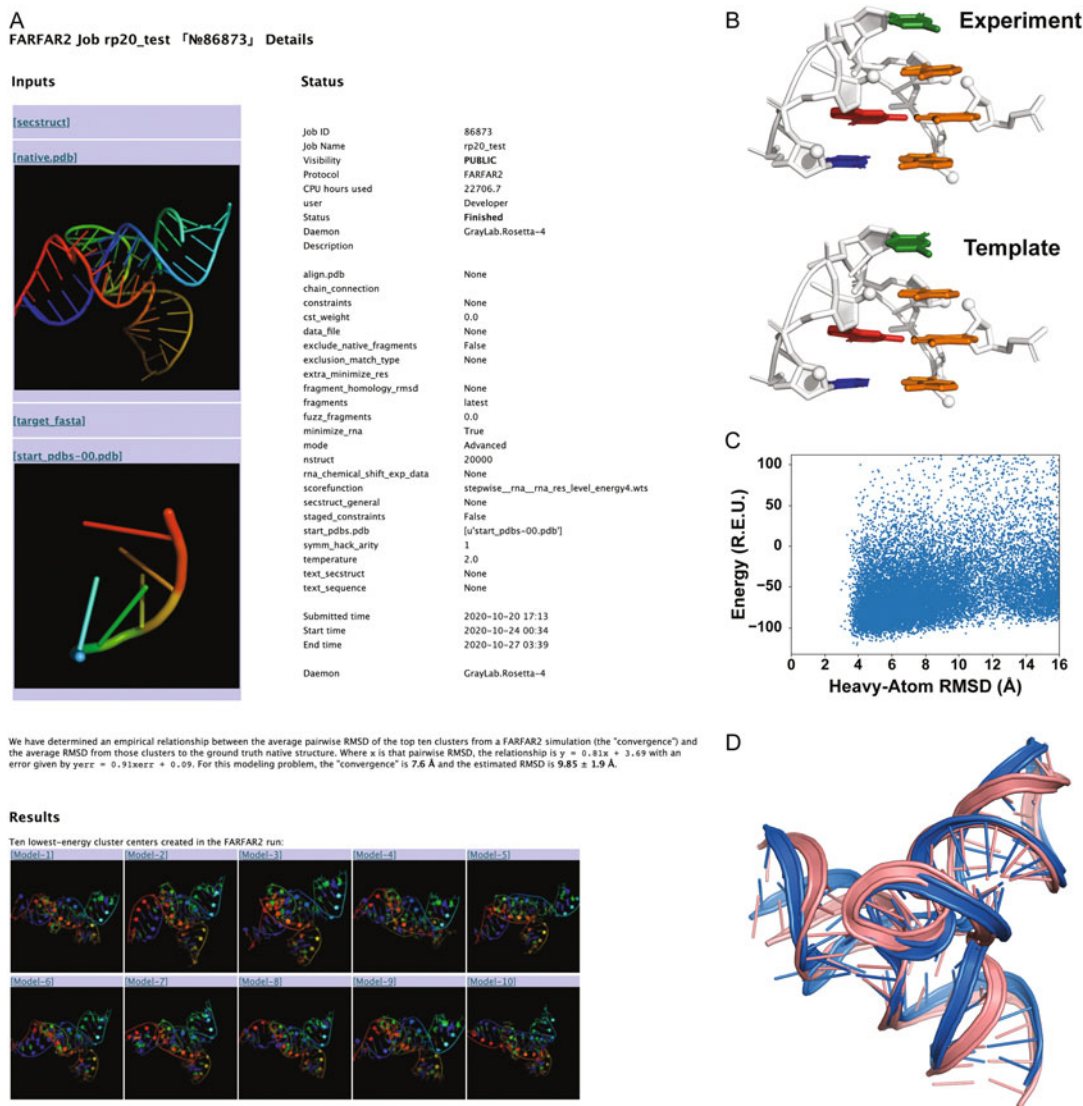
The user may have external experimental data to guide the modeling process. NMR chemical shifts may be specified using a variant on the STAR 2.1 format. Details of the full CS-ROSETTA-RNA protocol [26] are extensively documented in Rosetta demos available at https://www.rosettacommons.org/demos/latest/public/cs_rosetta_rna/README and application documentation at https://www.rosettacommons.org/docs/latest/application_documentation/rna/CS-Rosetta-RNA, but augmenting FARFAR2 modeling with chemical shift scoring requires only the specification of a STAR 2.1 format chemical shifts file as described here. Sub-heading 2.3 gives an example.

2.2.12 Number of Structures to Generate

The number of structures needed for a whole RNA structure varies considerably with its size and complexity. The presence of the intercalated T-loop tertiary contact aids the modeling, as that tertiary contact is sufficient to give the whole RNA a globular fold and restricts the possible low-energy structures. In our original blind challenge effort, and in our subsequent simulated benchmark, we were able to generate structures closer than 4.0 Å RMSD to the crystal structure with only a few thousand models. For the RNA-Puzzle 20 twister sister ribozyme 5Y87, we generate 20,000 models, simply to provide a thorough exploration of the modeling problem for this work.

2.2.13 Analyze the Results

The FARFAR2 ROSIE server (Fig. 3a) provides a set of data for this twister sister problem similar to the first pseudoknot benchmark case, albeit with some small differences. Because we provided a native PDB structure in this example, the server does not rescore



We have determined an empirical relationship between the average pairwise RMSD of the top ten clusters from a FARFAR2 simulation (the “convergence”) and the average RMSD from those clusters to the ground truth native structure. Where x is that pairwise RMSD, the relationship is $y = 0.81x + 3.69$ with an error given by $y_{err} = 0.31x_{err} + 0.09$. For this modeling problem, the “convergence” is 7.6 Å and the estimated RMSD is 9.85 ± 1.9 Å.

Fig. 3 (a) The job submission page that appears upon submitting the suggested inputs to simulate RNA-Puzzle 20. (b) A close-up comparison of the excellent template T-loop structure from 5T5A, as compared to the experimental coordinates for the target ribozyme 5Y87. (c) Analysis of the ensemble of FARFAR2 structures generated. Plotting the all-atom RMSD to the lowest-energy structure suggests good sampling and a scoring function that favors a single conformation. (d) While each cluster center is each fairly close to the experimental conformation (in marine), one of them (in pink) is especially near, with an RMSD of 3.91 Å

the ensemble to the lowest-energy model. Instead, it plots the score against RMSD to the provided native structure. In part, we can attribute the method’s success on a modeling problem of this complexity to the high similarity in T-loop conformation between the template and target structure (Fig. 3b). The resulting ensemble (Fig. 3c) may be interpreted similarly to the simpler pseudoknot

modeling problem above (Fig. 1), albeit with the certainty that the lowest RMSD structures are the most native-like due to the specification of the native structure as a reference here. So energetic minima far from the native structure must indicate scoring function issues. In this case, the resulting top 10 cluster centers automatically generated by the webserver have average inter-model RMSD of 7.6 Å, indicating 9.9 ± 1.9 Å minimum cluster RMSD to native. In fact, the clusters are each quite similar to the experimental structure and range from 5.3 to 9.3 Å, and the best cluster has a significantly superior RMSD, at only 3.91 Å RMSD (Fig. 3d). This suggests that the use of such an accurate template significantly helped structure prediction exceed typical expectations.

2.3 Additional Illustrations of Advanced Interface: Experimental Data

Experimental data can dramatically improve convergence and accuracy of RNA modeling. The FARFAR2 ROSIE server is well-equipped to handle two kinds of experimental data, MOHCA and NMR ^1H chemical shift data, briefly discussed here. (The recent Ribosolve pipeline integrates Rosetta RNA de novo modeling with cryo-EM; a separate ROSIE server is under development for that application and is not described here.)

2.3.1 MOHCA-Seq with FARFAR2

MOHCA [27] and MOHCA-seq [28] experiments use tethered hydroxyl radical sources and sequencing readouts to discover “strong” and “weak” signals of nucleotide-nucleotide proximity. These signals can be used to guide FARFAR2 modeling; each signal results in a restraint expressed as the sum of two functions, whose weights are given by the strength of the constraint. A “strong” restraint between chain A nucleotides 2 and 38 at their O2’ and C4’ atoms would be specified via:

```
AtomPair O2' 2A C4' 38A FADE 0 30 15 -4.00 4.00
AtomPair O2' 2A C4' 38A FADE -99 60 30 -36.00 36.00
```

Omission of the chain letter leads Rosetta to interpret the sequence position as an absolute number within the PDB (i.e., sequentially starting from 1), which may not match the numbering of the user’s PDB. The parameters for the FADE constraint are described in https://www.rosettacommons.org/docs/latest/rosetta_basics/file_types/constraint-file. The specification above gives a penalty smoothly ramping up to 4 Rosetta energy units as the inter-atom distance shifts away from 15 Å down to 0 Å and up to 30 Å and an additional penalty of up to 36.0 Rosetta units if the inter-atom distance exceeds 30 Å. A “weak” restraint would be:

```
AtomPair O2' 2A C4' 38A FADE 0 30 15 -0.80 0.80
AtomPair O2' 2A C4' 38A FADE -99 60 30 -7.20 7.20
```

that is, one-fifth the strength of the “strong” restraint.

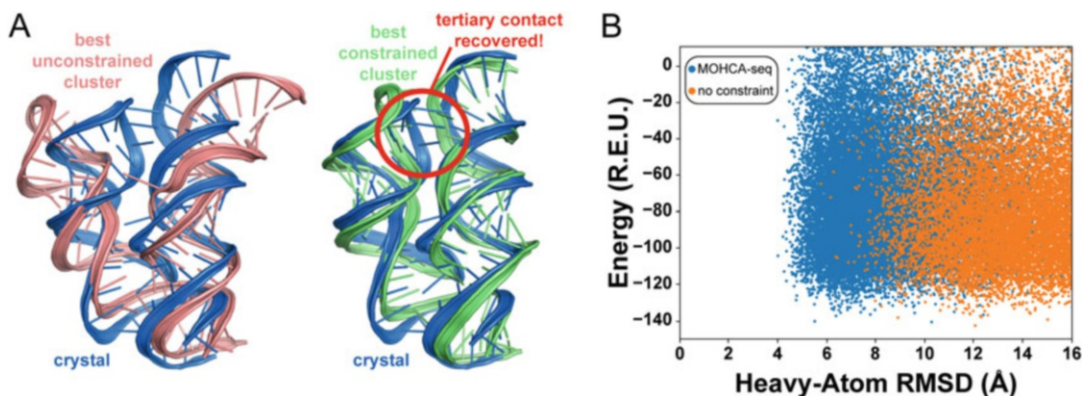


Fig. 4 A comparison of FARFAR2 simulation results run with and without MOHCA-seq constraints on a *V. cholerae* c-di-GMP riboswitch. (a) The simulation run without the benefit of MOHCA-seq constraints is unable to recover the key tertiary contact that defines the global fold of this riboswitch (left), while a simulation with MOHCA-seq constraints finds that tertiary contact naturally (right). (b) Looking at the ensemble of generated models as a whole, a large proportion of the models from the constrained simulation are closer to the experimental model than even the best models generated in the unconstrained simulation

We show the results of the FARFAR2 ROSIE server simulations conducted with and without MOHCA-seq constraints on a *Vibrio cholerae* c-di-GMP riboswitch (PDB code: 3IRW) [29]. Unlike the original protocol using FARFAR [28], FARFAR2 does not require pre-generation of helix ensembles, manual selection of models to minimize, or manual selection of a fraction of models to cluster. The constrained simulations approach much closer to the crystal conformation: Its second-lowest-energy cluster is at 5.52 Å RMSD, while the closest cluster among the top 10 for the unconstrained simulation is 8.91 Å RMSD (Fig. 4a). This advantage is far from chance; more than half of the models produced in the constrained simulation have RMSD less than 8.0 Å, versus less than 1% for the unconstrained simulation (Fig. 4b). FASTA, secondary structure, and constraint files necessary to reproduce this simulation are included in the GitHub repository (see Appendix).

2.3.2 Chemical Shift-Guided FARFAR2 (CS-Rosetta-RNA)

^1H chemical shifts alone can provide powerful information for constraining RNA folds, in many cases enabling atomic accuracy without requiring additional NMR experiments [26]. Due to improvements in the FARFAR2 protocol at baseline, the difference in performance on benchmark cases from the original CS-Rosetta-RNA study [26] is not as stark as for the earlier sampling protocol and scoring function (FARFAR rather than FARFAR2). Nonetheless, improvements from ^1H chemical shifts remain apparent, as illustrated by the tandem GA mismatch case 1MIS [30]. A comparison of two 500-structure ensembles shows that modeling guided by chemical shifts generates a much harsher score penalty for

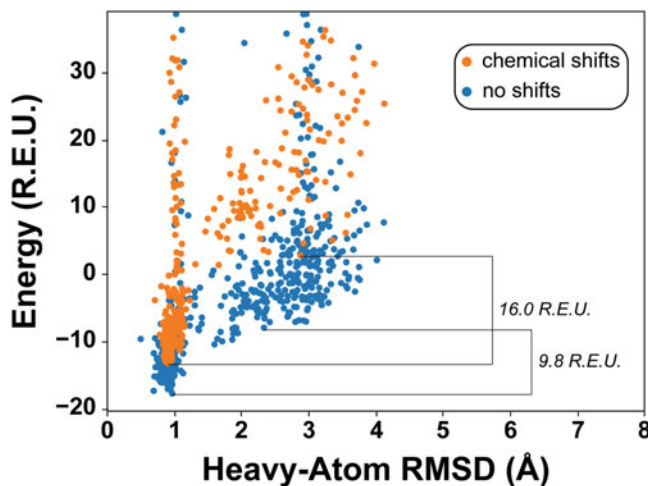


Fig. 5 A comparison of FARFAR2 simulations run with and without chemical shifts on a duplex RNA containing tandem GA pairs. FARFAR2 can find the correct structure even without chemical shift data, but the energy function favors correct conformations by a larger margin. We show the energy gap differentiating the lowest-energy correct structure (at most 1.0 Å RMSD) versus the lowest-energy incorrect structure (at least 2.0 Å RMSD)

models >2.0 Å of an experimental structure (an energy gap of 16.0 rather than 9.8 Rosetta energy units) (Fig. 5).

3 Conclusions

The ROSIE server for FARFAR2 provides a web interface to Rosetta's application for de novo modeling of complex RNA folds, targeted to users at diverse levels of expertise. Important stages of modeling, such as reducing a modeled ensemble to a handful of representative structures, are automated, enabling the user to come away with both full modeling output and also the most important models and their expected accuracy. Although the computational demands of RNA molecules remain high and atomic accuracy for intricate interactions remains difficult to achieve, the ROSIE interface ensures that nonexpert can access Rosetta RNA modeling code in a state that keeps pace with ongoing developments.

Acknowledgments

We thank Sergey Lyskov for his maintenance of the ROSIE web-server platform; lab members Rachael Kretsch, Ramya Rangan, and Ivan Zheludev for server testing; and Rachael Kretsch for reading the manuscript. We acknowledge funding from the National Institutes of Health (R35 GM 122579 and R21 CA 219847).

Appendix

All inputs to reproduce the modeling described here, as well as sample output data, are released freely at https://github.com/DasLab/FARFAR2_modeling_examples.

References

- Cech TR, Steitz JA (2014) The noncoding RNA revolution - trashing old rules to forge new ones. *Cell* 157:77–94
- Das R, Karanicolas J, Baker D (2010) Atomic accuracy in predicting and designing non-canonical RNA structure. *Nat Methods* 7: 291–294. <https://doi.org/10.1038/nmeth.1433>
- Ditzler MA, Otyepka M, Šponer J, Walter NG (2010) Molecular dynamics and quantum mechanics of RNA: conformational and chemical change we can believe in. *Acc Chem Res* 43(1):40–47. <https://doi.org/10.1021/ar900093g>
- Laing C, Schlick T (2011) Computational approaches to RNA structure prediction, analysis, and design. *Curr Opin Struct Biol* 21: 306–318
- Cao S, Chen SJ (2005) Predicting RNA folding thermodynamics with a reduced chain representation model. *RNA* 11(12):1884–1897. <https://doi.org/10.1261/rna.2109105>
- Lyskov S, Chou FC, Conchúir SÓ et al (2013) Serverification of molecular Modeling applications: the Rosetta online server that includes everyone (ROSIE). *PLoS One* 8(5):e63906. <https://doi.org/10.1371/journal.pone.0063906>
- Moretti R, Lyskov S, Das R et al (2018) Web-accessible molecular modeling with Rosetta: the Rosetta online server that includes everyone (ROSIE). *Protein Sci* 8(5):e63906. <https://doi.org/10.1002/pro.3313>
- Leman JK, Weitzner BD, Lewis SM et al (2020) Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nat Methods* 17(7):665–680
- Watkins AM, Rangan R, Das R (2020) FARFAR2: improved De novo Rosetta prediction of complex global RNA folds. *Structure* 28(8): 963–976. <https://doi.org/10.1016/j.str.2020.05.011>
- Magnus M, Boniecki MJ, Dawson W, Bujnicki JM (2016) SimRNAweb: a web server for RNA 3D structure modeling with optional restraints. *Nucleic Acids Res* 44(W1): W315–W319. <https://doi.org/10.1093/nar/gkw279>
- Biesiada M, Pachulska-Wieczorek K, Adamiak RW, Purzycka KJ (2016) RNAComposer and RNA 3D structure prediction for nanotechnology. *Methods* 103:120–127. <https://doi.org/10.1016/j.jymeth.2016.03.010>
- Krokhotin A, Houlihan K, Dokholyan NV (2015) iFoldRNA v2: folding RNA with constraints. *Bioinformatics* 31(17):2891–2893. <https://doi.org/10.1093/bioinformatics/btv221>
- Parisien M, Major F (2008) The MC-fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature* 452:51–55. <https://doi.org/10.1038/nature06684>
- Egli M, Minasov G, Su L, Rich A (2002) Metal ions and flexibility in a viral RNA pseudoknot at atomic resolution. *Proc Natl Acad Sci U S A* 99:4302–4307. <https://doi.org/10.1073/pnas.062055599>
- Zheng L, Mairhofer E, Teplova M et al (2017) Structure-based insights into self-cleavage by a four-way junctional twister-sister ribozyme. *Nat Commun* 8(1):1–12. <https://doi.org/10.1038/s41467-017-01276-y>
- Leontis NB, Westhof E (2001) Geometric nomenclature and classification of RNA base pairs. *RNA* 7(4):499–512. <https://doi.org/10.1017/S1355838201002515>
- Huang L, Lilley DMJ (2016) The kink turn, a key architectural element in RNA structure. *J Mol Biol* 428(5):790–801
- Abramovitz DL, Pyle AM (1997) Remarkable morphological variability of a common RNA folding motif: the GNRA tetraloop-receptor interaction. *J Mol Biol* 266(3):493–506. <https://doi.org/10.1006/jmbi.1996.0810>
- Geary C, Baudrey S, Jaeger L (2008) Comprehensive features of natural and in vitro selected GNRA tetraloop-binding receptors. *Nucleic Acids Res* 36(4):1138–1152. <https://doi.org/10.1093/nar/gkml048>
- Fiore JL, Nesbitt DJ (2013) An RNA folding motif: GNRA tetraloop-receptor interactions. *Q Rev Biophys* 46(3):223–264. <https://doi.org/10.1017/S0033583513000048>

21. Cheng CY, Chou FC, Kladwang W et al (2015) Consistent global structures of complex RNA states through multidimensional chemical mapping. *elife* 4:e07600. <https://doi.org/10.7554/eLife.07600>
22. Smith KD, Lipchock SV, Ames TD et al (2009) Structural basis of ligand binding by a c-di-GMP riboswitch. *Nat Struct Mol Biol* 16(12): 1218–1223. <https://doi.org/10.1038/nsmb.1702>
23. Watkins AM, Rangan R, Das R (2019) Using Rosetta for RNA homology modeling. *Methods Enzymol* 623:177–207. <https://doi.org/10.1016/bs.mie.2019.05.026>
24. Bisaria N, Greenfeld M, Limouse C et al (2016) Kinetic and thermodynamic framework for P4-P6 RNA reveals tertiary motif modularity and modulation of the folding preferred pathway. *Proc Natl Acad Sci USA*. <https://doi.org/10.1073/pnas.1525082113>
25. Liu Y, Wilson TJ, Lilley DMJ (2017) The structure of a nucleolytic ribozyme that employs a catalytic metal ion. *Nat Chem Biol* 13(5):508–513. <https://doi.org/10.1038/nchembio.2333>
26. Daldrop P, Reyes FE, Robinson DA et al (2011) Novel ligands for a purine riboswitch discovered by RNA-ligand docking. *Chem Biol* 18(3):324–335. <https://doi.org/10.1016/j.chembiol.2010.12.020>
27. Watkins AM, Geniesse C, Kladwang W et al (2018) Blind prediction of noncanonical RNA structure at atomic accuracy. *Sci Adv* 4(5): eaar5316. <https://doi.org/10.1126/sciadv.aar5316>
28. Sripakdeevong P, Cevcec M, Chang AT et al (2014) Structure determination of noncanonical RNA motifs guided by 1 H NMR chemical shifts. *Nat Methods* 11(4):413–416. <https://doi.org/10.1038/nmeth.2876>
29. Wu M, Turner DH (1996) Solution structure of (rGCGGACGC)₂ by two-dimensional NMR and the iterative relaxation matrix approach. *Biochemistry* 35(30):9677–9689. <https://doi.org/10.1021/bi960133q>



Automated 3D Design and Evaluation of RNA Nanostructures with RNAMake

Chris P. Jurich and Joseph D. Yesselman

Abstract

Despite growing interest in applying RNA's unique structural characteristics to solve diverse biotechnology and nanotechnology problems, there are few computational tools for targeted tertiary design. As a result, RNA 3D design is traditionally slow, resource-consuming, and dependent on expert modeling. In this chapter, we discuss our recently developed software package: RNAMake, a set of applications capable of designing RNA tertiary structures to solve various relevant nanotechnology problems and provide basic thermodynamic calculations for the generated designs. We provide in-depth examples and instructions for designing example RNA nanostructures such as minimal RNA sequences containing a single tertiary contact, generating RNAs that stabilize small-molecule ligands, and building tethers that link ribosomal subunits together. We also highlight the addition of a new Monte Carlo design algorithm and the ability to estimate the thermodynamic contribution of helical elements in RNA 3D structures.

Key words RNA tertiary structure, RNA design, Computer-guided design

1 Introduction

The emerging field of RNA nanotechnology seeks to generate nanoscale machines for biomedicine that harness RNAs' unique structural properties [1]. Unlike other biomolecules, the tertiary structure of RNA is composed of discrete and reoccurring substructures called tertiary "motifs." Many of these tertiary motifs adopt similar three-dimensional (3D) conformations despite appearing in a diverse set of structural contexts [2–7]. Exploiting symmetry, motif repetition, and expert modeling, these motifs have been assembled into rationally designed structures, including polygons, cubes, and sheets [8–13]. While significant advances have been made, there are two outstanding problems faced in the rational design of RNA 3D nanostructures: (1) Rationally designed RNAs lack the complexity observed in natural RNA machines that contain a larger repertoire of distinct interacting motifs giving rise to the functionality

observed in the biological RNA world [14–16]. (2) Tertiary motifs and while some can be approximated as rigid blocks, they are thermodynamic entities that are best represented by an ensemble of structures. There are known motifs that can radically change conformation in response to small-molecule binders [17]. To ultimately build RNA devices that can respond to the cellular environment requires a detailed description of each motif’s conformational thermodynamics. This chapter will present our recently developed toolkit RNAMake, which seeks to address these challenges with examples on how to run each algorithm.

1.1 RNAMake Software Can Automate the Design of RNA 3D Structures

RNAMake is the first software package that automates the assembly of RNA motifs into compact nanostructures. RNAMake contains (1) an extensive motif dictionary including the nearly 1000 motifs available from high-resolution crystallographic structures and (2) provides enumerative “pathfinding” algorithms to iteratively build up the necessary motifs to solve 3D design problems (Fig. 1a). In our initial study, we challenged RNAMake with three design challenges [18]. First, we revisited a 20-year-old problem of designing an RNA that aligns the two parts of the tetraloop/tetraloop receptor (TTR) within a single compact RNA chain. In total, we generated 16 solutions to this “miniTTR” problem, and through a combination of chemical mapping, Mg^{2+} titrations, and native gel electrophoresis, we observed strong evidence that 11 of the 16 constructs folded as designed, with 1 case confirming atomic accuracy over nearly the entire fold through crystallography. Second, we generated single stranded ribosomes by building a tether between the 23S and 16S rRNAs of the ribosome into a single RNA strand that supports *E. coli*. Growth [19–22]. This 3D design challenge requires solving the RNA motif pathfinding problem over $>100 \text{ \AA}$ distances and avoiding steric collisions with the ribosome’s RNA and protein components. Lastly, we demonstrated that RNAMake could improve the binding properties of two artificially selected RNA aptamers. We accomplished this by utilizing RNAMake to add of peripheral tertiary contacts that ‘lock’ these artificial aptamers into their bound conformation even in the absence of ligand. Such locking contacts could selectively increase the unbound state’s free energy and thereby improve the free energy difference to the bound state, leading to better affinity to small-molecule targets.

1.2 RNAMake- $\Delta\Delta G$: A Predictive Model for Helical RNA 3D Thermodynamics

RNAMake’s other feature is its conformational thermodynamic description of helices and some simple motifs. Helices are decomposed into base pair steps (i.e., two sequential base pairs) which allows for modeling of arbitrary helix sequences with minimal set of structure states. Base pair step conformational ensembles were determined by compiling all instances of that base pair step in structured RNAs from the RNA crystal structure database. This

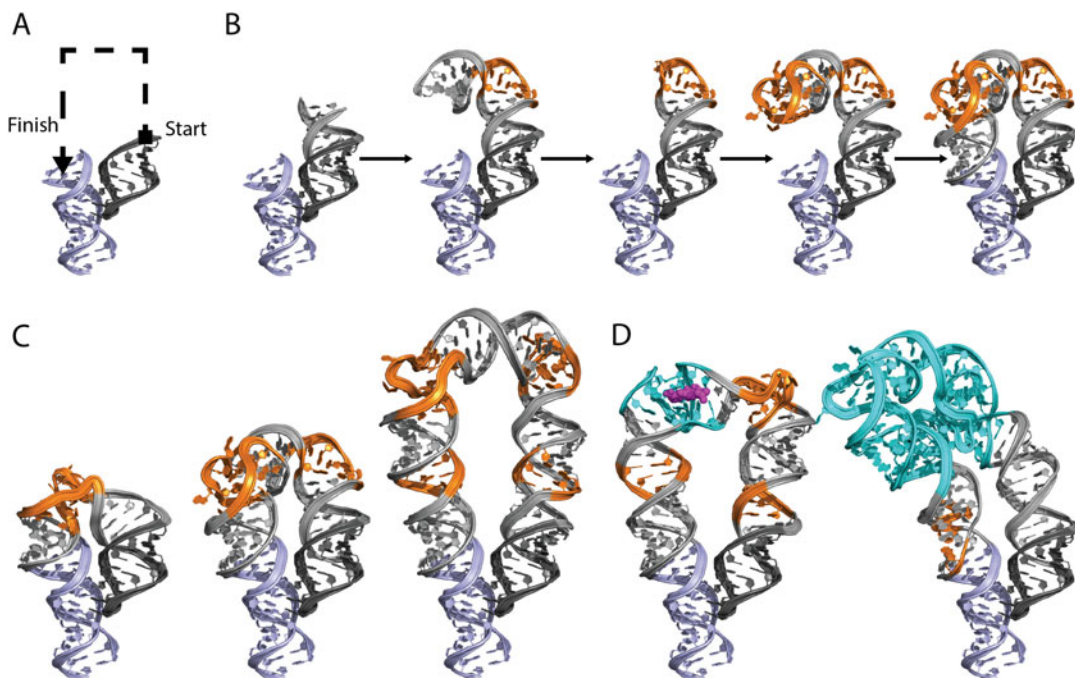


Fig. 1 RNAMake can build RNA 3D scaffolds from a library of motifs. **(a)** The start and end base pair that defines the constraints for RNAMake to build a 3D RNA segment. **(b)** Demonstration of RNAMake scaffold design algorithm, which builds an RNA path via the successive addition of motifs (orange) and helices (gray) from a starting base pair to the ending base pair. **(c)** Defining a solution topology with a specific number of helices and two-way junctions using the Monte Carlo search. **(d)** Incorporating user supplied RNA segments into RNAMake's generated design solution

quantitative 3D model of RNA tertiary assemblies blindly predicted the stabilities of thousands of RNA complexes (< 0.5 kcal/mol accuracy, 5 kcal/mol range). Using RNAMake- $\Delta\Delta G$, we were also able to develop low-resolution conformational thermodynamic descriptions for ~ 400 motifs that have known crystallographic structures [23].

In the following sections, we provide detailed example use cases for RNAMake, including input files, rendered outputs, and the exact command line arguments used. These examples cover both the generation of RNA nanostructures and predictions of their 3D thermodynamic stability.

2 Materials

RNAMake is written in C++11 and can be built with g++ and clang compilers. RNAMake is available for free for academic use at https://simtk.org/frs/?group_id=1749. Each application in RNAMake is built as a separate executable for ease of use. All

examples and files discussed in this book chapter are distributed with RNAMake and are contained in the subfolder “tests.”

3 Methods

3.1 *Automatic Design of RNA Nanostructures*

The RNAMake application `design_rna_scaffold` provides a central hub for the building of RNA 3D segments composed of the iterative buildup of RNA motifs and helices. There is a wide array of potential options; please see Table 1 for all commonly used options. The application requires three command line arguments: `--pdb` which specifies the path to the protein databank (PDB) file that contains the starting RNA structure and `--start_bp` and `--end_bp` which define the start and end of Watson and Crick base pairs to build between. Both must be specified as “[CHAIN ID][NUM]-[CHAIN ID][NUM].” For an example, the pairing between residues 100 and 112 on chain A would be “A100-A112.” Note that although the format of this option is checked before the algorithm runs, the validity of the supplied base pair is not checked until the 3D structure is loaded. The other common arguments are `--designs` to specify the number of solutions and `--dump_pdbs` which outputs a PDB file for each solution. See command line example below:

```
design_rna_scaffold --pdb inputs/min_tetraloop_receptor.pdb
--start_bp "A220-A253" --end_bp "A144-A159"
```

This first example defines the “miniTTR” problem. Building an RNA segment that generates a single continuous RNA strand that contains both the tetraloop and receptor of the tetraloop/tetraloop-receptor tertiary contact (Fig. 1). By default, `design_rna_scaffold` will utilize its classic pathfinding algorithm which balances the length of helical regions with the relative distance to the end base pair (Fig. 1b). This pathfinding search finds the optimal number of motifs and helices to use based on this criterion.

`design_rna_scaffold` outputs two files by default: “default.out” and “default.scores.” The out file stores all information required to regenerate each designed RNA segment. These can be supplied to other RNAMake applications for additional analysis. The score file is a comma delimited summary of all information about each design. A breakdown of what each column represents is summarized in Table 2.

RNAMake now contains a new Monte Carlo-based search that allows the user to specify the exact composition of the designed 3D RNA segment. A user can specify the use of the Monte Carlo-based search by setting `--search_type` to “MC” and supplying a motif path. This motif path informs RNAMake what set of motifs should be used at each position in the constructed RNA segment. Table 3

Table 1
List of commonly used flags and options in design_rna_scaffold application

Argument	Required?	Description	Allowed values
-h,--help	No	Provides information on options from within the command line. Will list a description of all commands listed here	--
Core inputs			
--pdb	Yes	Path to a PDB file containing the starting RNA strand(s). Note that both the starting and ending base pairs within must be Watson-crick	Directory string, /path/to/file.Pdb
--start_bp	Yes	The Watson-crick base pair from which the RNA design will be built. Format is “[CHAIN ID][NUM]-[CHAIN ID][NUM].” for an example, the pairing between residues 100 and 112 on chain A would be “A100-A112.” note that although the format of this option is checked before the algorithm runs, the validity of the supplied base pair is not checked until the 3D structure is loaded	Text, must adhere to format “[CHAIN ID][NUM]-[CHAIN ID][NUM]” (no spaces)
--end_bp	Yes	The Watson-crick base pair to which the RNA design will be built. Format is identical to that of --start_bp. See description above for more detail	Text, must adhere to format “[CHAIN ID][NUM]-[CHAIN ID][NUM]” (no spaces)
--designs	No	The number of designs that design_rna_scaffold attempts to generate. Default value is 1	Integer, greater than 0
--extra_pdbs	No	Allows supplying extra RNA segments in PDB files to be included in the motif_path	List of PDB files that is comma delimited
--log_level	No	Sets the global log level for design_rna_scaffold. Logging will be shown for events at and above the select log level. Default is “info”	In ascending order: Verbose, debug, info, warning, error, fatal
I/O options			
--dump_pdbs	No	Outputs a PDB file for each RNAMake solution	--
--out_file	No	Name of the output file where the program’s results will be stored	String input to a valid location. Ex: /valid/path/my_output.Out

(continued)

Table 1
(continued)

Argument	Required?	Description	Allowed values
--score_file	No	Name of the file where the scoring information for generated designs are stored	String input to a valid location. Ex: /valid/path/my_output.Scores
Search parameters			
--max_helix_length	No	The maximum number of base pairs a helix can be; default = 99	Positive integer greater than --min_helix_length
--min_helix_length	No	The minimum number of base pairs a helix can be; default = 4	Integer greater than 1 and less than --min_helix_length
--motif_path	No	The motif libraries to use in the generation of solutions. See Table 3 for available libraries	A list of motif libraries separated by commas
--no_sterics	No	Flag that turns off steric checks against the supplied RNA structure	--
--only_tether_opt	No	Flag that enables application to ignore supplied structures other than sterics	--
--search_cutoff	No	The largest score that will be accepted for an allowable solution; default = 5.0	Positive floating point number
--search_max_motifs	No	For the pathfinding search, the maximum number of motifs that can be used; default = 999	Positive integer
--search_max_size	No	The maximum number of nucleotides that can be used in a solution; default = 999,999	Positive integer
--search_type	No	The type of search to use; default = "path_finding"	Either "path_finding," "mc," or "exhaustive"
Sequence optimization parameters			
--skip_sequence_optimization	No	Flag that disables sequence optimization for all designs	--
--sequences_per_design	No	The number of sequences to try per motif design. Default is 1	Positive integer
Thermo fluc parameters			
--thermo_fluc	No	Flag to run a thermo fluctuation simulation of the designed segment after sequence optimization	--

Table 2
Description of columns found in design_rna_scaffold score file

Output value	Description
design_num	The design number, ascending from 0
design_score	The design score, how well the solution reached the target base pair
design_sequence	The full sequence of the supplied PDB and the constructed RNA segment. "N"s will appear for nucleotides in helical areas that can be set to any nucleotide in a base pair
design_structure	The full secondary structure in dot bracket notation
motif_uses	Which motifs from the RNAMake motif library were used
opt_num	The sequence optimization number if sequence optimization was performed
opt_sequence	The full sequence with the helix sequence optimized to match the desired secondary structure
opt_score	How well does the solution with the optimized sequence reach the target base pair
thermo_fluc_best_score	If thermo fluctuation simulation is performed, what was the best possible conformation that reached the target base pair
hit_count	How many times out of 100,000 trials did the solution reach the target base pair

Table 3
All motif libraries that are currently available to use in the Monte Carlo motif path declaration

Motif library name	Description
ideal_helices	Idealized helices between size of 2 base pairs to 20 base pairs
Twoway	All two-way junctions
Tcontact	All tertiary contacts
Hairpin	All hairpins
Nway	All junctions that have three or more base pair ends
flex_helices	Helices that were generated by RNAMake- $\Delta\Delta G$ are slightly distorted away from idealized geometry
unique_twoway	Clustered two-way junctions to reduce the total number of possibilities
bp_steps	All helical base pair steps

lists the current available motif libraries that can be included. In the command below, a motif path of “flex_helices,twoday,flex_helices” has been specified.

```
design_rna_scaffold --pdb inputs/min_tetraloop_receptor.pdb
--start_bp "A220-A253" --end_bp "A144-A159" --search_type mc
--motif_path "flex_helices,twoday,flex_helices"
```

This constrains RNAMake to find solutions that contain only a single two-way junction flanked by two nonideal helices. This path can be as long as the user desires: “flex_helices,twoday,flex_helices,twoday,flex_helices” corresponds to solutions with two distinct two-way junctions and three nonideal helices, and “flex_helices,twoday,flex_helices,twoday,flex_helices,twoday,flex_helices,twoday,flex_helices,twoday,flex_helices” has four two-way junctions and five nonideal helices. Examples of solutions to each of these constraints are highlighted in Fig. 1c.

RNAMake’s Monte Carlo search further permits users to supply their own RNA structures to be included in the motif path. These segments can be ranged from small-molecule binding aptamers to large segments of existing RNA structures. The below command illustrates how to supply a custom RNA segment into the motif path:

```
design_rna_scaffold --pdb inputs/min_tetraloop_receptor.pdb
--start_bp A220-A253 --end_bp A144-A159 --search_type mc --
motif_path "flex_helices,twoday,flex_helices,atp_aptamer,
flex_helices,twoday,flex_helices,twoday,flex_helices" --ex-
tra_pdbs inputs/atp_aptamer.pdb
```

Supplying the --extra_pdbs option with additional PDB files will permit RNAMake to incorporate user-supplied RNA structures into the motif path. In the case above, the user supplied the ATP aptamer which coupled with the motif path: “flex_helices,twoday,flex_helices,atp_aptamer,flex_helices,twoday,flex_helices,twoday,flex_helices” will be included in the fourth position in addition to three other two-way junctions and five helices. An example solution to these constraints is displayed in Fig. 1d.

RNAMake also supports the design of new RNA segments in large natural RNA systems such as the *E. coli* ribosome.

```
design_rna_scaffold --pdb inputs/ribosome.pdb --start_bp
B1449-B1454 --end_bp C1447-C1464 --only_tether_opt
```

In the command above, the input PDB contains the entire ribosome with the hairpin loop of the h44 on the 30S subunit

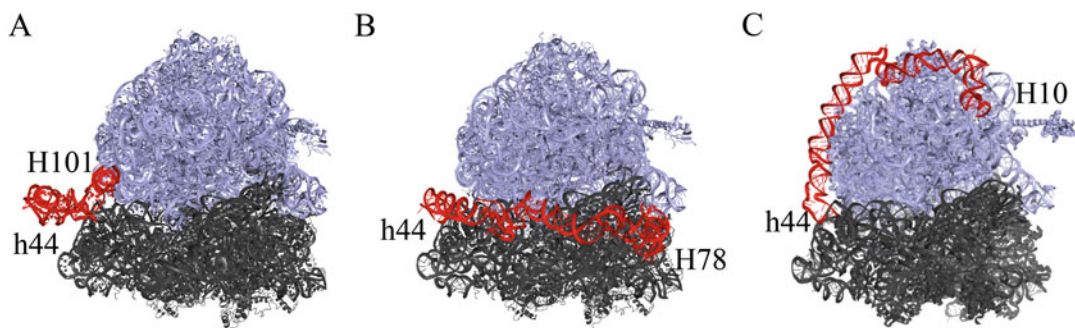


Fig. 2 RNAMake-generated ribosome tether designs. 50S subunit shown in light blue, 30S in dark gray, and RNAMake-generated tether in red. (a) Tether between helix h44 on the 30S subunit and helix H101 on the 50S subunit. (b) Tether between helix 44 and helix H78. (c) Tether between helix 44 and helix H10

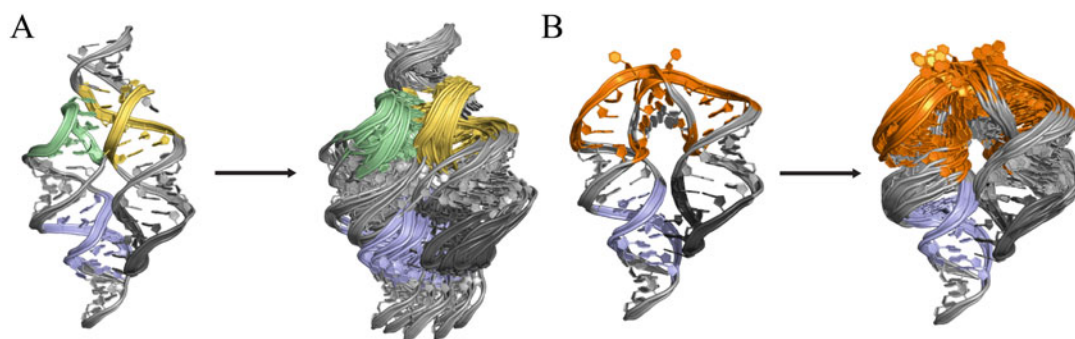


Fig. 3 3D thermodynamic conformational simulations. (a) The tectoRNA system composed of two tetraloop/tetraloop receptors. The GAAA tetraloop/receptor is in light blue and light gray, respectively. The GGAA tetraloop/receptor is in light green and yellow, respectively. RNAMake samples the 3D conformation of each base pair step finding a multitude of allowable conformations. (b) The miniTTR system, helix conformations are sampled from RNAMake's base pair step ensembles

and H101 on the 50S subunit removed. In this case, RNAMake will build an RNA segment between the two subunits creating a tethered ribosome (Fig. 2a). For large RNAs supplying the `--only_tether_opt` flag is recommended, which ignores the existing RNA structure for primary sequence optimization. RNAMake automatically builds a steric representation of all other RNA nucleotides and proteins found in the PDB file. RNAMake can build long tethers that avoid steric collisions to distal parts of the ribosome (Fig. 2b, c).

3.2 Performing 3D Thermodynamic Stability Predictions

The other major feature of RNAMake is its ability to estimate 3D thermodynamic stability through its ensemble representation of base pair steps. Originally these simulations were constrained to the tectoRNA system, a model heterodimer composed of two tetraloop/tetraloop receptor tertiary contacts (Fig. 3a)

[23, 24]. The calculations can still be performed using the `ddg_tecto` application. An example command line call is shown below:

```
ddg_tecto -fseq CUAGGAAUCUGGAAGUACCGAGGAAACUCGGUACUCCUGUGUC
CUAG -fss "(((((((.....((((((((((((((((.....))))))))))))))....)))))"
-cseq CUAGGAUAUGGAAGAUCUCCGGGAACGAGGAUCUCCUAAGUCCUAG -css
"(((((((.....((((((((((((((((.....))))))))))))))....)))))"
```

where `-fseq` and `-fss` define the sequence and secondary structure of the first strand of the tecto system and `-cseq` and `-css` define the sequence and secondary structure of the second strand.

Recently, these thermal fluctuation calculations were added to the `design_rna_scaffold` application by supplying the `--thermo_fluc` flag as shown below:

```
design_rna_scaffold --pdb inputs/min_tetraloop_receptor.pdb
--start_bp A220-A253 --end_bp A144-A159 --thermo_fluc
```

The thermal fluctuation will sample the helical base pair steps through a Monte Carlo simulation reporting the likelihood the solution will reach the defined base pair. These simulations only take into account the base pair steps of the helices; if there are significant number of non-helical motifs in a solution, these calculations will not report the true stability of the entire system.

References

1. Jasinski D, Haque F, Binzel DW, Guo P (2017) Advancement of the emerging field of RNA nanotechnology. *ACS Nano* 11:1142–1164. <https://doi.org/10.1021/acsnano.6b05737>
2. Leontis NB, Westhof E (2002) The annotation of RNA motifs. *Comp Funct Genomics* 3: 518–524. <https://doi.org/10.1002/cfg.213>
3. Petrov AI, Zirbel CL, Leontis NB (2013) Automated classification of RNA 3D motifs and the RNA 3D motif atlas. *RNA* 19: 1327–1340. <https://doi.org/10.1261/rna.039438.113>
4. Parlea LG, Sweeney BA, Hosseini-Asanjan M, Zirbel CL, Leontis NB (2016) The RNA 3D motif atlas: computational methods for extraction, organization and evaluation of RNA motifs. *Methods* 103:99–119. <https://doi.org/10.1016/j.ymeth.2016.04.025>
5. Sarver M, Zirbel CL, Stombaugh J, Mokdad A, Leontis NB (2008) FR3D: finding local and composite recurrent structural motifs in RNA 3D structures. *J Math Biol* 56:215–252. <https://doi.org/10.1007/s00285-007-0110-x>
6. Hendrix DK, Brenner SE, Holbrook SR (2005) RNA structural motifs: building blocks of a modular biomolecule. *Q Rev Biophys* 38: 221–243. <https://doi.org/10.1017/S0033583506004215>
7. Leontis NB, Westhof E (2003) Analysis of RNA motifs. *Curr Opin Struct Biol* 13: 300–308. [https://doi.org/10.1016/S0959-440X\(03\)00076-9](https://doi.org/10.1016/S0959-440X(03)00076-9)
8. Grabow WW, Jaeger L (2014) RNA self-assembly and RNA nanotechnology. *Acc Chem Res* 47:1871–1880. <https://doi.org/10.1021/ar500076k>
9. Grabow WW, Zakrevsky P, Afonin KA, Chworos A, Shapiro BA, Jaeger L (2011) Self-assembling RNA nanorings based on RNAI/II inverse kissing complexes. *Nano Lett* 11:878–887. <https://doi.org/10.1021/nl104271s>

10. Dibrov SM, McLean J, Parsons J, Hermann T (2011) Self-assembling RNA square. *Proc Natl Acad Sci U S A* 108:6405–6408. <https://doi.org/10.1073/pnas.1017999108>
11. Guo P (2010) The emerging field of RNA nanotechnology. *Nat Nanotechnol* 5: 833–842. <https://doi.org/10.1038/nnano.2010.231>
12. Khisamutdinov EF, Jasinski DL, Li H, Zhang K, Chiu W, Guo P (2016) Fabrication of RNA 3D nanoprisms for loading and protection of small RNAs and model drugs. *Adv Mater Weinheim* 28:10079–10087. <https://doi.org/10.1002/adma.201603180>
13. Geary C, Rothemund PWK, Andersen ES (2014) RNA nanostructures. A single-stranded architecture for cotranscriptional folding of RNA nanostructures. *Science* 345:799–804. <https://doi.org/10.1126/science.1253920>
14. Wimberly BT, Brodersen DE, Clemons WM, Morgan-Warren RJ, Carter AP et al (2000) Structure of the 30S ribosomal subunit. *Nature* 407:327–339. <https://doi.org/10.1038/35030006>
15. Nguyen THD, Galej WP, Bai X, Savva CG, Newman AJ, Scheres SHW et al (2015) The architecture of the spliceosomal U4/U6.U5 tri-snRNP. *Nature* 523:47–52. <https://doi.org/10.1038/nature14548>
16. Woodson SA (2005) Structure and assembly of group I introns. *Curr Opin Struct Biol* 15: 324–330. <https://doi.org/10.1016/j.sbi.2005.05.007>
17. Dibrov SM, Ding K, Brunn ND, Parker MA, Bergdahl BM, Wyles DL et al (2012) Structure of a hepatitis C virus RNA domain in complex with a translation inhibitor reveals a binding mode reminiscent of riboswitches. *Proc Natl Acad Sci U S A* 109:5223–5228. <https://doi.org/10.1073/pnas.1118699109>
18. Yesselman JD, Eiler D, Carlson ED, Gotrik MR, d’Aquino AE, Ooms AN et al (2019) Computational design of three-dimensional RNA structure and function. *Nat Nanotechnol* 14:866–873. <https://doi.org/10.1038/s41565-019-0517-8>
19. Fried SD, Schmied WH, Uttamapinant C, Chin JW (2015) Ribosome subunit stapling for orthogonal translation in *E. coli*. *Angew Chem Int Ed Engl* 54:12791–12794. <https://doi.org/10.1002/anie.201506311>
20. Orelle C, Carlson ED, Szal T, Florin T, Jewett MC, Mankin AS (2015) Protein synthesis by ribosomes with tethered subunits. *Nature* 524: 119–124. <https://doi.org/10.1038/nature14862>
21. Carlson ED (2015) Creating Ribo-T: (design, build, test)ⁿ. *ACS Synth Biol* 4:1173–1175. <https://doi.org/10.1021/acssynbio.5b00218>
22. Schmied WH, Tnimov Z, Uttamapinant C, Rae CD, Fried SD, Chin JW (2018) Controlling orthogonal ribosome subunit interactions enables evolution of new function. *Nature* 564:444–448. <https://doi.org/10.1038/s41586-018-0773-z>
23. Denny SK, Bisaria N, Yesselman JD, Das R, Herschlag D, Greenleaf WJ (2018) High-throughput investigation of diverse junction elements in RNA tertiary folding. *Cell* 174: 377–390.e20. <https://doi.org/10.1016/j.cell.2018.05.038>
24. Yesselman JD, Denny SK, Bisaria N, Herschlag D, Greenleaf WJ, Das R (2019) Sequence-dependent RNA helix conformational preferences predictably impact tertiary structure formation. *Proc Natl Acad Sci U S A* 116:16847–16855. <https://doi.org/10.1073/pnas.1901530116>



RNA 3D Structure Comparison Using RNA-Puzzles Toolkit

Marcin Magnus and Zhichao Miao

Abstract

Computational modeling of RNA three-dimensional (3D) structure may help in unrevealing the molecular mechanisms of RNA molecules and in designing molecules with novel functions. An unbiased blind assessment to benchmark the computational modeling is required to understand the achievements and bottlenecks of the prediction, while a standard structure comparison protocol is necessary. RNA-Puzzles is a community-wide effort on the assessment of blind prediction of RNA tertiary structures. And RNA-Puzzles toolkit is a computational resource derived from RNA-Puzzles, which includes (i) decoy sets generated by different RNA 3D structure prediction methods; (ii) 3D structure normalization, analysis, manipulation, and visualization tools; and (iii) 3D structure comparison metric tools. In this chapter, we illustrate a standard RNA 3D structure prediction assessment protocol using the selected tools from RNA-Puzzles toolkit: rna-tools and RNA_assessment.

Key words Structure comparison, RNA 3D structure prediction, Base pair, RNA-Puzzles

1 Introduction

Ribonucleic acid or RNA is a central type of molecule in a great variety of biological processes throughout the central dogma [1–3], including transcription regulation [4], RNA splicing [5, 6], and protein translation [7]. The functional diversity of RNA is attributed to its ability to form specific 3D structures that can carry out biological functions [8–10]. Thus, the accurate and rapid modeling of RNA 3D structure is an important step in understanding the molecular function as well as its interaction with other macromolecules and ligands. Historically, the first manual prediction of tRNA tertiary structure marked the emergence of bioinformatics [11].

During the last decade, a good number of RNA 3D structure prediction algorithms have been actively developed and improved. These prediction methods cover approaches similar to protein structure prediction, including comparative modeling (e.g., Mod-eRNA [12]), fragment assembly (e.g., RNAComposer [13], 3dRNA [14], and VfoldLA [15]), and de novo modeling (e.g.,

NAST [16], iFoldRNA [17, 18], and SimRNA [19]). To blind compare these prediction methods and understand the bottlenecks in the field, RNA-Puzzles, which is a community effort in assessing RNA 3D structure prediction, was initiated 10 years ago. To understand the performance and limitation of the existing programs, an unbiased assessment workflow is necessary, while some RNA-specific metrics have been developed and tested [20].

As one of the most widely used metrics in 3D structure comparison [21], root-mean-square deviation (RMSD) measures the root mean square deviation of the Euclidean distances of the aligned atoms after superimposition. However, it generalizes the errors over the whole structure. And the errors in linker regions may result in topological change, thus leading to high values in RMSD even if other parts of the structure are similar to the regions in the reference structure. Considering the number of degrees of freedom in RNA structure, RNA is more likely to be influenced by such local errors than protein structure. Besides, RMSD cannot take into account the base pair feature of RNA.

Consequently, some RNA-specific metrics have been conceived: Interaction network fidelity (INF) was designed to evaluate the accuracy of interaction prediction, while deformation profile [20] was introduced to complement single value evaluation metrics, and P -value [22] was used to balance the effect of sequence length in RMSD. The INF, defined as the Matthews correlation coefficient (MCC) between the interactions of the reference structure and that of the predicted structure, indicates the consistency between the prediction and the reference structure in terms of interactions. And the evaluated interaction types have been categorized into Watson-Crick interaction, *non*-Watson-Crick interaction, and base stacking. Deformation profile (DP) uses a 2D distance matrix, which can be visualized by a heatmap, to represent the average distance between a prediction and the reference structure. The color in the heatmap correlates to the local similarity between the structures. There are also other novel metrics being developed in recent years, including the mean of circular quantities (MCQ) [23] and the Longest Continuous Segments in Torsion Angle space (LCS-TA) [24], which measure the structural similarity in the torsion angle space. These RNA structure comparison metrics together with RNA structure decoy sets and RNA structure normalization, analysis, manipulation, and visualization tools are available in RNA-Puzzles toolkit as a computational resource [25].

In this chapter, we use some tools in the RNA-Puzzles toolkit to show a standard RNA 3D structure comparison protocol. This workflow mimics the RNA 3D structure prediction scenario in RNA-Puzzles: Both predicted RNA structures (as the predicted structures) and experimentally determined reference RNA structures (as the reference structures) are available in the comparison.

We show steps in (1) reformatting RNA structures, (2) calculating the comparison metrics, and (3) visualizing the results.

2 Materials

Please note that the programs described in this chapter are run in the standard Python 3 environment.

2.1 Input Data

In an RNA 3D structure prediction evaluation, we have a list of predicted structures and at least one reference structure available. If more than one reference structure of the same sequence has been experimentally determined or multiple chains of the same sequence exist in an asymmetric biological unit, all of these structures of the same sequence are used as reference structures. And the one with the lowest RMSD to a given predicted structure is deemed as the reference structure for that prediction. The Brookhaven PDB [26] format is used for input protein files. However, the predicted structures are generated from different RNA structure prediction programs, some of which are output from molecular dynamics optimization, thus demonstrating a good diversity in the format. In this chapter, we will use structures submitted to RNA-Puzzles as examples. All these examples can be found at GitHub: https://github.com/RNA-Puzzles/standardized_dataset.

2.2 Programs in RNA-Puzzles Toolkit

The computational tools in RNA-Puzzles toolkit include rna-tools, RNA-assessment, and other related programs. In this chapter, we focus on the usage of rna-tools and RNA-assessment.

rna-tools is a set of RNA structure tools, which constitute the main tools of RNA-Puzzles toolkit. It provides a unique interface for (i) RNA sequence alignment and database search, (ii) RNA 3D structure prediction and refinement, (iii) RNA structure analysis and clustering, and (iv) RNA structure formatting, manipulation, and visualization. rna-tools provides a Python Package Index (pip) installation:

```
$ pip install rna-tools
```

Advanced installation and configuration can be found at <https://rna-tools.readthedocs.io/en/latest/install.html>.

RNA_normalizer and deformation profile are programs used for RNA structure prediction evaluation in RNA-Puzzles. RNA_normalizer can be used to standardize the structure file format as well as to measure the assessment metrics, including RMSD, INF, deformation index, and *P*-value. Deformation profile is provided as an independent package to measure the deformation profile and create the heatmap results. RNA_normalizer can be installed through the GitHub repository:

```
$ pip install -e \
git+https://github.com/RNA-Puzzles/RNA_assessment.git#egg=r-
na-normalizer
```

Deformation profile does not require an installation but needs to be downloaded from GitHub:

```
$ git clone https://github.com/RNA-Puzzles/Deformation_Pro-
file.git
```

The dependencies, including Biopython [27] and numpy [28], need to be installed.

2.3 Website

All the datasets and computational tools in RNA-Puzzles toolkit are available at GitHub: <https://github.com/RNA-Puzzles/RNA-Puzzles-toolkit-overview>.

3 Methods

In this section, we illustrate the structure prediction evaluation workflow in RNA-Puzzles, which includes three steps when comparing the predicted structures with the reference structure: (i) Normalize the structure format; (ii) calculate the structure comparison metrics; and (iii) visualize the results. Here, we use cases in RNA-Puzzles as showcases.

3.1 Standardize the Structure Format

Example datasets, which contain all the raw submitted prediction and standardized structures from RNA-Puzzles, can be downloaded from RNA-Puzzles toolkit GitHub https://github.com/RNA-Puzzles/standardized_dataset

rna-tools provides a command for the download:

```
$ rna_pdb_toolsx.py --fetch rp
```

Git can also be used for download:

```
$ git clone https://github.com/RNA-Puzzles/standardized_data-
set.git
```

However, the raw submitted predictions, as well as the experimentally determined reference structures, may use different nomenclature for the atoms and residues, while a variety of additional ligands and atoms, e.g., metal ions, water, sulfates, and crystallization specific heteroatoms, are often included in these structures. Therefore, both the predicted and reference structures need to be standardized before comparison.

3.1.1 Nomenclature Standardization

To align the atoms in the predicted structure to the ones in the reference structure, the atom and residue names need to be the same in both structures. Thus, the structure files are standardized before measuring the comparison metrics. Rna-tools provides an interface to standardize all the atom and residue names:

```
$ rna_pdb_toolsx.py --get-rnapuzzle-ready --inplace --suffix
rpr *.pdb
```

where “--get-rnapuzzle-ready” specifies the operation, “--inplace” indicates that the standardized results are saved in new files with an “rpr” suffix, and “*.pdb” indicates that all PDB files in the directory should be processed.

RNA_normalizer also provides an alternative to standardize the structure files. The “parse” function in the program parses the structure file and standardizes the given atom name and residue name directories. The operation can be performed in a Jupyter notebook (shown in the example of RNA_normalizer:

https://github.com/RNA-Puzzles/RNA_assessment/blob/master/example.ipynb)

The python functions are as follows:

```
handle = RNA_normalizer.PDBNormalizer( RESIDUES_LIST, ATOMS_
LIST )
ok = handle.parse( in_file, out_file )
```

where “RESIDUES_LIST” and “ATOMS_LIST” are the dictionary files for the residue name and atom name. “in_file” and “out_file” are the structure files before and after the standardization.

However, these automated procedures may not deal with missing atoms, sequence mismatch between the predicted and reference structures, and the chain numbering problem. In RNA-Puzzles, RNA_normalizer skips the mismatches in the sequences and the missing atoms. The RNA_normalizer example notebook demonstrates a full example to perform the structure standardization and to measure the structure comparison metrics. To complement RNA_normalizer and fix the RNA structures, rna-tools provides a set of more rigorous tools, which are elaborated below.

3.1.2 Missing Atoms

The “--get-rnapuzzle-ready” option in rna-tools provides a fix for the missing atoms and is able to process all the known cases in RNA-Puzzles. If many atoms are missing, pdbfixer (<https://github.com/openmm/pdbfixer>) is recommended to fix the missing atoms. For the rna-tools fixed structures, the added missing atoms are listed at the top of the output file as “REMARK,” for example:

```

$ rna_pdb_toolsx.py --get-rnapuzzle-ready 17_solution_0.pdb
REMARK 250 Model edited with rna-tools
REMARK 250 ver 3.4.8+2.g508b05e.dirty
REMARK 250 https://github.com/mmagnum/rna-tools
REMARK 250 Mon Jun 29 11:21:27 2020
REMARK 250 Fixed atoms/residues:
REMARK 250 - add O2' in chain: B <Residue G het= resseq=53
icode= > residue # 6
ATOM    1  P    C A  1    -29.720 -19.750   3.190  1.00183.16
P
ATOM    2  OP1  C A  1    -28.490 -20.280   2.531  1.00180.35
O
ATOM    3  OP2  C A  1    -29.638 -18.728   4.275  1.00185.39
O

```

The output shows that an O2' atom is missing in the reference (solution) structure for RNA-Puzzle PZ17 (PDB ID: 5K7C). And a user needs to decide if the added missing atoms should be considered when calculating the all-atom RMSD.

Further, *rna-tools* provides a novel tool, *diffpdb.py*, to investigate the difference between two structure files. It facilitates the identification of the missing and mismatching atoms/residues by running the command line (in “rp17/misc” folder):

```

$ diffpdb.py --names 17_0_solution_5K7C_MissAtom_rpr.pdb
17_Bujnicki_1_rpr_del.pdb

```

The results from *diffpdb.py* can be visualized by *DiffMerge* (<https://sourcegear.com/diffmerge/>), which highlights the differences between the files (Fig. 1).

3.1.3 Missing Fragments

In some structures, fragments can be missing. For example, some dynamic regions of the experimental structures are not determined. In such a situation, it is suggested to skip these regions in the comparison. We provide two options from *RNA_normalizer* and from *rna-tools*, respectively. *RNA_normalizer* specifies the structure regions to be compared by giving a “.index” file when loading the structure in python:

```

handle = RNA_normalizer.PDBStruct()
handle.load( struct_file, index_file )
seq = handle.raw_sequence()

```

where *struct_file* is the input structure file and the *index_file* is the “.index” file to specify the region. For example, “A:1:45, B:57:15” means the structure includes two fragments. The first fragment starts from residue 1 of chain A and the length is 45 nucleotides, while the second starts from residue 57 of chain B and the length is

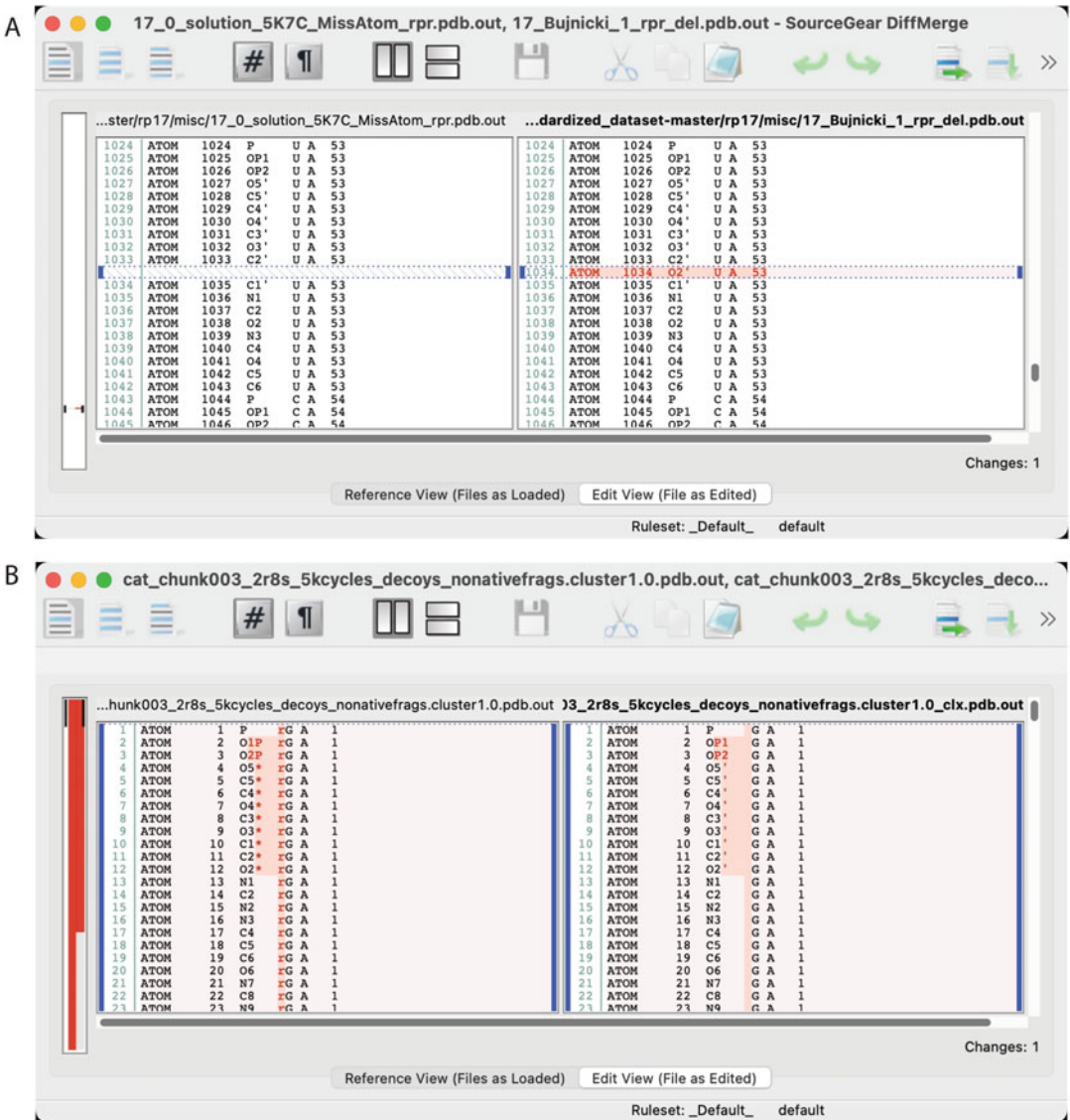


Fig. 1 Sample visualization of structure file comparison. Diffpdb.py (a) identifies one atom difference and (b) format variation between two structure files

15 nucleotides. And the “.raw_sequence()” function returns the sequence of the specified region.

Alternatively, if the aligned region is unknown, rna-tools can be used to identify the aligned regions and delete the missing fragments:

```
$ rna_pdb_toolsx.py --get-seq --fasta *.pdb
>13_solution_4XW7_rpr.pdb A:1-45 57-71
GGGUCGUGACUGGCGAACAGGUGGGAACCACCGGGAGCGACCCGCCGCCUGGGC
```

```
>13_Adamiak_1_rpr.pdb A:1-71
GGGUCGUGACUGGCGAACAGGUGGGAAACCACCGGGGAGCGACCCCGGCAUC-
GAUAGCCGCCCGCCUGGGC
(..)
```

According to the difference in the sequences, the fragment of nucleotides 46–56 is missing in the reference structure (PDB 4XW7). But the predicted structures cover the full target sequence (Fig. 2). Therefore, only the regions which can be aligned to the reference structure (A:1–45 and A:57–71) should be considered in structure comparison.

rna-tools can be used to delete the unaligned regions (A:46–56 in the example):

```
$ rna_pdb_toolsx.py --delete A:46-56 --inplace *_rpr.pdb
```

where “--delete” specifies the operation, “--inplace” indicates to save the output in new files, and “*_rpr.pdb” suggests to process all the files ending with “_rpr.pdb” in the directory.

To check if the results have the same sequence, the “--get-seq” operation can be used again:

```
$ rna_pdb_toolsx.py --get-seq --fasta *.pdb
>13_0_solution_4XW7_rpr.pdb A:1-45 57-71
GGGUCGUGACUGGCGAACAGGUGGGAAACCACCGGGGAGCGACCCGCCCGCCUGGGC
>13_Adamiak_1_rpr_del46-56.pdb A:1-45 57-71
GGGUCGUGACUGGCGAACAGGUGGGAAACCACCGGGGAGCGACCCGCCCGCCUGGGC
```



Fig. 2 Reference and predicted structures. The reference structure (“13_0_solution_4XW7_rpr”), shown as green, is missing the fragment of nucleotides 46 to 56. The predicted structure (“13_Das_7_1_rpr”) is aligned to the reference structure, while the missing fragment is shown as orange in the predicted structure

3.1.4 Sequence Mismatches

In some cases, the sequence of the predicted structures can be slightly different from the sequence in the reference structure. Similar to the missing atoms, these mismatched nucleotides can be either omitted or rebuilt in the comparison. To omit the mismatched nucleotides, the same approaches (use `RNA_normalizer` to specify the aligned region or use `rna-tools` to delete the unaligned region) as dealing with the missing fragments can be used. `Rna-tools` also provides an interface to `ModeRNA` [12] to rebuild the mismatched nucleotides in the predicted structures to match the sequence of the reference structure. Here, we use `PZ3` in `RNA-Puzzles` as an example; the pairwise sequence alignment is as follows:

```
> 3_solution_0_rpr A:1-84
CUCUGGAGAGAACCGUUUAAUCGGUCGCCGAAGGAGCAAGCUCUGCGGAAACGCAGAGU-
GAAACUCUCAGGCAAAGGACAGAG
|||||||||||||||||||||||||||||||||||||||||.|.|||||||
||||||||||||||||||||
CUCUGGAGAGAACCGUUUAAUCGGUCGCCGAAGGAGCAAGCUCUGCGCUAUGCAGAGU-
GAAACUCUCAGGCAAAGGACAGAG
> target sequence (3_bujnicki_1_rpr A:1-84)
```

Nucleotides 48, 50, and 52 in the predicted structures need to be mutated to G, A, and C:

```
$ rna_pdb_toolsx.py --mutate 'A:48G+50A+52C' \
    --inplace --suffix mut \
    --ignore-files 3_solution \
    *.pdb
```

In the command, “`--mutate`” specifies the operation to mutate the three nucleotides [48, 50, 52] in chain “A” to G, A, and C, respectively. “`--inplace --suffix mut`” indicates to save changes to new files and a suffix “`mut`” will be added, while “`*.pdb`” specifies to process all PDB files in the directory. However, “`3_solution_0_rpr.pdb`” is the reference structure and should not be processed. Therefore, the “`--ignore-files 3_solution`” option skips the processing of all files that start with “`3_solution.`”

Again, the “`--get-seq`” option can be used to check if the mutations have been correctly rebuilt (the result is shown in Fig. 3a):

```
$ rna_pdb_toolsx.py --get-seq \
    --oneline 3_bujnicki_1_rpr* \
    --color-seq --compact
```

```

A (py37) [mx] rp03$ git:(master) * rna_pdb_toolsx.py --get-seq --oneline 3_bujnicki_1_rpr* --color-seq --compact
UCUCGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCUCUGGCGGUUGCGGGGUGGCUUCGGGGCGGGGGGGG # A:1-84 # 3_bujnicki_1_rpr
UCUCGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCUCUGGCGGUUGCGGGGUGGCUUCGGGGCGGGGGGGG # A:1-84 # 3_bujnicki_1_rpr_mut

B GCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_RNAComposer_Human_2
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_RNAComposer_Human_3
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_RNAComposer_Human_4
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_RNAComposer_Human_5
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_1
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_10
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_2
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_3
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_4
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_5
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_6
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_7
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_8
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_RW3D_9
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_simRNA_1
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_simRNA_2
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-40 GCGGGGGGGGGGGGGGGGGGGGCUUCUUGGC # B:1-22 # 19_simRNA_3
GGCGGGGGGGGGCCCGUUUUUGGGGUCGCGCCGGGGGGGGGGCCCGCC # A:1-62 # 19_solution_0
    
```

Fig. 3 Example of RNA sequence comparison by rna-tools. The “--get-seq” operation in rna-tools can be used to (a) compare the mutated sequences to the wild-type sequence and (b) show the sequences and the chain names of all structure files in a folder (e.g., the “rp19/raw” data folder)

3.1.5 Chain Identifiers

Some predicted structures may have different chain names than others, or multiple chains can be ordered in a different way. Dealing with such cases, RNA_normalizer uses the index file to specify the aligned region, and the nucleotides are ordered according to the order specified in the index file. Thus, the chain names are not affecting the comparison. Rna-tools can rename the chains to standardize the files. We use PZ19 in RNA-Puzzles, for example, and start by checking the sequences using the “--get-seq” operation of rna-tools (results are shown in Fig. 3b):

```

$ rna_pdb_toolsx.py --get-seq --oneline --color-seq --compact
*.pdb
    
```

The predicted structures named the chains in two different ways: as one chain (A:1–62) and as two chains (A:1–40, B:1–22). rna-tools can reformat the one chain files into two chains, by changing the nucleotides A:41–62 to B:1–22:

```

$ rna_pdb_toolsx.py --edit 'A:41-62>B:1-22' --inplace {*RW3D*,
*solution*}.pdb
    
```

where “--edit” is the operation to use “A:41–62 > B:1–22” specifies that nucleotides 41–62 of chain A are renamed to nucleotides 1–22 of chain B. And “{*RW3D*,*solution*}.pdb” indicates that only PDB files with the phrases “RW3D” and “solution” are processed, since these structure are named as one chain.

Then, “--get-seq” operation is used to validate the correction:

```
$ rna_pdb_toolsx.py --get-seq --oneline --color-seq \
                        --compact {*RW3D*.pdb,*solution*.pdb}
GGCGGGGGGGCGGGGCCCGUCCCGUGGCGGCCGGGGGGCCGGCCCC # A:1-40 GGGG
GGGGCGGCGGGGCGGCUCUCCUGGC # B:1-22 # 19_RW3D_1
(...)
GGCGGGGGGGCGGGGCCCGUCCCGUGGCGGCCGGGGGGCCGGCCCC # A:1-40
GGGGGGGGCGGCGGGGCGGCUCUCCUGGC # B:1-22 # 19_solution_0
```

3.2 Calculate Structure Comparison Metrics

After structure standardization, the predicted structures and the reference structure(s) are compared by a number of metrics. Here, we introduce the calculation of RMSD, *P*-value, INF, and DP using RNA-Puzzles toolkit.

RNA_normalizer includes a class “PDBComparer” to measure all these metrics. And the python function below returns all the comparison metrics:

```
def calc_all_metrics(ref_file, ref_index, pred_file, pred_in-
dex):
    ref_struct = RNA_normalizer.PDBStruct()
    ref_struct.load( ref_file, ref_index )
    ref_seq = ref_struct.raw_sequence()
    pred_struct = RNA_normalizer.PDBStruct()
    pred_struct.load( pred_file, pred_index )
    pred_seq = pred_struct.raw_sequence()
    comparer = RNA_normalizer.PDBComparer()
    rmsd = comparer.rmsd( pred_struct, ref_struct )
    pvalue = comparer.pvalue( rmsd, len(pred_seq), "-" )
    INF_ALL = comparer.INF( pred_struct, ref_struct,
type="ALL" )
    DI_ALL = rmsd / INF_ALL
    INF_WC = comparer.INF( pred_struct, ref_struct,
type="PAIR_2D" )
    INF_NWC = comparer.INF( pred_struct, ref_struct,
type="PAIR_3D" )
    INF_STACK = comparer.INF( pred_struct, ref_struct,
type="STACK" )
    return (rmsd, pvalue, DI_ALL, INF_ALL, INF_WC, INF_NWC,
INF_STACK)
```

The `ref_file` and `ref_index` are the structure file and index file for the reference structure, while `pred_file` and `pred_index` are the structure file and index file for the predicted structure, respectively. This function returns metrics of all-atom RMSD, *P*-value, deformation index for the whole structure, INF for all interactions, as well as INF for Watson-Crick interactions, for *non*-Watson-Crick

interactions, and for base stackings. For the interactions, RNA-normalizer uses MC-annotate [29] to extract all the different types of interactions. Details can be found from the GitHub example of RNA_normalizer. Besides RNA_normalizer, we also show the application of rna-tools as below.

3.2.1 Root-Mean-Square Deviation (RMSD)

The “rna_calc_rmsd.py” tool in rna-tools measures the RMSD for multiple predicted structures against one reference structure:

```
$ rna_calc_rmsd.py --target-fn 19_0_solution_5T5A_rpr.pdb \
                    --sort-results \
                    --print-results \
                    *.pdb

# method: all-atom-built-in
# of models: 55
# of atoms used: 1325

          fn  rmsd_all
0      19_0_solution_5T5A_rpr.pdb  0.00
11     19_Chen_Human_1_rpr.pdb    5.53
12     19_Chen_Human_2_rpr.pdb    7.41
34     19_RNAComposer_3_rpr.pdb   9.64
41    19_RNAComposer_Human_5_rpr.pdb 9.86
14     19_Chen_Human_4_rpr.pdb    9.92
17     19_Das_Human_2_rpr.pdb   10.15
7      19_Bujnicki_Human_2_rpr.pdb 10.23
33     19_RNAComposer_2_rpr.pdb  10.82
20     19_Das_Human_5_rpr.pdb   10.89
(...)
25     19_Ding_Human_5_rpr.pdb   20.13
15     19_Chen_Human_5_rpr.pdb   22.72

csv was created!  rmsds.csv
```

where “--target-fn” points to the reference structure file and “--sort-results” specifies that the output RMSD result is sorted. By default, the result is saved in a CSV file, and the “--print-results” option also prints the result on the screen. “*.pdb” indicates that all the PDB files in the directory are processed.

To specify the fragments to be compared, the “--target-selection” and “--model-selection” options can be used, while the “--model-ignore-selection” skips unwanted atoms or residues in the predicted structure. We use PZ17 in RNA-Puzzles as an example:

```
$ rna_calc_rmsd.py -t 17_0_solution_5K7C_MissAtomResi53_rpr.pdb \
                    --target-selection A:1-47+52-62 \
                    --model-selection A:1-47+52-62 \
                    --model-ignore-selection A/57/02\'
                    --print-results --sort-results \
```



```

--ignore-files 'solution' \
*.pdb
# method: all-atom-built-in
# of models: 107
# of atoms used: 1237

                fn  rmsd_all
88      17_SimRNAAS2_1_rpr.pdb    5.18
29      17_Das_4_rpr.pdb        7.17
94      17_SimRNAAS2_7_rpr.pdb   7.67
26      17_Das_1_rpr.pdb        8.66
33      17_Das_8_rpr.pdb        8.71
34      17_Das_9_rpr.pdb        8.80
17      17_Chen_5_rpr.pdb        9.46
(...)

```

This command uses fragments 1–45 and 52–62 of chain A in both reference structure (target) and predicted structure (model) for comparison but ignores the O2' atom of nucleotide 57 of chain A in the predicted structure.

In some cases, more than one reference structures are available. Rna-tools provides the `rna_calc_rmsd_multi_targets.py` tool to deal with this situation:

```

$ rna_calc_rmsd_multi_targets.py --models *.pdb \
                                --targets solutions/*.pdb \
                                --target-selection A:1-27+29-41
\
                                --model-selection A:1-27+29-41

```

An example result is shown in Table 1.

3.2.2 Interaction Network Fidelity (INF) and Deformation Index (DI)

INF better evaluates the accuracy of interaction than RMSD, while the interaction types can be categorized into Watson-Crick interactions, *non*-Watson-Crick interactions, and base stackings. And DI measures the ratio between RMSD and INF. The `rna_calc_inf.py` tool in rna-tools, which measures all the INF scores based on contacts detected with ClARNA [30], can be used in a similar manner as `rna_calc_rmsd.py`:

```

$ rna_calc_inf.py --target-fn 17_0_solution_5K7C_rpr.pdb \
                  --target-selection A:1-47+52-62 \
                  --model-selection A:1-47+52-62 \
                  --print-results --sort-results
                  *.pdb

                target
fn  inf_all  inf_stack  inf_WC  inf_nWC  sns_WC  ppv_WC
sns_nWC  ppv_nWC

```

Table 1
Example results from the “rna_calc_rmsd_multi_targets.py.” Each row shows the comparison between the predicted structures and different reference structures

Structure	21_solution_0_ChainA.pdb	21_solution_0_ChainB.pdb	21_solution_1_ChainA.pdb	21_solution_1_ChainB.pdb	21_solution_2.pdb	mean	median	min	max	sd	variance
21_3dRNA_1_rpr.pdb	12.17	12.11	12.17	12.11	12.11	12.1	12.11	12.11	12.17	0.03	0.00108
21_3dRNA_2_rpr.pdb	12.48	12.43	12.48	12.43	12.4	12.4	12.43	12.4	12.48	0.04	0.00123
21_3dRNA_3_rpr.pdb	12.04	12.05	12.04	12.05	12.13	12.1	12.05	12.04	12.13	0.04	0.00147
21_3dRNA_4_rpr.pdb	18.01	17.96	18.01	17.96	18.06	18	18.01	17.96	18.06	0.04	0.00175
21_3dRNA_5_rpr.pdb	17.61	17.49	17.61	17.49	17.57	17.6	17.57	17.49	17.61	0.06	0.00368
21_Adamiak_1_rpr.pdb	4.64	4.61	4.64	4.61	4.64	4.63	4.64	4.61	4.64	0.02	0.00027
21_Adamiak_2_rpr.pdb	4.65	4.62	4.65	4.62	4.64	4.64	4.64	4.62	4.65	0.02	0.00023
21_Adamiak_3_rpr.pdb	4.61	4.58	4.61	4.58	4.58	4.59	4.58	4.58	4.61	0.02	0.00027
21_Adamiak_4_rpr.pdb	4.58	4.51	4.58	4.51	4.54	4.54	4.54	4.51	4.58	0.04	0.00123
21_Adamiak_5_rpr.pdb	5.5	5.56	5.5	5.56	5.56	5.54	5.56	5.5	5.56	0.03	0.00108
21_ChenHighLig2_1_rpr.pdb	4.01	3.97	4.01	3.97	4.07	4.01	4.01	3.97	4.07	0.04	0.00168
21_ChenHighLig2_2_rpr.pdb	3.79	3.75	3.79	3.75	3.71	3.76	3.75	3.71	3.79	0.03	0.00112
21_ChenHighLig2_3_rpr.pdb	5.08	4.95	5.08	4.95	5.01	5.01	5.01	4.95	5.08	0.07	0.00423
21_ChenHighLig2_4_rpr.pdb	4.59	4.59	4.59	4.59	4.65	4.6	4.59	4.59	4.65	0.03	0.00072
21_ChenHighLig2_5_rpr.pdb	7	6.97	7	6.97	6.96	6.98	6.97	6.96	7	0.02	0.00035

```

2  17_0_solution_5K7C_rpr_sel.pdb  17_0_solution_5K7C_MissA-
tomResi53_rpr_sel.pdb      1.00      1.00      1.00      1.00      1.00
1.00      1.00      1.00
5  17_0_solution_5K7C_rpr_sel.pdb                                17_0_so-
lution_5K7C_rpr_sel.pdb      1.00            1.00      1.00      1.00
1.00      1.00      1.00      1.00
2 7      17_0_solution_5K7C_rpr_sel.pdb
17_Das_1_rpr_sel.pdb      0.81            0.83      0.92      0.41
0.94      0.90      0.33      0.50
2 3      17_0_solution_5K7C_rpr_sel.pdb
17_Chen_2_rpr_sel.pdb      0.77            0.79      0.84      0.38
0.89      0.80      0.22      0.67
    
```

The results are saved in the “inf.csv” file, while “--print-results” option allows the print on screen. The fragment selection (--target-selection and --model-selection) and skipping (--model-ignore-selection) options are the same as used in rna_calc_rmsd.py.

3.2.3 Deformation Profile (DP)

DP measures the structure similarity as a 2D distance matrix representing the average distance between a prediction and the reference structure (Fig. 4a). Deformation profile is implemented as a program in RNA-Puzzles toolkit independent of RNA_normalizer or rna-tools.

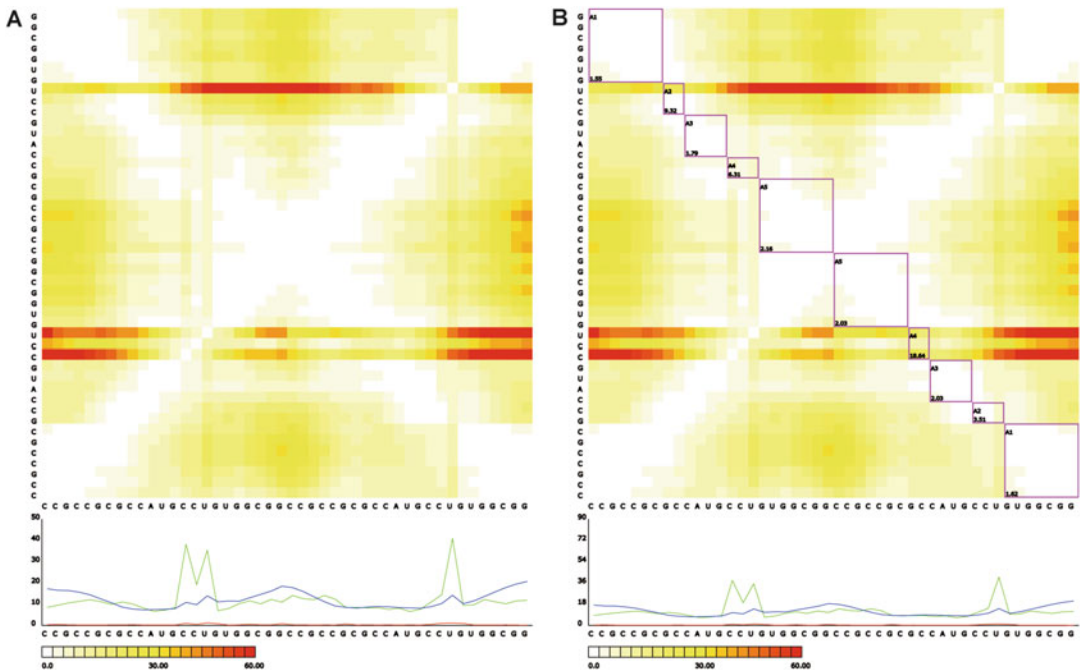


Fig. 4 Comparison between Bujnicki model 1 in PZ1 of RNA-Puzzles and the reference structure. (a) The default output from deformation profile. (b) The output with “-c” option, which specifies the helical regions in the structure

DP (dp.py) can be applied to the standardized predicted and reference structures:

```
$ python dp.py path_to_file/ref.pdb path_to_file/pred.pdb
```

“ref.pdb” and “pred.pdb” are the structure files to be compared. By default, the DP produces two output files: an SVG image, which is a heatmap representation of the DP Matrix, and a “.dat” file that contains the values of the DP matrix. The -c option in DP allows specifying all the parameters in a configuration file and plotting the secondary structure in the output heatmap. We show an example of the configuration file as follows (PZ3 in RNA-Puzzles):

```
ref_model = [("ref.pdb", 0)]
cmp_model = [("pred.pdb", 0)]
helices = [("H1", 0, 5, 73, 5),
           ("H2", 9, 6, 21, 5),
           ("H3", 26, 3, 63, 3),
           ("H4", 32, 4, 56, 5),
           ("H5", 39, 7, 46, 7)]
loops = [("L1", 5, 4), ("L2", 15, 6), ("L3", 36, 3), ("L4",
53, 4), ("L5", 66, 7)]
draw = ["H1", "H2", "H3", "H4", "H5", "L1", "L2", "L3", "L4",
"L5", "H1xH2:H1-H2", "H1xH3:H1-H3"]
```

The “ref_model” and “cmp_model” specify the input PDB files, while the number “0” following the “ref.pdb” and “pred.pdb” indicates that the first model in the PDB file is used. To use the second model, the number needs to be changed to “1.” The “helices” parameter labels a list of the helical regions in the structure, while each helix is marked as a label name followed by four numbers. The first and third numbers are the start positions for the first and the second chain of the helix, and the second and fourth numbers are the length of the helix in the first and the second strands. The “loops” parameter marks the loop regions as a list. Each element in the list is a loop region, which is annotated as a label name, the start position of the loop and the length of the loop. The “draw” parameter assigns the regions and labels to be annotated on the output heatmap.

To run the DP with this configuration file, we can use the following command:

```
$ python dp.py -c ex1.cfg
```

where ex1.cfg is the name of the configuration file. An example result is shown in Fig. 4b.

3.2.4 Other Metrics

The clash score, which indicates the geometrical feasibility, can be calculated by Molprobrity [31]. There are two steps in the processing:

- (i) Reformat the files using reduce-build:

```
$ reduce-build dir-to-standardized-files output-dir
```

- (ii) Analyze the structures with online-analysis:

```
$ oneline-analysis -nocbeta -norota -norama output-dir > out-  
putfile.txt
```

The reformatted files are saved at the directory “output-dir,” and the Molprobrity analyzed results are saved in the outputfile.txt file.

For the usage of other evaluation metrics, such as MCQ and LCS-TA, please refer to the tools at GitHub (<https://github.com/RNA-Puzzles>).

3.3 Visualize the Comparison Results

3.3.1 PyMOL Visualization

We suggest to use PyMOL for structure visualization, while the “align_all” function from the PyMOLRNA plugin of rna-tools enables structural alignment of all objects to the first object in a PyMOL session:

```
$ PyMOL> align_all
```

In PyMOL, the models can be moved up and down using the right mouse button. The function also returns RMSD values against the first object in the PyMOL command shell (Fig. 5).

3.3.2 Structure Clustering

When no reference structure is available, clustering the predicted structures may facilitate the identification of consensus in the prediction. The Clanstix tool in rna-tools converts the all-against-all RMSD distance matrix into a format readable by CLANS [32], which is a program to visualize the pairwise similarity between structures using the Fruchterman-Reingold graph layout algorithm. Using PZ17 in RNA-Puzzles as an example, we first calculate the all-vs-all RMSD distance matrix:

```
$ rna_calc_rmsd_all_vs_all.py -i struc -o rp17_rmsd.csv
```

Then, Clanstix (rna_clanstix.py) converts the CSV result file into a CLANS readable format:

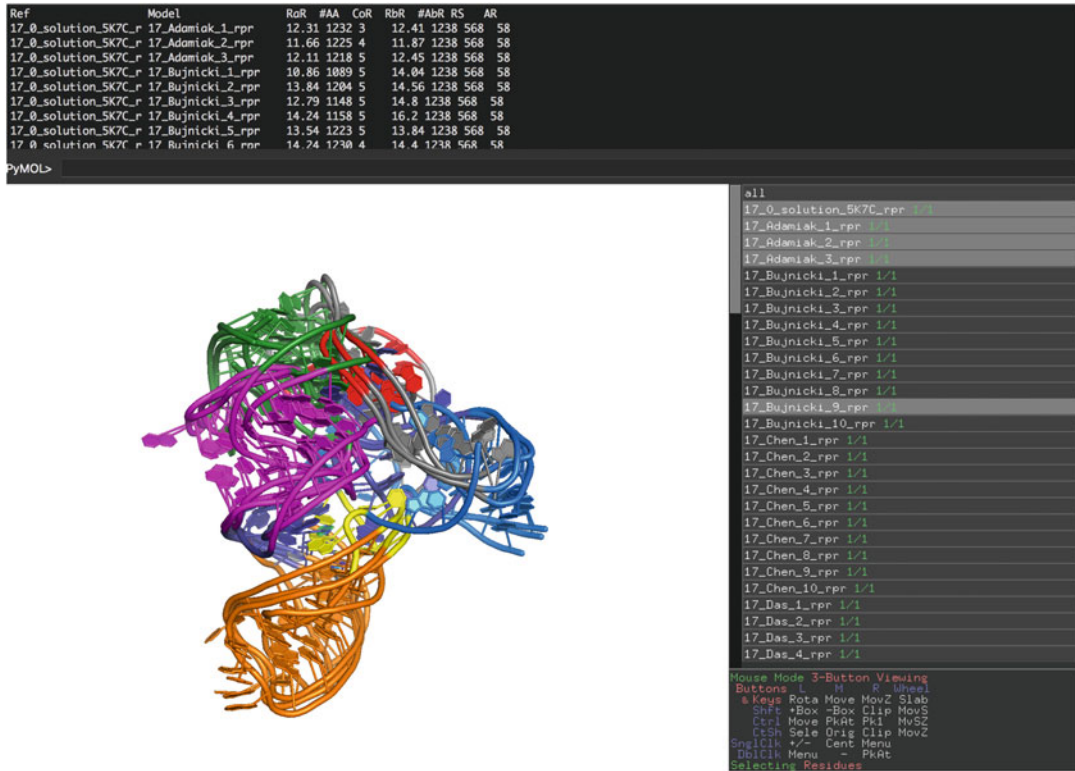


Fig. 5 An example of the “align_all” tool of PyMOL4RNA plugin. All predicted structures of PZ17 in RNA-Puzzles are superimposed to the reference structure (17_0_solution_5K7C)

```

$ rna_clanstix.py --groups 1:native+3:Adamiak+10:Bujnicki+10:
Chen+3:DasExp+10:Das+10:Ding+3:Dohkolyan+10:Major+10:
RNAComposerAS1+10:RNAComposerAS2+9:SimRNAwebAS1+9:
SimRNAwebAS2+10:Xiao \
rp17_rmsd.csv
    
```

where “--groups” specifies the number of models that belong to the given group (“1:native” means that the first model in the input file is the reference structure and should be labeled in the same way in the CLANS session, as shown in Fig. 6). In the CLANS visualization, there are three main structural clusters: (i) a cluster in the center, which includes the reference structure, Fig. 6a; (ii) a cluster of automated prediction methods – the pseudoknot was not recovered in these structures (the pseudoknot region is shown as violet in the figure), Fig. 6c; and (iii) a cluster of a different topology, Fig. 6b.

3.3.3 Contacts Analysis

The ClaRNA tool in rna-tools enables the analysis of base pair contacts in PyMOL. Using PZ17 in RNA-Puzzles (rp17/del) as an example, we first open PyMOL, load models, and select the

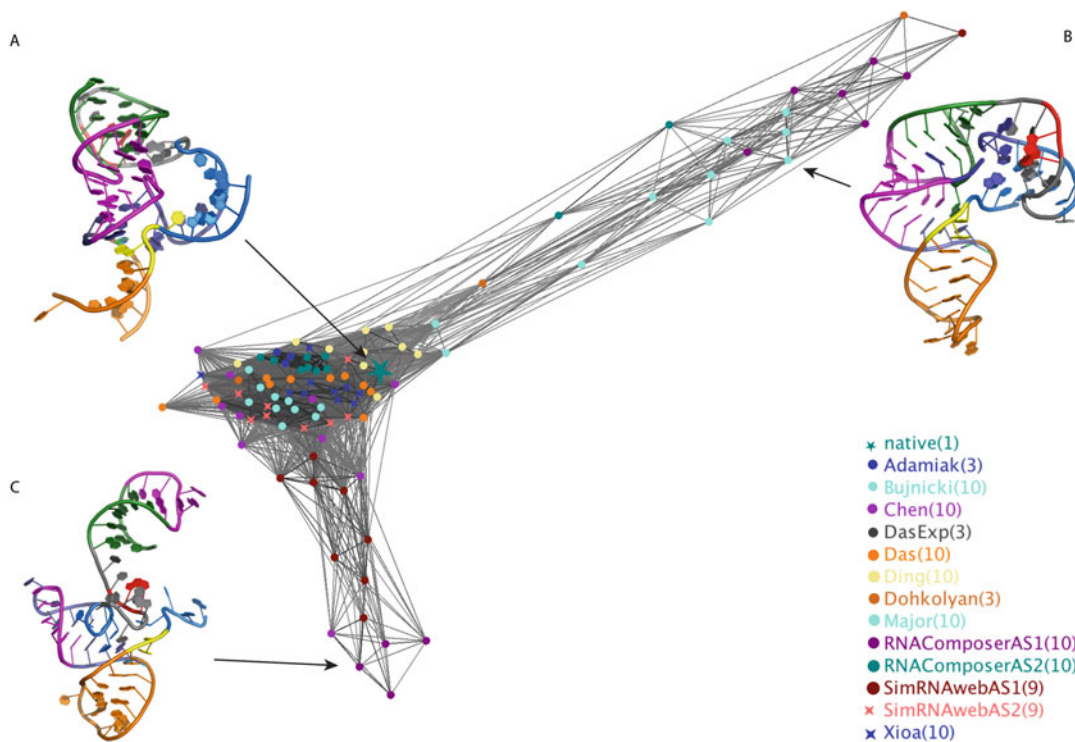


Fig. 6 The structure clustering result of PZ17 in RNA-Puzzles using the Clanstix/CLANS

nucleotides involved in the catalysis of the pistol ribozyme: nucleotides 29, 42, 57, and 58 of PDB 5K7C. Then, we run the “clarna” command in PyMOL command shell:

```
$ PyMOL> clarna
17_0_solution_5K7C_MissAtomResi53_rpr
-----
chains: A 29 58
A 29 A 58 bp U U WW_cis 0.8238
A 42 A 57 bp G G HS_tran 0.7572
17_Bujnicki_1_rpr
-----
chains: A 29 58
A 57 A 58 bp G U >> 0.9879
```

According to the results, the reference structure (PDB ID: 5K7C) includes two base pair interactions: a *cis*-Watson-Watson interaction between nucleotides 29 and 58 and a *trans*-Hoogsteen-Sugar interaction between nucleotides 42 and 57. But the predicted structure (Bujnicki model 1) only predicts a stacking between 57 and 58, as shown in Fig. 7.

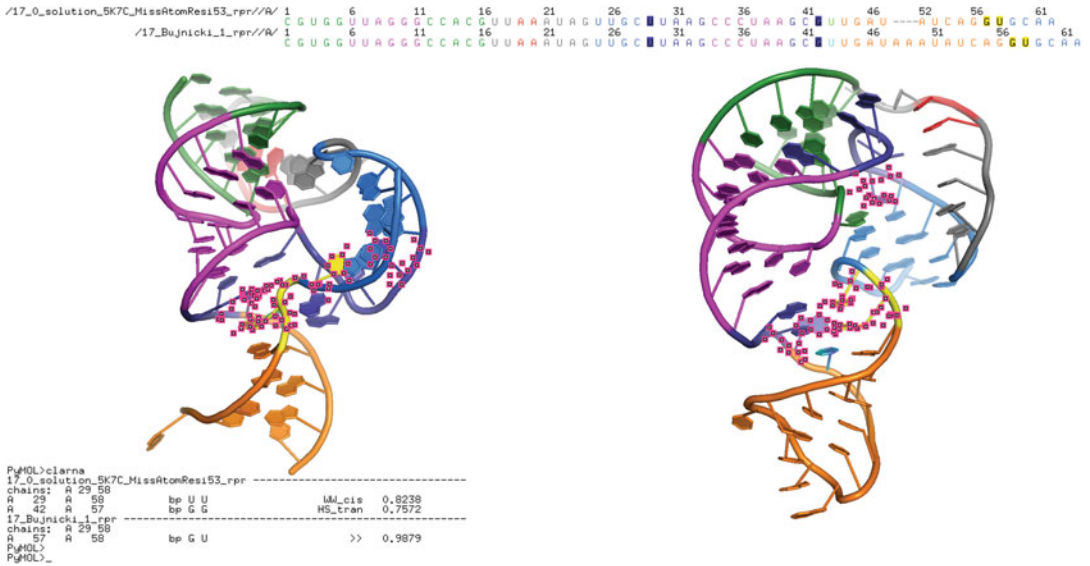


Fig. 7 The comparison of base pair contacts in PZ17 of RNA-Puzzles using “Clarna.” The example compares the nucleotides 29, 42, 57, 58 interactions in the reference structure (left) and the predicted structure (right, Bujnicki model 1)

3.3.4 *Generate Preview Thumbnails*

To have a quick inspection of hundreds of structure models, rna-tools provides a tool (pymol_preview_generator.py) to generate preview thumbnails for all PDB files in a folder. The following command generates the preview thumbnails in the structure folder:

```
$ pymol_preview_generator.py *.pdb
```

Multiple files can be selected in the file manager and overlaid together, as shown in Fig. 8. This program may help to pick the right model in the folder.

4 **Notes**

1. As only a limited number of chains and atoms can be deposited in PDB format, mmCIF format provides an alternative to save structures. As the predicted RNA structures are normally within the capability of PDB format, RNA-Puzzles toolkit uses PDB format for most of the tools. Rna-tools provides a script to convert mmCIF format to PDB format:

```
$ rna_pdb_toolsx.py -cif2pdb <file.cif>
```

or from PDB format to mmCIF format:

```
$ rna_pdb_toolsx.py -pdb2cif <file.cif>
```

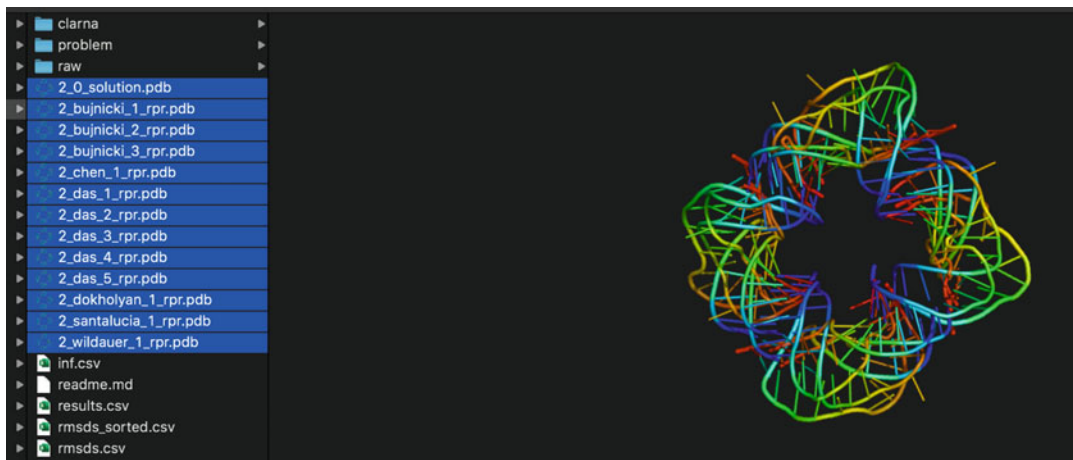



Fig. 8 The overlay of the selected PDB files in the file manager. The structures of PZ2 of RNA-Puzzles selected in Finder (the file manager in macOS) are shown as the results of “PyMOL Preview Generator” (pymol_preview_generator.py)

2. It is possible that new atom names and residue names may exist in a structure file, e.g., a new ligand is found in the reference structure. The existing atom name and residue name dictionaries in the program may not be able to deal with such a situation. The new names need to be added to the dictionaries, which are in the files “atom.list” and “residue.list” in the “data” folder of the RNA_normalizer program (and specified as RESIDUES_LIST and ATOMS_LIST in the python function). The first column in the file contains the atom or residue names before normalization, while the second column is the standardized names. We use “-“for the standardized name if the atom or residue need to be ignored.
3. Sometimes, the sequence for the predicted structure can be slightly different from the reference structure. In this case, there are two possible treatments: (1) Skip the different nucleotides in structure comparison; (2) mutate the nucleotide(s) in the predicted structure to the same ones as in the reference structure. In RNA-Puzzles evaluation, we use the first treatment and RNA_normalizer reads in an index file, which contains all the aligned nucleotides but skipping the unaligned ones. Nevertheless, if one needs to use the predicted structures as decoys in developing a force field, it is suggested to use the second option. Rna-tools provides functionality to help with the structural mutagenesis (the “—mutate” operation in rna_pdb_toolsx.py).
4. The functions to measure INF are the same for both RNA_normalizer and rna-tools. But the way to extract base pair

interactions is different for these programs. RNA_normalizer uses MC-annotate to identify the base pair interactions, while rna-tools uses ClaRNA for the same job. Therefore, the results can be slightly different.

Acknowledgments

We thank Maciej Antczak, Tomasz Zok, Jakub Wiedemann, and Marta Szachniuk for collaboration in the RNA-Puzzles toolkit project. We also thank the authors of the libraries used in our programs. We thank Rhiju Das for the idea of “align all” and Wayne Dawson for sharing unpublished code for “clarna_play.” We would like to also acknowledge all users of our tools for valuable feedback.

Funding

M.M. was supported by the “Regenerative Mechanisms for Health-ReMedy” grant MAB/20172, carried out within the International Research Agendas Program of the Foundation for Polish Science cofinanced by the European Union under the European Regional Development Fund. Z.M. was supported by the Single Cell Gene Expression Atlas grant 108437/Z/15/Z from the Wellcome Trust.

References

1. Strobel EJ, Watters KE, Loughrey D, Lucks JB (2016) RNA systems biology: uniting functional discoveries and structural tools to understand global roles of RNAs. *Curr Opin Biotechnol* 39:182–191
2. Sharp PA (2009) The centrality of RNA. *Cell* 136:577–580
3. Cech TR, Steitz JA (2014) The noncoding RNA revolution—trashing old rules to forge new ones. *Cell* 157:77–94
4. Long Y, Wang X, Youmans DT, Cech TR (2017) How do lncRNAs regulate transcription? *Sci Adv* 3:eaa02110
5. Buratti E, Baralle FE (2004) Influence of RNA secondary structure on the pre-mRNA splicing process. *Mol Cell Biol* 24:10505–10514
6. Luco RF, Misteli T (2011) More than a splicing code: integrating the role of RNA, chromatin and non-coding RNA in alternative splicing regulation. *Curr Opin Genet Dev* 21:366–372
7. Valencia-Sanchez MA, Liu J, Hannon GJ, Parker R (2006) Control of translation and mRNA degradation by miRNAs and siRNAs. *Genes Dev* 20:515–524
8. Al-Hashimi HM, Walter NG (2008) RNA dynamics: it is about time. *Curr Opin Struct Biol* 18:321–329
9. Mustoe AM, Brooks CL, Al-Hashimi HM (2014) Hierarchy of RNA functional dynamics. *Annu Rev Biochem* 83:441–466
10. Mustoe AM, Busan S, Rice GM et al (2018) Pervasive regulatory functions of mRNA structure revealed by high-resolution SHAPE probing. *Cell* 173:181–195.e18
11. Levitt M (1969) Detailed molecular model for transfer ribonucleic acid. *Nature* 224:759–763
12. Rother M, Rother K, Puton T, Bujnicki JM (2011) ModeRNA: a tool for comparative modeling of RNA 3D structure. *Nucleic Acids Res* 39:4007–4022

13. Popenda M, Szachniuk M, Antczak M et al (2012) Automated 3D structure composition for large RNAs. *Nucleic Acids Res* 40:e112
14. Zhao Y, Huang Y, Gong Z et al (2012) Automated and fast building of three-dimensional RNA structures. *Sci Rep* 2:734
15. Xu X, Zhao C, Chen SJ (2019) VfoldLA: a web server for loop assembly-based prediction of putative 3D RNA structures. *J Struct Biol* 207:235–240
16. Jonikas MA, Radmer RJ, Laederach A et al (2009) Coarse-grained modeling of large RNA molecules with knowledge-based potentials and structural filters. *RNA* 15:189–199
17. Sharma S, Ding F, Dokholyan NV (2008) iFoldRNA: three-dimensional RNA structure prediction and folding. *Bioinformatics* 24: 1951–1952
18. Krokhotin A, Houlihan K, Dokholyan NV (2015) iFoldRNA v2: folding RNA with constraints. *Bioinformatics* 31:2891–2893
19. Boniecki MJ, Lach G, Dawson WK et al (2016) SimRNA: a coarse-grained method for RNA folding simulations and 3D structure prediction. *Nucleic Acids Res* 44:e63
20. Parisien M, Cruz JA, Westhof E, Major F (2009) New metrics for comparing and assessing discrepancies between RNA 3D structures and models. *RNA* 15:1875–1885
21. Kufareva I, Abagyan R (2012) Methods of protein structure comparison. *Methods Mol Biol* 857:231–257
22. Hajdin CE, Ding F, Dokholyan NV, Weeks KM (2010) On the significance of an RNA tertiary structure prediction. *RNA* 16: 1340–1349
23. Zok T, Popenda M, Szachniuk M (2014) MCQ4Structures to compute similarity of molecule structures. *CEJOR* 22:457–473
24. Wiedemann J, Zok T, Milostan M, Szachniuk M (2017) LCS-TA to identify similar fragments in RNA 3D structures. *BMC Bioinformatics* 18:456
25. Magnus M, Antczak M, Zok T et al (2020) RNA-puzzles toolkit: a computational resource of RNA 3D structure benchmark datasets, structure manipulation, and evaluation tools. *Nucleic Acids Res* 48:576–588
26. Berman HM, Westbrook J, Feng Z et al (2000) The Protein Data Bank. *Nucleic Acids Res* 28: 235–242
27. Cock PJA, Antao T, Chang JT et al (2009) Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics* 25:1422–1423
28. Nelli F (2018) The NumPy library. *Python Data Analytics*:49–85
29. Parisien M, Major F (2008) The MC-fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature* 452:51–55
30. Waleń T, Chojnowski G, Gierski P, Bujnicki JM (2014) ClaRNA: a classifier of contacts in RNA 3D structures based on a comparative analysis of various classification schemes. *Nucleic Acids Res* 42:e151
31. Williams CJ, Headd JJ, Moriarty NW et al (2018) MolProbity: more and better reference data for improved all-atom structure validation. *Protein Sci* 27:293–315
32. Frickey T, Lupas A (2004) CLANS: a Java application for visualizing protein families based on pairwise similarity. *Bioinformatics* 20:3702–3704

INDEX

A

AlifoldZ 125
 Aptamer vii, ix, x, 252, 258

B

Base pairing probability (BPP) x, 1–4, 7,
 10–12, 15–19, 21, 28–32, 36, 39, 40, 42, 43, 100,
 109, 112–116, 123, 130, 131, 143, 151–153,
 155, 156, 181
 Beam search viii, 16, 24
 Benjamini-Hochberg test 127
 BLAST 125, 165, 166
 Boltzmann ensemble 17, 18, 37, 109
 Bowtie2 222
 BPSEQ format 81, 82, 85
 BUMHMM ix, 219–226, 229

C

CapR 2, 9, 10, 37
 c-di-GMP riboswitch 246
 CentroidFold viii, 3–7, 10–12,
 89, 101, 102, 227
 CentroidHomfold 7
 ChiX 179, 180
 Clanstix 279, 281
 ClaRNA 276, 280, 281, 284
 CLIP-seq ix, 197, 202
 CMfinder 125
 Competing endogenous RNAs (ceRNAs) 169
 Conditional log-linear models (CLLMs) 93, 94, 103
 Conditional random fields (CRFs) 4, 7, 93
 ContextFold 90, 94, 101–103
 CONTRAfold 4, 16–19, 24,
 36, 90, 94, 101–103
 CopraRNA 177–179
 CRISPRdirect 177, 187–188
 CS²BP²-Plot 140
 CS-ROSETTA-RNA 243, 246–247

D

DeepBind 197
 Deep neural networks (DNNs) vi, 197–214

Deformation profile (DP) 264–266,
 273, 277–279
 DiffMerge 269
 diffRNABow 130
 Dimethyl sulfate-sequencing (DMS-seq) v, 218
 dN/dS test 125
 DotKnot 18
 Dynamic programming (DP) v, vii, viii, 15,
 16, 18, 36, 37, 39, 46, 51, 52, 93, 96–98, 109–
 112, 122, 129, 148, 154, 155

E

European Nucleotide Archive (ENA) 86
 Evofold 46, 125
 Expression Quantitative Trait Locus (eQTL) 108

F

FARFAR2 xii, 233–247
 Free energy v, vii, viii, 7–9, 15–18,
 21, 23–29, 38, 39, 42, 43, 46, 51–73, 89, 90, 92–
 98, 107, 109–111, 122, 252
 Freiburg RNA tools viii, 178

G

γ -centroid structure 38, 41
 GC content 109, 115, 116, 200, 212
 Generalized centroid estimator (GCE), *see* γ -centroid
 structure
 Genetic algorithm x, 150
 GHOSTX 166
 Gibbs free energy 17
 Global importance analysis (GIA) 198–200,
 204–214
 GRAAL 149

H

Hairpin loop 9, 25, 42, 68, 90, 92,
 96, 97, 199, 201, 210, 258
 Hamming distance viii, 2, 10–13
 HARI 129, 130
 Hidden Markov models (HMMs) 7, 93, 219,
 220, 224, 225, 227

High-throughput RNA structural analysis
(HTS analysis) 217–229
HOMER 40
HuR 108, 200
Hybridization energy 164–166,
178, 179, 181

I

iFoldRNA 234, 264
Indel 124, 126, 128,
136, 138, 144
Influenza virus ix, 224–228
IntaRNA 164–166, 177–181
Integer programming (IP) 7, 177
Interaction network fidelity (INF) 264, 265,
273, 276–277, 284
In vivo click SHAPE (icSHAPE) 218, 220, 222
IPKnot 7, 8, 18, 224
Irreproducible discovery rate (IDR) 220, 227
IsoRank 150, 154, 161

K

Ka/Ks test 125

L

LandscapeFold vii, 49–73
LAST 7, 165
LinearFold 16–28
LinearPartition 16–19, 21–23, 28–30
LncRRISearch ix, 170, 171, 178, 189–192
lncTASR 169
Localfold 45
LocARNA-P 46
Longest Continuous Segments in Torsion Angle space
(LCS-TA) 264, 279
Long noncoding RNA (lncRNA) 108, 123,
125, 135–142, 163, 164, 167–171, 178, 189, 190

M

Maximum expected accuracy (MEA) vii, 4, 7,
16, 18, 21, 22, 28–32, 37–39, 44, 89, 96, 99–103
MC-annotate 274, 284
McCas skill model 4, 7, 9–11
Mean of circular quantities (MCQ) 264, 279
Mfold 35, 45, 89, 91
MicroRNA (miRNA) ix, 108, 163,
169, 176–178, 182, 183, 191
Minimum free energy (MFE) vii, viii, 4,
15–18, 21, 23–25, 37–39, 49, 51, 52, 69, 72, 91,
92, 96–98, 102, 103, 109, 114, 122–124, 130,
131, 133, 134, 143, 179–181
miRNA sponge, *see* Competing endogenous RNA
(ceRNA)

ModeRNA 271
Molprobit 279
Multi loop 67, 199, 201
Multiplexed •OH cleavage analysis (MOHCA) 239,
245
MutaRNA 109
MXfold viii, 90, 95, 101–103

N

NAST 264
Nearest-neighbor model 58–63,
72, 90–97, 123
Needleman-Wunch algorithm 154, 155
Non-coding RNAs (ncRNAs) vi, viii, ix, xii,
35, 40, 46, 49, 79, 121–126, 128–131, 137, 144,
147, 163, 175, 176, 178, 182, 191, 233
NUPACK 7
Nussinov algorithm 93, 100

P

p53 113, 116–119
Parallel analysis of RNA structure (PARS) 108,
218, 220
Parasor 36–42, 44–46, 109–116
Partition function v, 15–19, 28–29,
31, 43, 98–99, 109, 122–124, 129, 130, 164
PEELING vii, 80, 82–84, 86, 87
Piwi-interacting RNAs (piRNAs) 176, 178, 191
planeGraph2tree vii, viii, 81,
82, 84, 85, 87
Polymer physics model vii, 68
Position weight matrix (PWM) 197, 198
Positive selection ix, 121, 125–130,
137, 139, 140, 144
ProbCons 7
PROBer ix, 219–220, 222–224, 229
ProbKnot 18
Protein Data Bank (PDB) v, xi–xiii,
234, 236, 237, 239–243, 245, 246, 254, 255,
257–259, 265, 267, 269–272, 274, 278, 281–283
PseudoBase++ 81, 85, 86
PSMAlign 80
PyMOL 279–283

Q

qrna 125

R

Raccess 2, 10, 37, 42
Radial viii, 109–115, 117, 118
Random walk model 150–151, 161
Rchange 2, 7–10, 109
reactIDR ix, 219–222, 226–229

- Reactivityvi, ix, 218,
219, 223–225, 228
- remuRNA 109
- Renal cell carcinoma (RCC) 169
- ResidualBind ix, 197–214
- Reverse transcription (RT)218, 219, 221, 226
- Rfam7, 86, 101,
155–157, 160
- Rfoldviii, 19, 37, 41–44, 110–112
- RIblastix, 164–172, 189, 191
- RiboSNitchviii, 108–110, 114–118
- Ribosomal RNAs (rRNAs)v, xii, 18–19,
25, 30, 31, 79, 184–186, 220, 242, 252
- Riboswitch107, 108, 240, 246
- RintDviii, 2, 10–13
- RintW2, 11–13
- RIsearch2ix, 164–166, 169, 171, 172
- RMDB database108
- RNAalifold124, 126, 130
- RNA binding protein (RBP)ix, 40, 108, 197–214
- RNAcentral16, 17
- RNAcofold177, 179–181
- RNAcompeteix, 197, 199–202, 209–212
- RNAComposer263
- RNA 3D structurev, vi, xi–xiii,
251–260, 263–284
- RNAex108
- RNAfold10, 16–19, 23, 24, 45,
46, 101, 102, 123, 124, 130, 135
- RNAforester124, 126, 130, 136, 137
- RNALfold124
- RNAmakexii, 251–260
- RNAmutants109
- RNA_normalizer265, 267, 269,
271–274, 277, 283, 284
- RNAplfold19, 45, 199, 201, 211
- RNA-Puzzlesxii–xiii, 234, 237, 239,
240, 243, 244, 263–284
- RNA-RNA interactions (RRIs)viii, ix,
163–172, 175–192
- RNAseClust46
- RNAsenp109, 124, 126–128,
130, 132, 136
- RNAstructure16, 18, 89, 91, 101, 102, 155
- RNA-tools178, 265–272, 274,
276, 277, 279, 280, 282, 284
- RNAup177, 179–181
- RNAz46, 125
- Root-mean-square deviation (RMSD)xiii, 235–237,
241, 243–247, 264, 265, 269, 273–276, 279
- Rosetta234, 235, 237,
239, 242, 243, 245, 247
- ROSIE233–247
- RTED81
- Rtoolsviii, 1–13
- ## S
- Sankoff algorithm148
- ScanFold46
- Seed-and-extendix, 150, 165, 166, 171
- Selective 2'-hydroxyl acylation analyzed by primer
extension and sequencing (SHAPE-seq)v,
218, 223–228
- SELEXix, 49
- SHAPE and mutational profiling (SHAPE-MaP)vi,
218, 219
- Short interference RNAs (siRNAs)ix, 177
- SimRNA264
- Simulated annealing150
- Single nucleotide polymorphisms (SNPs)107–119,
124, 144
- Single nucleotide variants (SNVs)107–119,
198, 212
- SISSIZ125
- Small nuclear RNAs (snoRNAs)176, 178,
182, 184–187, 191
- SMETANA150
- snoGPS178, 182–187
- Snoscan178, 182–187
- SNPfold108, 109
- SpliceMap125
- SRP RNA19–22
- SSS-testix, x, 121, 125–130,
132–138, 140, 141, 144
- Stem probabilityviii, 40, 41,
109, 110, 112–114, 116–119
- Stochastic context-free grammars (SCFGs)42, 93
- Structure probingv, vi, ix, 46
- ## T
- TargetScan177, 182, 183
- tectoRNA259
- Tetraloop/tetraloop receptor (TTR)238,
252, 259
- Thermodynamic parameters89, 90, 95
- 3dRNAxii, 263, 275
- ThreshKnot18
- T-loop237, 239–244
- TOPASx, 147–161
- Topological centroid treeviii, 81, 83–87
- TORNADO93
- Trimomatic222
- tRNAxii, 3, 6, 8–10, 19, 25,
30, 31, 137, 156–158, 237, 242, 263
- Turner modelviii, 123

U

UNAFold89, 91
UTR.....163, 171, 176, 182, 183

V

VARNA.....7, 8, 156, 159
VfoldLA263
ViennaRNA packageviii, 4, 7,
101, 108, 132

W

Wobble pairvii, 107, 176, 179, 238

Z

Zuker's suboptimal algorithm17