Computational analysis of ribonucleic acid basepairs in RNA structure and RNA-RNA interactions

by

Daniel Lai

B.Sc., The University of British Columbia, 2010

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate and Postdoctoral Studies
(Bioinformatics)

The University of British Columbia
(Vancouver)

April 2016

© Daniel Lai, 2016
Abstract

Ribonucleic acids (RNA), are an essential part of cellular function, transcribed from DNA and translated into protein. Rather than a passive informational medium, RNA can also be highly functional and regulatory. Certain RNAs fold into specific structures giving it enzymatic properties, while others bind to specific targets to guide regulatory processes. With the advent of next-generation sequencing, a large number of novel non-coding RNAs have been discovered through whole-transcriptome sequencing. Many efforts have been made to study the structure and binding partners of these novel RNAs, in order to determine their function and roles.

This work begins with a description of my R package R4RNA for manipulating RNA basepair data, the building blocks of RNA structure and RNA binding. The package deals with the input/output and manipulation of RNA basepair and sequence data, along with statistical and visualization methods for evaluation, interpretation and presentation. We also describe R-CHIE, a visualization tool and web server built on R4RNA that visualizes complex RNA basepairs in conjunction with sequence alignments. We then conduct the largest known evaluation of RNA-RNA interaction methods to date, running state-of-the-art tools on curated experimentally validated datasets. We end with a review of cotranscriptional RNA basepair formation, summarizing biological, theoretical and computational methods for the process, and future directions for improving classical methods in RNA structure prediction.

All content chapters of this thesis has been peer-reviewed and published. The work on R4RNA has led to two publications, with the package used to great visual effect by various publications and also adopted by the RNA structure database RFAM. My assessment of RNA-RNA interaction is at present the only published evaluation of its kind, and will hopefully become a benchmark for future tool development and a guide to selecting appropriate tools and algorithms. Our published review on RNA cotranscriptional folding is well-received, being the first review specifically on its topic.
Preface


I was the main developer of the project, with fellow graduate students Jeff Proctor and Alice Jing-Yun Zhu as the two other main test users of the package before public release. Their feedback was subsequently used to improve and debug the package, with code contributions mainly by myself and Jeff Proctor. Dr. Irmtraud Meyer provided the idea for the package, and the funds and resources required to achieve the end product.


The corresponding analysis and manuscript were done by myself with ideas for the project, supervision and manuscript editing provided by Dr. Irmtraud Meyer.


Dr. Irmtraud Meyer completed the first draft. Fellow graduate student Jeff Proctor and myself addressed reviewer feedback with revisions to the text and figures of the manuscript, sharing first authorship on the final publication.
# Table of Contents

Abstract .................................................................................. ii

Preface .................................................................................... iii

Table of Contents ....................................................................... iv

List of Tables ........................................................................... viii

List of Figures ........................................................................... ix

Acknowledgments ..................................................................... xi

1 Introduction ........................................................................... 1

1.1 RNA biology ........................................................................ 1
  1.1.1 The role of RNA ............................................................ 1
  1.1.2 RNA chemistry and structure ........................................ 3

1.2 Functional RNA structures .................................................. 5
  1.2.1 Structured non-protein coding RNA ............................... 7
  1.2.2 Structured protein-coding RNA ...................................... 12
  1.2.3 RNA-RNA interactions ................................................ 15

1.3 Solving RNA structure experimentally ................................. 19
  1.3.1 Primary structure .......................................................... 19
  1.3.2 Secondary structure ....................................................... 19
  1.3.3 Tertiary structure .......................................................... 19
  1.3.4 Caveats ........................................................................ 20

1.4 Computational prediction of RNA basepairing ...................... 21
1.4.1 RNA secondary structure prediction ........................................... 21
1.4.2 Energy-based predictions ......................................................... 21
1.4.3 Evolutionary-based methods ..................................................... 23
1.5 Objectives and contributions ....................................................... 24

2 Visualization of RNA Basepairs and Alignments .......................... 25
2.1 Visualizing RNA structure .......................................................... 26
2.1.1 Primary structure ................................................................. 26
2.1.2 Secondary structure ............................................................. 26
2.1.3 Tertiary structure ................................................................. 27
2.2 Types of RNA secondary structure visualization .......................... 28
2.2.1 Dot plots ........................................................................... 28
2.2.2 Circle and linear diagrams ..................................................... 28
2.2.3 Stem-loop diagrams ............................................................. 29
2.2.4 Other diagrams ................................................................. 31
2.2.5 Conserved RNA structure ..................................................... 32
2.3 Materials and methods ............................................................... 35
2.3.1 The R-CHIE web server ....................................................... 35
2.3.2 Input ............................................................................. 43
2.3.3 Output ............................................................................. 43
2.3.4 The R4RNA R package ....................................................... 43
2.4 Results and discussion ............................................................... 44

3 Assessment of RNA-RNA Interaction Prediction Methods .............. 48
3.1 Introduction ............................................................................ 48
3.2 Algorithm strategies ................................................................. 49
3.3 Methods .............................................................................. 51
3.3.1 Energy-based RNA-RNA interaction prediction programs .......... 51
3.3.2 Comparative RNA-RNA interaction prediction tools ................. 56
3.3.3 Multiple sequence alignment generation programs ...................... 59
3.3.4 Related works and tools ....................................................... 61
3.4 Datasets .............................................................................. 62
3.4.1 Bacterial sRNA .................................................................. 62
3.4.2 Fungal snoRNA ................................................................. 63
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.3</td>
<td>Multiple sequence alignments</td>
<td>64</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Performance measures</td>
<td>65</td>
</tr>
<tr>
<td>3.5</td>
<td>Results</td>
<td>66</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Minimum free energy results on sRNA dataset</td>
<td>66</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Suboptimal interaction results on sRNA dataset</td>
<td>70</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Effect of increasing target sequence size on sRNA dataset</td>
<td>76</td>
</tr>
<tr>
<td>3.5.4</td>
<td>MFE energy-based results on snoRNA dataset</td>
<td>76</td>
</tr>
<tr>
<td>3.5.5</td>
<td>Suboptimal energy-based results on snoRNA dataset</td>
<td>81</td>
</tr>
<tr>
<td>3.5.6</td>
<td>Comparative predictions for sRNA dataset</td>
<td>88</td>
</tr>
<tr>
<td>3.5.7</td>
<td>Comparative predictions for snoRNA dataset</td>
<td>89</td>
</tr>
<tr>
<td>3.5.8</td>
<td>Effect of different aligners on predictive performance</td>
<td>94</td>
</tr>
<tr>
<td>3.5.9</td>
<td>Combining energy-based and comparative results</td>
<td>99</td>
</tr>
<tr>
<td>3.5.10</td>
<td>Basepair covariation in datasets and background</td>
<td>99</td>
</tr>
<tr>
<td>3.6</td>
<td>Discussion</td>
<td>101</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Settings and overfitting</td>
<td>101</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Performance effects of conservation</td>
<td>103</td>
</tr>
<tr>
<td>3.6.3</td>
<td>Target size and interaction search space</td>
<td>104</td>
</tr>
<tr>
<td>3.6.4</td>
<td>Runtime and memory performance</td>
<td>104</td>
</tr>
<tr>
<td>3.7</td>
<td>Conclusion</td>
<td>105</td>
</tr>
<tr>
<td>4</td>
<td>Cotranscriptional RNA Folding and Prediction</td>
<td>109</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction and motivation</td>
<td>109</td>
</tr>
<tr>
<td>4.2</td>
<td>Experimental and theoretical evidence for cotranscriptional folding</td>
<td>111</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Directionality of transcription</td>
<td>111</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Transcription, transcription speed and variations thereof</td>
<td>112</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Self-interactions including transient RNA structures</td>
<td>112</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Interactions with other molecules</td>
<td>115</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Summary</td>
<td>119</td>
</tr>
<tr>
<td>4.3</td>
<td>Capturing cotranscriptional folding in methods for RNA secondary structure prediction</td>
<td>123</td>
</tr>
</tbody>
</table>
List of Tables

Table 3.1 RNA-RNA interaction methods used in assessment ........................................ 51
Table 3.2 Energy-based MFE sRNA results: default vs optimal settings ....................... 69
Table 3.3 Energy-based sRNA results: MFE vs suboptimal ........................................ 75
Table 3.4 Energy-based MFE snoRNA result: short vs full-length input ....................... 80
Table 3.5 Energy-based snoRNA results: MFE vs suboptimal .................................... 87
Table 3.6 Comparative suboptimal sRNA results as a function of %ID .......................... 88
Table 3.7 Comparative suboptimal snoRNA results as a function of %ID ..................... 91
Table 3.8 Energy-based suboptimal snoRNA results: short vs full-length input ........... 93
Table 3.9 Comparative suboptimal snoRNA result: short vs full-length input .............. 94
Table 3.10 Combining energy-based and comparative results ................................... 100
Table 3.11 Dataset covariation and conservation scores ........................................... 101
# List of Figures

| Figure 1.1 | Ribonucleic acid monomer chemical structure | 4 |
| Figure 1.2 | Ribonucleoside chemical structures | 5 |
| Figure 1.3 | RNA Watson-Crick and wobble basepairs | 6 |
| Figure 1.4 | Yeast tRNA-Phe structure | 7 |
| Figure 1.5 | Ribosomal RNA catalytic action | 9 |
| Figure 1.6 | Riboswitch translation and transcriptional control | 11 |
| Figure 1.7 | RNA-RNA interaction examples: snoRNA and miRNA | 17 |
| Figure 1.8 | Bacterial small RNA RyhB binding sodB mRNA | 18 |
| Figure 2.1 | Dot plot of RNAFOLD predicted Cripavirus structure | 29 |
| Figure 2.2 | Circular diagram of Cripavirus IRES | 30 |
| Figure 2.3 | Linear diagram of Cripavirus IRES | 30 |
| Figure 2.4 | Stem-loop diagram of Cripavirus IRES | 31 |
| Figure 2.5 | Variant stem-loop diagram of Cripavirus IRES | 32 |
| Figure 2.6 | Mountain plot of RNAFOLD predicted Cripavirus IRES | 33 |
| Figure 2.7 | RALEE plot showing conserved alignment basepairs | 34 |
| Figure 2.8 | R-chie single structure arc diagram | 36 |
| Figure 2.9 | R-chie double structure arc diagram | 37 |
| Figure 2.10 | R-chie overlapping structure arc diagram | 38 |
| Figure 2.11 | R-chie single structure covariation plot | 39 |
| Figure 2.12 | R-chie single structure lettered covariation plot | 39 |
| Figure 2.13 | R-chie double structure covariation plot | 41 |
| Figure 2.14 | R-chie overlapping structure covariation plot | 42 |
| Figure 2.15 | SAM riboswitch alternative structure by Zhu et al. | 45 |
| Figure 2.16 | RMPR structure determined by SHAPE chemical probing by Rogler et al. | 46 |
Figure 2.17 Representative RB1 wild-type and mutant structures by Kutchko et al. 47

Figure 3.1 MCC heatmap of MFE energy-based sRNA results: default vs optimal options 67
Figure 3.2 Clustering energy-based sRNA results 68
Figure 3.3 MCC heatmap of energy-based sRNA results: MFE vs suboptimal 72
Figure 3.4 MCC distribution of MFE energy-based sRNA results 73
Figure 3.5 MCC distribution of suboptimal energy-based sRNA results 74
Figure 3.6 Energy-based sRNA results as a function of input length 78
Figure 3.7 MCC heatmap of MFE energy-based MFE snoRNA results: short vs full-length 79
Figure 3.8 MCC distribution of MFE energy-based snoRNA results 83
Figure 3.9 MCC distribution of suboptimal energy-based snoRNA results 84
Figure 3.10 MCC heatmap of suboptimal energy-based snoRNA results: short vs full-length 85
Figure 3.11 Clustering energy-based snoRNA results 86
Figure 3.12 Comparative sRNA results as a function of minimum %ID 89
Figure 3.13 TPR vs PPV density plot for energy-based sRNA results 90
Figure 3.14 Comparative snoRNA results as a function of %ID 92
Figure 3.15 Comparative MCC performance across different aligners 96
Figure 3.16 Comparative TPR performance across different aligners 97
Figure 3.17 Comparative PPV performance across different aligners 98
Figure 3.18 RNA-RNA interaction dataset basepair covariation 102
Figure 3.19 CPU runtimes as a function of input length 106
Figure 3.20 Memory usage as a function of input length 107

Figure 4.1 Alternative transient RNA structures in the hok-sok system 127
Figure 4.2 Hypothetical cotranscriptional folding pathways 128
Acknowledgments

This work would not have been possible without the time and support of those around me. Firstly, I’d like to thank my supervisor Irmtraud Meyer for guiding me throughout the degree and also co-supervisor Paul Pavlidis for his guidance in the final year. I’d also like to thank my supervisory committee members Steven Jones and Martin Hirst for their guidance and feedback through the duration of my degree.

I also extend my thanks to my lab mates in the Meyer and Pavlidis lab that I’ve had the pleasure of working along side with throughout these years. Special thanks go to the Meyer Lab members Rodrigo Goya, Jeff Proctor, Alborz Mazloomian, Evan Gatev, Yang Shu, Alice Zhu and Ian Wood.

I’d like to thank the University of British Columbia, CIHR/MSFHR Bioinformatics Training Program and the Natural Sciences and Engineering Research Council of Canada for their generous funding enabling this research.

Finally, I’d like to thank my family for their encouragement and support.
Chapter 1

Introduction

We start by introducing the concept of basepairing in ribonucleic acids (RNA), its history, biology and significance to molecular biology. We then provide a general history and summary of computational methods developed thus far to predict basepairs, establishing the state of the art in the field.

1.1 RNA biology

1.1.1 The role of RNA

The central dogma of molecular biology proposed by Francis Crick in 1958 hypothesized a flow of information in all living things from DNA to RNA to protein [1, 2], sidelining ribonucleic acids as a transient intermediate state. By 1961, experimental evidence published for DNA replication [3], RNA transcription [4–6] and protein translation [7] seemed to support this casting of RNA as an “unstable intermediate” [8], although much remained to be discovered regarding RNA’s potential in regulating the steps of the dogma.

In 1961, whilst speculating the mechanisms of protein induction and repression, François Jacob and Jacques Monad suggested the existence of messenger RNAs (mRNA) transcribed from DNA, which could be under the influence of “regulator genes”, ultimately affecting the observed rates of protein synthesis [9]. At the time of writing however, they were unsure whether the functional product of regulator genes would be of protein or RNA, and whether this product acted upon the DNA gene or the mRNA transcript.

In following decades of research, the general conclusion seemed to be that proteins were the
functional product of regulator genes, producing DNA-binding proteins controlling transcription (e.g. transcription factors) [10], or RNA-binding proteins affecting mRNA splicing and stability [11]. The main role of RNA was slightly expanded, to include messenger RNAs, ribosomal RNAs (rRNA) [8] and transfer RNAs (tRNA) [1], seen simply as scaffolding adaptors for the protein translation machinery, respectively. While cases of RNA-mediated gene regulation were found, they seemed to be the exception rather than the rule, with only eleven naturally occurring cases observed in prokaryotes, none in eukaryotes by 1988 [12].

By 1998 however, the role of RNA in gene regulation would begin to take a huge turn, starting with the publication by Fire and Mello showing that double-stranded RNA could effectively affect gene expression in Caenorhabditis elegans through RNA interference (RNAi) [13]. Followed by demonstrations that the same technique could be applied to human cells in 2001 by the Tuschl group [14]. RNA-mediated gene regulation is now considered critical regulatory process in gene expression, with implications in disease etiology, diagnosis, prognosis, and therapy [15–17].

While the Human Genome Project released its first draft of DNA sequences by the early 2001 [18, 19], large-scale efforts to characterize all RNA transcripts in mammals were also in full swing. Early results by the FANTOM consortium for mouse utilized expressed sequence tag (EST) and full-length cDNA (complementary DNA, generated from RNA by reverse transcriptase) sequences along with tiling arrays, initially focusing on identifying protein-coding transcripts [20, 21]. The unexpected discovery a large number of novel non-coding RNAs led to the use of unbiased transcript identification techniques, such as serial analysis of gene expression (SAGE) [22] and cap analysis gene expression (CAGE) [23]. Major findings of these studies include the pervasive transcription of the entire mammalian genome, the majority of these transcripts being non-coding from intronic and intergenic regions [24].

With the advent of next-generation sequencing (NGS) starting in 2005, DNA could be sequenced at a fraction of the time and cost of previous Sanger-based methods [25–27]. Protocols for RNA-seq soon followed, applying next-generation sequencing technologies to sequencing cDNA from whole transcriptomes, allowing for the high-throughput generation of RNA data [28, 29]. Studies such as the ENcyclopedia of DNA Elements (ENCODE) Project have since shown that while only $\sim 2\%$ of the genome is protein-coding, $> 75\%$ of the genome is cumulatively transcribed throughout the human body [30–33], likely containing novel ncRNAs. [34–37]. Besides post-transcriptional gene regulation via RNA interference, non-coding RNAs have been shown to be involved in chromatin structure [32, 38], epigenetic regulation [39, 40], RNA splicing [41, 42], and catalysis [43, 44]. With the exact mechanisms, classes and number of RNAs remaining to be determined, the study of RNA shows much promise in yielding novel insights into the function of
living organisms neglected for over half a century.

1.1.2 RNA chemistry and structure

A RNA molecule, or ribonucleic acid, is a biomolecule chemically consisting of a phosphate connected to a ribose sugar from which extends a nitrogenous base (Figure 1.1). Multiple RNA monomers can form a linear polymer by forming covalent phosphodiester bonds between the phosphate of one molecule to the sugar of another, forming a sugar-phosphate backbone off which hangs various bases. The most commonly used four bases and their abbreviations in RNA are adenine (A), guanine (G), cytosine (C), uracil (U) (Figure 1.2). Chemically, C and U are pyrimidines, containing a hexagonal heterocyclic ring made of four carbon and two nitrogen atoms. A and G on the other hand, are purines, containing the hexagonal pyrimidine fused to a pentagonal imidizole ring, physically larger than the purines. Differentiating one base from another is various nitrogen-containing amines and oxygen-containing carbonyls that surround the heterocyclic rings. The series of basepairs in this linear polymer, often described as a string of letters consisting of A, C, G and U, is known as the primary structure or sequence of the RNA.

When two complementary basepairs are placed with nucleotides facing each other in the correct orientation, a non-covalent hydrogen bond forms. Specifically, amines can act as hydrogen bond donors with a relatively positive charge, while carbonyls act as acceptors with a relatively negative charge. In addition to the right hydrogen bond donors and acceptors, the physical location and orientation of these molecular groups has to be correct, resulting in the very specific complementary Watson-Crick basepairs of A with U, and G with C (Figure 1.3). In both pairings, we have a smaller pyrimidine pairing with a larger purine, with G-C pairs forming three hydrogen bonds and A-U pairs forming two. Following these same steric and electrochemical considerations then, a third wobble basepair of G-U is also valid which forms two hydrogen bonds, bringing the list of canonical basepairs to three [45] (Figure 1.3). A description of the primary structure, in addition to all the basepaired and unpaired nucleotides, describes the secondary structure, which can often be visualized as paired stems and unpaired loops on a two dimensional figure.

While basepaired stems are represented as a ladder-like figure on a two-dimensional figure, in reality it assumes a double helix structure much like double-stranded DNA [46, 47]. In the absence of positive cations, the negatively-charged phosphate backbone of RNA helices repels against other parts of the RNA, preventing the formation of a compact structure [48]. With the addition of positive cations such as Mg$^{2+}$, these positively charged ions bind to specific locations on the RNA, allowing for the formation of stable compact three-dimensional RNA structures, or the ter-
Figure 1.1: Two ribonucleic monomer shown each with a ribose sugar (a furanose ring consisting of four carbons and one oxygen atom), from which a nucleotide base extends from the 1’ carbon (guanine and ‘R’), connecting with phosphates between the 3’ carbon of one ribose to the 5’ carbon of another.

© Warraich Sahib. Retrieved October 09, 2015 from Wikimedia Commons Used under Creative Commons Attribution-ShareAlike 3.0 Unported License

Tertiary structure via non-covalent bonds (Figure 1.4). In the majority of cases, the hydrogen bonds formed as part of the secondary structure are stable enough to remain unchanged upon tertiary structure formation. A fully solved tertiary structure describes the three-dimensional coordinates of each atom in an RNA molecule relative to each other, with varying degrees of resolution (atomic accuracy) depending on its experiment and purpose.

Having described primary, secondary, and tertiary structure then, it is hopefully obvious how RNA folding has been described as hierarchical, with each sequential step contingent on the correct formation of the previous in the majority of known cases [49]. The actual formation of these structures is also highly dependent on the immediate environment of the molecule, with temperature, pH, and ionic concentration of the solution playing large effects on primary, secondary and tertiary structure stability. Adding to the complexity in vivo, RNA folding is a highly kinetic process, occurring whilst being transcribed, and potentially being influenced intentionally and indirectly by
Figure 1.2: The four basic RNA nucleotides, shown here as nucleosides (attached to ribose). Uracil and cytosine are pyrimidines (six-membered heterocyclic ring composed of two nitrogens), while adenine and guanine are purines (fused pyrimidine-imidazole ring system).


other macromolecules in the cell.

1.2 Functional RNA structures

While adopting a structure does not necessarily mean that the RNA will be functional, many functional examples of RNA are due to a highly specific structure. Broadly, we can classify functional structured RNAs into coding and non-coding RNA. For non-coding structured RNAs, the most well-known examples are most likely tRNA and rRNAs, with other key examples being catalytic
Figure 1.3: Watson-Crick and wobble basepairing formed between pyrimidine-purine pairs. Three basepairs ordered in order of strength, with the strongest GC pair forming three hydrogen bonds.


ribozymes. For coding RNAs, RNA structure often plays a more regulatory role, often found in the untranslated regions of the transcript but also within the coding region [50]. Structure on coding
RNAs have been found to directly influence translation efficiency, guide RNA localization, and RNA stability [51].

### 1.2.1 Structured non-protein coding RNA

There exists a relatively small number of functional non-protein coding structured RNAs, varying in size and roles. Some of these are completely independent, some exist in larger ribonucleoprotein complexes, while others are commonly reoccurring motifs and structures found in a large number of RNA sequences.
Transfer RNA and ribosomal RNA

The earliest studied example of ncRNA is transfer RNA, whose secondary cloverleaf structure was proposed in 1965 [52], paving the way to the full structure that was determined a decade later [53, 54] (Figure 1.4).

Unlike tRNA, which carries out its function independently as a single molecule, ribosomal RNA (rRNA), is an example of a structured RNA that functions within a ribonucleoprotein (RNP) complex. rRNA was initially assumed to be structural while ribosomal protein defined ribosome function [55], but the current understanding is that rRNA is a ribozyme or a catalytic RNA and is responsible for the chemical reactions that define translation [56] (Figure 1.5). Due to the size and complexity of the ribosome, initially only the secondary structure could be studied by phylogenetic comparison of primary sequences around the early 80s [55]. It was not until year 2000 that we had high-resolution crystal structures resolving the tertiary structure, which also confirmed the ribosome as a ribozyme [57-59].

Ribozymes

The first discovered ribozyme is actually much earlier, and was made in 1982 [43, 61], where the intron in the rRNA of the protozoan ciliate *Tetrahymena thermophila* was found to auto-excise in the absence of proteins and other small RNAs. In time, it was shown that the splicing was intrinsic to the RNA structure, and that this structure was found in a wide number of other viruses, bacteria and eukaryotic organelles and is known today as self-splicing Group I introns [62, 63]. Much of the catalytic sites and mechanisms of the structure had been determined and explained using secondary structures [62] within a decade of its discovery, and only 15 years after was a tertiary structured crystallized and solved [64].

A similar but mechanistically distinct group of self-splicing introns were also discovered in 1982, known today as the Group II introns [65]. These were shown to be self-splicing by 1986, and are excised in a manner similar to eukaryotic nuclear pre-mRNA introns [62, 65]. So far these self-splicing structures have only been found in bacteria and the organelles (mitochondria and chloroplast) of fungi, plants, and protists [66]. As of today, no Group II intron has been found in the nuclear genome of eukaryotes [66]. Crystallized in 2008, the catalytic core of Group II introns have been found to be structurally and functionally similar to certain small nuclear RNAs in the eukaryotic nuclear spliceosome, implying an evolutionary relationship [67].

Very recently, it was shown that yeast small nuclear RNAs (snRNAs) within the eukaryotic spliceosome RNP complex are also ribozymes [68], functioning in a manner analogous to Group
Figure 1.5:
Current understanding of ribosomal peptide-bond formation [60] a chemical reaction binding two peptides (R₁, R₂) at the end of tRNAs. b Binding of the aminoacyl-tRNA (aa-tRNA: amino acid bound tRNA) to the A site and peptidyl-tRNA (protein bound tRNA) to the P site. c Positioning of peptidyl-tRNA A72 2’OH and rRNA A2451 N3 around the α-amino nucleophile. D, E current model of catalysis involving the shuffling a series of protons around peptidyl-tRNA A72 2’OH.

II self-splicing enzymes. Like ribosomes, further research and understanding of the molecular chemistry has shown the RNA moieties to be the catalytic component, while proteins are found to be more structural in nature that previously assumed. It has thus been hypothesized that these are remnants of an ancient “RNA world” (discussed in detail later) absent of DNA and protein, having seen the catalytic potential of RNA [69].

Following soon after the discovery of self-splicing introns in 1982, the RNA moieties of the Ribonuclease P (RNaseP) RNP was determined to be the active component and another example of a structured ribozyme [44, 70]. Homologs were found to be ubiquitous in all branches of life, responsible for processing the 5’ leader sequence of precursor tRNA [71], although only those in Bacteria have been shown to work without protein subunits in vitro. The tertiary structure of RNaseP was crystallized and solved nearly two decades later in 2005 [72, 73].

The RNAseP and splicing ribozymes described thus far are specifically metalloenzymes, requiring the presence of specific metal ions that are positioned in specific locations in the catalytic site with direct roles in catalysis [61]. Thus far these have been the most common and abundant types of ribozymes, and was thought that all ribozymes had to be metalloenzymes.

This was until the final abundant class of ribozymes currently known was discovered in 1986 [74, 75], as an autolytic structural motif found in plant viruses. Now known as the hammerhead ribozyme, these self-cleaving ribozymes can cleave their own phosphodiester backbone in the absence of metal ions under correct conditions. Like earlier discussed examples, the three-dimensional structure was unknown until crystallization efforts were successful in the mid 1990s [76, 77], becoming the first ribozyme to be successfully crystallized [61].

Riboswitches

Discovered in the early 2000s and characterized by 2002 [78], riboswitches are structures in the 5’ untranslated region of mRNAs that control the expression of the downstream transcript [79] (Figure 1.6). Found predominantly in eubacteria, these structures undergo structural changes when bound to specific metabolites, inhibiting the expression of the gene. The gene expresses a product that is usually part of the biosynthesis or transport pathway of the bound metabolite, thus creating a self-regulating feedback loop to control metabolite levels [80].

A growing list of a dozen or so metabolite-specific classes exist, with a few crystallized examples giving us a clear view of their mechanisms [81]. Generally, a riboswitch has two distinct structural conformations, one that allows for translation or transcription of the downstream product, while the other one inhibits it commonly by blocking the ribosome, or causing a pre-mature
release of the polymerase [82]. Depending on the riboswitch, structures can range from a small pseudoknot, to complex multi-stem structures [81].

Figure 1.6: General control mechanisms for riboswitches. For transcriptional control, the unbound state allows the formation of the anti-terminator structure which prevents the formation of the terminator stem. When bound, the terminator stem prevents the full-length transcription of the mRNA. In translational control, the binding of the metabolite makes the ribosome binding site (RBS) inaccessible, preventing translation of the mRNA.


Long non-coding RNAs

Out of the > 60,000 recently sequenced long non-coding RNAs (IncRNAs) in human [83], there exist several well-characterized examples where structural motifs have been strongly suggested to play a part in their function. The most well-studied is the Xist RNA, discovered in the early 90s, and found to play an essential role in X-chromosome inactivation [84]. Over 16kb long, the full structure remains to be solved, but studies have shown that secondary structures in specific regions
are involved in the recruitment of histone modification proteins leading to gene silencing [85–87]. Other gene silencing secondary structures have been found on other IncRNAs, such as HOTAIR [88], a 2148nt long RNA that silences the HoxD locus involved in epidermal tissue development [89] and ANRIL, which when mutated has been found to be associated with heart disease [90, 91] and cancer [92].

1.2.2 Structured protein-coding RNA

Even for protein-coding messenger RNAs, there have been examples where RNA structures are required for the correct processing of the transcript. Many examples have been reviewed by other works [50, 51], and we highlight a few examples below to demonstrate the role of RNA structure.

### Splicing

In an example demonstrated on the cardiac troponin T (cTNT) gene, a stem-loop structure 3’ of intron 4 is targeted by the protein MBNL1, which when bound represses the inclusion of the exon 5. In patients with the genetic disorder myotonic dystrophy type 1, (CTG)$_n$ repeat expansions in the 3’UTR of the DMPK gene result in CUG repeats its transcript that form stable RNA stem-loops that improperly sequester MBNL1. The depletion of MBNL1 allows splicing factor U2AF65 to bind the 3’ region of intron 4 on cTNT in the absence of the stem-loop, causing the inclusion of exon 5. Stabilization of the stem-loop structure has been shown to block U2AF65 binding [93].

In a simpler example, the SMN2 gene has been shown to normally skip exon 7 due to a stem-loop structure 5’ of the exon preventing the recruitment of splicing factor U1. In point mutations experiments, it has been shown that weakening this stem-loop promotes the inclusion of exon 7, and that compensatory mutations restoring the stem-loop restores the skipping [94].

In general, RNA structures have been found to prevent spliceosome assembly by hiding single-stranded splice sites and enhancer binding sites. Alternatively, the same stem-loops can also hide splicing repressor sites promoting splicing [95]. Genomic wide scans have shown an association between conserved RNA secondary structures and splice-site selection, suggesting many more uncharacterized examples in the human genome [96].

### Localization

In yeast, when the protein folding capacity of the endoplasmic reticulum is compromised, the transmembrane Ire1 protein form foci that recruit the HAC1 mRNA to activate the unfolded protein response. It has been shown that a stem-loop structure in the 3’ UTR containing a conserved
bipartite sequence element is necessary and sufficient for the localization of the \textit{HAC1} mRNA to these loci, \textit{provided the transcript remains untranslated}. To ensure the lack of translation, the single intron in \textit{HAC1} forms a structure that binds to the 5’UTR resulting in ribosomal stalling. Once recruited, Ire1 initiates the non-conventional splicing of the \textit{HAC1} intron, releasing the ribosomal stalling and allowing the translation of HAC1 activating the unfolded protein response involving 7-8\% of the yeast genome [97].

In a more general case, “RNA zipcodes” located on the 3’UTR are known to target specific mRNAs to subcellular regions in eukaryotic organisms via anchoring and transport. Of the zipcodes known, a wide variety of lengths, sequences, structures and trans-acting factors are involved, making further generalization difficult [51, 98].

\textbf{Translation}

The presence of RNA structure can often physically stall the progress of ribosomes, preventing translation from completing such as the intron structure in the \textit{HAC1} example above.

Another example is a metabolite-independent “RNA switch” in the human \textit{VEGFA} 3’UTR that is highly analogous to bacterial riboswitches [99]. Specifically, during normal normoxia, cell signaling protein interferon-gamma (IFN-\(\gamma\)), represses the translation of several proteins via the IFN-\(\gamma\)-activated inhibitor of translation complex (GAIT), and additionally also induces the proteosomal degradation of heterogeneous nuclear ribonucleoprotein L (HNRNPL). In these conditions, GAIT binds to the VEGFA switch, causing a conformational change that inhibits translation. In hypoxic conditions, IFN-\(\gamma\)-mediated proteosomal degradation of HNRNPL is blocked by the proteasome inhibitor MG132, increasing the levels of HNRNPL. HNRNPL competes with GAIT to bind the VEGF switch, out-competing GAIT and changes the switch to a more stable structure that also allows VEGFA to undergo translation. Like a standard riboswitch then, the switch can assume two structures, one inducive of translation while the other not. Instead of the presence of a metabolite causing the conformation change however, it is the mutually exclusive, stimulus-dependent binding of either GAIT or HNRNPL [99].

Instead of blocking translation, RNA structures are also often used to initiate non-conventional translation by forming Internal Ribosome Entry Sites (IRES), characterized in 1988 [100]. Whereas normal translation begins with the recognition and binding of initiation factors at the 5’-end cap of the mRNA transcript, many mRNAs in viruses and eukaryotes have been found to contain highly structured IRES in the 5’ UTR that allow these transcript to bypass cap-dependent translation [101]. Highly variable in structure, sequence and size, depending on the specific structure unique to the
family, species, or even transcript, the ribosome docks with the IRES without initiation factors, a start codon, and may even mimic the initiating tRNA [101, 102]. In a very recent work, it was shown that bacterial ribosomes were capable of translating eukaryotic mRNAs using IRES, despite billions of years of divergence since the last universal ancestor [103].

Yet another example seen in viruses to initiate non-conventional translation is ribosome shunting, characterized in the early 90s [104]. While a ribosome scans the transcript to start translation, it encounters a specific secondary structure causing it to dissociate and continue downstream of the structure. The region bypassed is often inhibitory to scanning due to strong secondary structure or multiple start codons [105].

Degradation

RNA structures have also been found to be required for the targeted degradation of specific transcripts. In an example found in yeast, the RPS28B gene forms a conserved hairpin structure on its 3'UTR. In the presence of abundant Rps28b protein, a self-regulating loop occurs when Rps28b binds the hairpin loop, causing the recruitment of decapping machinery that leads to the degradation of the transcript [106].

In humans, many examples of single nucleotide polymorphisms (SNP) causing a change in RNA secondary structure resulting in a decrease of mRNA stability have been found [50, 51]. A specific example is observed in the catechol-O-methyltransferase gene, where three haplotypes exhibit three different structures, correlating with different levels of pain sensitivity. It has been shown that the more stable the resulting structure is, the lower the expression of the gene was, correlating with higher pain sensitivity [107].

Conclusion and the RNA World

It has been hypothesized that ribozymes and riboswitches may be ancient remnants of an RNA World [108], where RNA was the sole propagator of genetic information independent of DNA and proteins. First proposed by Carl Woese in 1967 in *The Genetic Code* along with publications by Crick [109] and Orgel [110], it has been hypothesized since the 60s that modern life may have roots in a world where RNA was both the carrier and propagator of genetic information. The theory postulates that over time, DNA took over the role of information storage being more chemically stable, and proteins came to dominate the catalysis of reactions and regulation of systems [108, 111]. Whether RNA spontaneously started life (RNA-first) or whether some alternative system lead up to RNAs remains to be determined [112, 113].
Using the basic building blocks of basepairs and tertiary interactions, RNAs can form structures varying from simple hairpins to complex multidomain structures. Functional structures are found in all branches of life across both coding and non-coding transcripts. At times, they simply serve as passive but specific binding targets for protein complexes to initiate other processes such as degradation and localization. RNA structures can also be highly active, such as riboswitches regulating gene expression and ribozymes actively ligating, cleaving and splicing. A growing list of synonymous and untranslated mutations associated with diseases are also being explained by misfolded RNA, giving further motivation to determine and study RNA structures.

Whole-transcriptome probes have shown an abundance of previously unstudied structures, promising the study of RNA structures to be an active field of research in years to come [50]. While solving the tertiary structures remain the golden standard for RNA structure determination, the study of secondary structure can be done in a much more cost and time-efficient manner. At the level of secondary structures, computational predictions can greatly aid and speed up the search for structures, highlighting the importance and need for fast and accurate basepair prediction algorithms.

### 1.2.3 RNA-RNA interactions

While we have only focused on RNA structure where basepairing occurs intramolecularly, there also exists functional instances of intermolecular basepairing, or RNA-RNA interactions (RRI). Like RNA structure, basepairs are the building blocks of RRI, serving as the stabilizing force in some examples while only serving as a sequence-specific adaptors in others. A small but highly common example of an RRI is the specific binding of tRNA anti-codons to mRNA codons in the translation process. The consistency of Watson-Crick basepairs allows the unambiguous binding of codons and anti-codons in the first two bases, while the flexibility of the third wobble base allows for a single tRNA anti-codon to bind to multiple codons. Like RNA structures, RRI can vary greatly in sequence, size, structure and function, and we present a few known examples below.

**snRNA**

Previously mentioned as structured catalytic non-coding RNAs, small nuclear RNAs (snRNA) involved in splicing also undergo RRI within the spliceosome in eukaryotes. Multiple snRNAs contained within ribonucleoproteins binding specifically to the donor site, branch point and acceptor sites on the mRNA sequence [114]. Multiple interactions between the snRNAs also form a complex structure that enables the splicing to occur, consisting of short intermolecular basepairs [115].
snoRNA

While only a few dozen snRNAs exist in the cell, more than 200 unique small nucleolar RNAs (snoRNAs) are present in cells, and are one of the largest groups of trans-acting ncRNAs currently known [116]. Split into C/D box and H/ACA box RNAs, they bind to specific sites on the unprocessed rRNA transcript to guide RNA methylation (C/D box) or pseudouridylation (H/ACA box) events (Figure 1.7). Found in all branches of life, snoRNAs are essential for the proper functioning of the ribosome [116].

C/D box and H/ACA box snoRNAs each have short consensus sequence motifs and specific RNA structures, recognized by proteins and contained within RNPs. The unique sequences of the snoRNAs, act as adaptors allowing catalytic RNP complexes to be targeted to specific rRNA modification sites [116, 117].

Small ncRNAs

Perhaps the most recent and well-known example of functional RRI found in eukaryotes, are microRNAs and small interfering RNAs involved in RNA interference (Figure 1.7, bottom). MicroRNAs (miRNA) are short RNAs roughly 22 in length that act as adaptors to target protein complexes to specific mRNA sites [119]. Transcribed as a long hairpin, the pre-miRNA hairpin is cleaved by RNase proteins Drosha and Dicer in the nucleus and cytoplasm respectively, and eventually loaded in a RNP complex known as RNA-induced silencing complexes (RISC). The RISC can then specifically basepair complementary binding sites on mRNAs, causing translation repression or degradation [119, 120].

Small interfering RNAs (siRNA) are a highly similar system, also incorporated into the RISC, but with RNA sequences originally from exogenous sources, such as viruses in the cytoplasm. Dicer processes foreign transcripts into short sequences of roughly 21 bases, loads it into RISC, binds to complementary sequences and trigger RNA cleavage and degradation [119, 121]. A third class of small ncRNAs are Piwi-associated RNAs (piRNA), which are 24 to 30 bases long, go through a related but unique set of processing enzymes to target and suppress transposable elements in germline cells [119].

In all three types of these small ncRNAs, the RNA acts as an adaptor to specifically bind intended targets while the protein complex carries out the function. In miRNA and siRNA, the first seven or so nucleotide have been found to be seed regions which are required to perfectly bind with the target, although a sequence having a perfect complement seed region is not necessarily a target [122]. Plants have a similar pathways for miRNA and siRNA, but involve different processing
Figure 1.7:

Three common RNA-RNA interactions from a review by Meyer [118]. Part of a ribonucleoprotein complex C/D box snoRNAs and H/ACA snoRNAs bind to specific rRNA sites to enable highly specific RNA base modifications. The snoRNAs adopt highly specific intramolecular structure in addition to forming intermolecular interactions with rRNA. In contrast, miRNA does not adopt any complex structure, and form a highly complementary binding with targets sites on mRNA.


Small RNAs

Small RNAs or sRNAs, are non-coding regulatory RNAs found in bacteria, shown to bind to the translational start sites of mRNAs, controlling the stability and translation of their targets (Fig-
Figure 1.8:

Example RNA-RNA interaction between bacterial small RNA RyhB and sodB mRNA modulated by the Hfq chaperone protein. The sodB mRNA start codon (green) is normally protected in a small stem-loop (left), but becomes exposed when Hfq binds the RNA causing a structural change. The exposure of the sodB start codon allows the sRNA RyhB to bind, stopping translation. The new RNA-RNA complex becomes translationally inactive and susceptible to degradation.


Used with permission from John Wiley and Sons.
1.3 Solving RNA structure experimentally

1.3.1 Primary structure

The primary structure or sequence of the RNA can be experimentally determined by the sequencing of complementary DNA using classical Sanger or modern high-throughput technologies [27, 125]. As RNA is typically too unstable to sequence directly, it is common practice to obtain the complementary DNA through the usage of reverse transcriptase. With the DNA sequence of interest, di-deoxy or Sanger sequencing [126] can be done manually on a gel for small-scale experiments. For larger experiments, various commercial technologies and protocols are available for fast and cost-effective large-scale experiments [127].

1.3.2 Secondary structure

Rough secondary structure information can be obtained via RNA probing and footprinting, involving chemical modification and enzymatic cleavage of the structure, followed by subsequent identification of affected bases via sequencing of the treated RNA [50, 128]. A concrete example is the usage of the RNase V1 ribonuclease (discovered in Caspian cobra venom [129]), which specifically targets and cleaves helical basepaired structures. After a controlled incomplete V1 digestion of a homogeneous solution of the structure of interest, analysis of the products and determining cleavage sites can give a rough estimation of bases that were paired but not their basepaired targets [50, 130]. Enzymes that target unpaired bases include RNases I, T1 and A, S1 nuclease, and chemicals such as DMS (targets adenosine and cytosine), CMCT (uridine), kethoxal (guanine), NMIA (unbiased) and Pb$^{2+}$ (unbiased cleaving). The nucleases cleave RNA sites as expected, while the chemicals typically modify specific unpaired nucleotides in addition to reaching locations sterically inaccessible by the larger enzyme complexes. For reagents that target unpaired structures, the absence of reactions can mean that the regions is paired, but may simply be inaccessible due to structural location or the presence of other macromolecules. Depending and reagents and protocol, RNA structure probing can be done in vitro and in vivo (DMS, Pb$^{2+}$), and coupled with high-throughput sequencing techniques for whole-transcriptome RNA structure information [50].

1.3.3 Tertiary structure

The gold standard for structure determination is solving the tertiary structure, with the most common methods for RNA being X-ray crystallography [131] and nuclear magnetic resonance (NMR) spectroscopy [132]. X-ray crystallography requires the non-trivial task of crystallizing a purified
sample of the RNA/RNP of interest, but once done, enables atomic resolution of large structures such as the ribosome [57–59]. In NMR spectroscopy, no crystal is required, and a homogeneous liquid solution of the RNA of interest is sufficient, although the resolution (atomic clarity) of the structure decreases as input sequence increases, with high resolution structures limited to sequences of 50 bases or less [133]. Other methods include small-angle X-ray scattering (SAXS) [134], which uses X-ray scattering like X-ray crystallography and does not require a crystal, but does require the availability of one of the limited synchrotron beam facilities globally [135]. Cryo-electron microscopy (cryo-EM) has also been used to determine external shapes of RNA structures, done by the rapid freezing of a thin film of homogeneous molecules in solution, followed by visualization via an electron microscope and image processing [136, 137]. Finally, a recent alternative to the standard NMR process called solid-state NMR (ssNMR) has been demonstrated to be applicable to solving RNA structure [138], does not require a crystalline sample and can address molecules far beyond the size possible for standard solution-based NMR [138].

1.3.4 Caveats

Depending on the technique, the RNA may need to be modified, such as removing highly unstable regions to help crystal formation, or isolating a subunit of a larger structure to fit within experimental limitations. In addition, the need to extract, purify, and amplify the RNA of interest prior to structure determination means that the structure may not be representative of the functional structure in vivo. This is especially true if the native structure of RNA depended on specific RNA-binding proteins or ATP-dependent helicases [139] or is in reality one of many possible structures assumed in cell such as in the case of riboswitches. In vitro folding often occurs with fully synthesized molecules, purposely purified, amplified, and made to fold in a controlled solvent medium. In contrast, in vivo folding typically occurs during transcription (i.e. cotranscriptionally), whilst the nascent transcript emerges from the polymerase into the cellular milieu [140–142]. The incremental addition of bases during transcription means that local structures can form on the emerging 5'-end, which may induce or prevent the formation of specific basepairs with the 3'-end when it later emerges [141–143]. Other biomolecules like proteins and RNAs in the cell can also bind and process RNAs cotranscriptionally, adding another layer of complexity to RNA folding in vivo [144].
1.4 Computational prediction of RNA basepairing

1.4.1 RNA secondary structure prediction

While solving the tertiary structure remains the gold standard for studying RNA structure, in practice studying novel functional RNAs on the level of secondary structure basepairs is often a highly productive alternative to waiting for a tertiary solution. In many of the examples listed for functional RNAs previously, the tertiary structure remains unsolved, with many of those that have solutions coming decades after initial discoveries. Due to the hierarchical folding of RNA, functionality and structures solved on the basepair level are almost always present and applicable to the tertiary structure in contrast to proteins [49]. As basepairing rules are consistent and the main stabilizing force in many functional RNA structures, the prediction of RNA secondary structure through basepairing rules remains an important tool for experimentalists.

Due to the time and cost associated with determining the secondary and tertiary structure of RNAs experimentally, efforts have been made in the development of computational approaches to predict RNA structures \textit{in silico} given a primary sequence since the 70s. Due to the hierarchical nature of RNA structure formation, the accurate prediction of secondary structure is an active area of research required for the accurate prediction of tertiary structure [49]. Generally, computational approaches to predicting RNA secondary structure (henceforth referred to simply as \textit{structure}) can be classified as those that are energy-based and those that are comparative or evolutionary-based as discussed below.

1.4.2 Energy-based predictions

Conceptualized in the early 70s, energy-based methods operate on the assumption that the most stable structure taken by an RNA sequence at thermodynamic equilibrium, is the unique functionally relevant one [145]. The RNA structure is imagined to be built from individual components such as stabilizing basepairs and unstabilizing unpaired loops, bulges, and ends, each having a unique energy contribution which is summed to obtain the overall energy. The two challenges are then defining the set of components and the energy contributions or the energy model, and obtaining an efficient algorithm to explore all relevant structures.

Energy models have gone through various revisions, differing due to experimental setup and structural elements, with the most commonly used set referred to as the Turner model [146]. Most importantly, it was found that using the energy contribution of basepair stacking (two basepairs stacked together) gave much more accurate predictions than energy contributions from single base-
pairs. Recently, another set of parameters was also determined by Andronescu et al., by using machine learning to reverse engineer energy contributions for structural elements given solved structures with known overall energy levels [147, 148].

Early developments in finding an algorithm to efficiently explore the structure search space begin with bruteforce combinatorial approaches starting in 1975 [149], which do not scale well past several hundred basepairs. A breakthrough came a few years later with the usage of recursive **dynamic programming** algorithms [150] which can easily determine the single best structure for sequences up to kilobases in length. When combined with stacking energy models by the late 80s [151], the modern RNA folding “Zuker algorithm” came into being, and has been the dominant algorithm used for RNA secondary structure prediction.

The most popular energy-based methods—such as Mfold [152] and RNAfold [153] from the Vienna RNA Package—are so-called minimum free energy (MFE) algorithms that utilize dynamic programming, guaranteed to determine the structure with the lowest energy in a time and space efficient manner [154, 155]. Such algorithms take advantage of the fact that a stable overall structure is composed of stable sub-structures, and thus the algorithm can safely ignore highly unstable sub-structures to save on computation and re-use calculations for sub-structures reoccurring in multiple candidate solutions.

While efficient and effective, in practice, the accuracy of predicted MFE structures fall as the length of predicted sequences increase beyond a hundred bases, attributed to a few reasons [156]. Pseudoknotted basepairs are simply ignored, as including pseudoknots means that a structure cannot be split into independent discrete substructures, violating a required property for fast dynamic programming algorithms. Discussed in more details in the following chapter, pseudoknots are a prevalent structural motif formed by non-nested basepairs, found in nearly every organism with many examples found in functional RNAs [157]. Algorithmically, the classical pseudoknot-free MFE methods run in $O(n^3)$ time and $O(n^2)$ memory, where $n$ is the sequence length [152]. Without resorting to heuristics, pseudoknot MFE methods increase the complexity to $O(n^6)$ time and $O(n^4)$, quickly making it intractable for longer sequences of interest [158].

Current energy models fail to account for the complexity of folding in vivo, effectively simulating folding in vitro. Given the potential kinetics and interactions during folding, the theoretical MFE structure may simply not form due to structures being trapped in metastable structures during folding or as a result of in vivo interactions [156]. In attempts to alleviate this, methods that explicitly simulate kinetic folding [159–161] and incorporate cotranscriptional effects into the standard MFE energy model [162] have been developed.
1.4.3 Evolutionary-based methods

The comparative approach to structure prediction is based on the observation that if an RNA and its structure are important for some biological function, they are likely to be conserved, with tRNA and rRNA being the classical examples [163]. Thus, given a set of aligned homologous RNA sequences (typically determined by primary sequence conservation), we should be able to identify its structure using signals of conservation [163]. The key signal of basepair conservation are so-called covarying bases that are the result of compensatory mutations. These bases appear as two positions in the multiple sequence alignment, where all base form valid basepairs in all species, but the primary sequence differs between species, hypothesized to arise from a mutation on one side that negative impacts the structure followed by a compensatory mutation in the other side to restore the basepair [164]. Computationally then, the idea is to calculate the correlated evolution (e.g. covariation) between all potentially basepaired alignment columns, and return the set of basepaired positions that show significant non-random covariation.

Early comparative algorithms implement the idea above just as described—computing covariation scores in all possible positions pairs—but ignore the phylogenetic distance between the species from which the RNA sequences are derived, and return a set of individual basepairs. These, however, may contain conflicting (i.e. mutually incompatible) basepairs that cannot be combined into a valid RNA secondary structure that could form in real life [165]. Modern implementations utilizing phylogenetic stochastic context-free grammars (phylo-SCFG) resolve both problems by scoring mutations between alignment columns of sequences as a function of phylogenetic distance, and using dynamic programming to determine a pseudoknot-free structure optimizing the overall conservation score [166]. Scanning versions of phylo-SCFGs [167] have been successfully applied to discover conserved ncRNAs in vertebrate [168] and insect [169] whole-genome alignments. Given a multiple sequence alignment of sufficient quality and divergence, comparative (and hybrid) methods have been shown to significantly outperform MFE methods in terms of sensitivity and specificity of known basepairs [170]. However, given alignments that fail to properly align conserved structures severely impact the performance of comparative RNA prediction tools, which in practice limits the applicability of the tools depending on the sequencing, homologs, and conservation of a sequence of interest. Described as an “chicken-and-egg” problem, it is difficult to predict a structure without a good structure-based alignment, but obtaining a good structure-based alignment is difficult without knowing the structure beforehand, as basepair covariation often leads to primary sequence divergence [171]. Certain tools that simultaneously fold and align structures to alleviate this problem have been developed [171–174], but rely heavily on heuristics to remain
tractable as input lengths increase.

1.5 Objectives and contributions

The remainder of my thesis consists of three chapters, primarily composed of four published manuscripts. Having described RNA structure determination and prediction in this chapter, the following Chapter 2 focuses on the visualization of RNA secondary structures essential to explaining and analyzing RNA structures. In addition to discussing common forms of RNA visualization, we make our own contributions in the form of the R package R4RNA for visualization secondary structures with sequence alignment information. In Chapter 3, we discuss the prediction of RNA-RNA interactions, highly related to RNA structure prediction algorithms just described. We summarize the current state-of-the-art for RRI prediction tools, and evaluate the performance of select tools on the largest experimentally validated dataset collected to date. Finally, Chapter 4 conducts an in-depth review of some outstanding issues in computational RNA structure prediction, especially those relating to in vivo effects and cotranscriptional folding touched upon in this chapter.
Chapter 2

Visualization of RNA Basepairs and Alignments

This chapter outlines the foundational software code base that was developed to facilitate the entirety of the research I conducted. Termed R4RNA, this package consists of a collection of functions in the programming language R. The package itself can roughly be split into three function sets. The first set of functions deal with the reading, writing and converting the multiple text formats for RNA basepairs, essential for reading the unique output formats of various tools. The second set of functions deal with the statistical analysis of basepairs, serving as the basis for much of the numerical analysis in my research. The final set of functions are responsible for the visualization of basepairs, both for publication quality figures and also ad hoc graphs for sanity checks, debugging, interpretation, and analysis.

This work has been published as one of the only RNA analysis packages in R. We also host a web server version front-end of the visualization component, also published as part of a larger collection of Meyer Lab web servers:


While the format input/output and statistical parts of the package are essential and highly useful, they are not of particular intellectual novelty. The manual listing function names and details and vignette containing examples usage released with the R package are available as part of the Appendix of this thesis.
The programming language or R was selected due to its high adoption in the bioinformatics community due to the presence of the Bioconductor repository for open source computational biology and bioinformatics packages [175]. The open-source nature of R also allows users free and easy access to the environment required to run our package and accompanying tools. Specific to this project, the interactive console native to R allows for quick responsive results when computing RNA structure and visualizing figures for ad hoc analyses. R has powerful graphing abilities allows users to easily combine our plots with existing R graphs, as seen in a recent work combining our graphs with chemical probing data [176].

The remainder of this chapter focuses on the contributions that this package brings to the field of RNA visualization.

2.1 Visualizing RNA structure

As detailed in the introduction, RNA structure can be described as primary, secondary, and tertiary at the molecular level. As direct observation of RNA molecules is not feasible with current technologies, the next best option is to create graphical representations of RNA molecules for study and analysis. Rather than three-dimensional models of RNA at a scaled-up size however, these figures are often more abstract, conveying the defining features of primary, secondary, or tertiary structures.

2.1.1 Primary structure

Molecularly, the primary structure describes the linear sequence of covalently bonded nucleotides. When studying and visualizing at this level of abstraction, the relative spatial coordinates and identity of individual atoms are ignored. Since the four common RNA nucleotides can be unambiguously identified, primary structure can represented as a string of text characters, typically consisting of A, U, G and U, corresponding to the four RNA nucleotides. Also referred to simply as the RNA sequence, this string of nucleotides does not need to be in a straight line, which is often the case in secondary structures.

2.1.2 Secondary structure

The secondary structure of RNA describes all basepairs in the primary structure in addition to hydrogen bond basepairs forming between the nucleotides. Again, the relative spatial coordinates of atoms are not considered at this level of abstraction, thus the only new information is the identity of the pairs. In its most abstract form, the secondary structure can be described as a primary
structure sequence, accompanied by a list of paired positions or tuples. Each tuple is a pair of unordered numbers, with each indicating a specific RNA identified by its numerical position in the sequence, and the entire tuple indicating that the two basepair.

Given a tuple or basepair, based solely on the identity of the paired nucleotides, it can be termed canonical if it forms a stable Watson-Crick or wobble basepair, or non-canonical for any other combination of nucleotides. Given two basepairs, depending on their positions, they can be described as nested basepairs if the positions of one pair falls completely within the range of another. Consequently an unnested basepair is then any basepair that is not well nested, with only one of its two positions falling within the positional range of another basepair. Physically, a series of well nested basepairs results in a straight ladder-like stack of basepairs, that often forms a helical structure or helix. The helix, also referred to as a stem, ends in an unpaired segment we refer to as the loop due to limitations in RNA flexibility. Unnested basepairs form pseudoknots which physically occur when the loop of a helix binds with an unpaired region on the same molecule, at a position beyond the stem that forms it. Technically, a pseudoknot can be defined as two pairs with positions $i : j$ and $i' : j'$ whose relative positions are in the order $i < i' < j < j'$. Finally, depending on how the list of basepair tuples was constructed, other anomalies can also occur, such as incompatible basepairs, which describe two basepairs that overlap at one or more position, making them mutually exclusive in reality, as a single base cannot form a canonical basepair with more than one other nucleotide. A specific type of incompatible basepairs are duplicating basepairs where both positions are identical.

For some programs, some of these basepair types are a persistent challenge in secondary structure prediction and planar visualization [177]. In the classical “stem-loop” diagrams of displaying secondary structures on a plane, pseudoknots often result in undesired overlapping elements (Figure 2.4), which require stretching and manual adjustment of stems to display [178]. In contrast to primary structure, many different and unique methods of displaying secondary structure information exist in various publications.

### 2.1.3 Tertiary structure

The tertiary structure of RNA describes the full three-dimensional structure of the molecule, specifically the relative spatial coordinates of all the atoms. Visually, this is often represented as a three-dimensional model, with different levels of abstraction: from figures showing balls of electron cloud for each atom, to more abstract figures showing just the backbone and secondary bonds in three-dimensional space. In reality, this is still highly abstract, as it fails to capture the
2.2 Types of RNA secondary structure visualization

Due to experimental and computational limitations, it is highly unlikely for accurate tertiary structure information to be available for the majority of RNAs, especially those newly discovered. The next best option is to work on the level of secondary structure, knowledge from which will typically be directly applicable to the full tertiary structure if it is ever determined. Thus, a significant amount of research has been put into the generation and display of RNA secondary structure.

2.2.1 Dot plots

As secondary structure is defined as a list of tuples, after of a textual list of numbers, the next simplest form would arguably by a dot plot [179, 180]. A dot plot consists of a $N \times N$ grid for an RNA sequence of length $N$, and a filled in cell or “dot” at $x,y$ denotes the existence of a basepair between positions $x$ and $y$ (Figure 2.1). While not referred to as dot plots, examples are seen as early as 1971 [145], used to predict secondary structures in RNA. While not the most visually aesthetic at times, it does have the benefit of being able to show mutually exclusive basepairs which would not be able to form simultaneously in reality. In popular packages such as the Vienna RNA Package [181], dot plots are often used to display the results of suboptimal predictions, with each dot displaying the probability that a basepair will form instead of a discrete all or nothing solution.

2.2.2 Circle and linear diagrams

Another early method of visualizing RNA secondary structure is the circular format, seen in one of the earliest works of structure prediction in 1978 [183]. In a circle diagram, the primary sequence is laid out in a circular pattern, and a chord is drawn between two positions representing a basepair between them (Figure 2.2).

A highly related format is the linear diagram, which is effectively a circular diagram that has been unrolled resulting in the primary sequence being laid out in a straight horizontal format (Figure 2.3). Whereas basepairs were mathematical chords in the circular format, they are often seen in the form of rectangular arcs [184] or elliptical arcs [185, 186]. Regardless of exact shape, the arc represents a single basepairs, with the two ends connected to the basepairs on the primary sequence that basepairs in the structure.

For both circular and linear plots, well-nested and unnested basepairs are easily distinguish-
able by the presence of non-overlapping and overlapping arcs/chords, respectively. Conflicting basepairs can also be easily displayed, with two arcs connecting to a single base.

### 2.2.3 Stem-loop diagrams

The most common computationally generated method of visualizing secondary structures are stem-loop diagrams, showing non-pseudoknotted structures in a planar format. Basepaired regions are visualized as ladder-like stems, and unpaired regions are loops, bulges, and loose-ends enclosed, within, or flanking these stems, respectively (Figure 2.4 & 2.5). Manually created stem-loops are seen in the earliest works in RNA prediction [187] in the early 60s, with the automated visualization methods emerging two decades later [188]. Modern implementation of stem-loop visualization
Figure 2.2:
Circular diagram of the Cripavirus IRES by VARNA [186], the primary structure is seen as a circular line, running 5’ to 3’, start at the bottom going clockwise, with each basepair displayed as a chord connecting two positions. Pseudoknots are easy to determine by the intersecting chords.

Figure 2.3:
Linear diagram of the Cripavirus IRES by VARNA [186], the primary structure is seen as a horizontal line, running 5’ to 3’, left to right, with each basepair displayed as an arc connecting two positions. Pseudoknots are easy to determine by the intersecting arcs.
Figure 2.4: Stem-loop diagram of the Cripavirus IRES by VARNA [186], showing basepaired stems and unpaired loops, bulges, and ends. Basepairs connecting loop to loop across the structure are unnested pseudoknoted basepairs, which can be confusing to display on this format.

diagrams are highly customizable, require minimal user-adjustment, and can even visualize pseudoknots with aesthetically pleasing results [178, 185, 186]. Conflicting and duplicating basepairs however, are not possible to display in a single diagram.

2.2.4 Other diagrams

Besides the four types of visualization methods listed, many other formats exist, some more common than others. The dot-bracket or Vienna format is an extremely common method of digitally storing basepair information [153], which can also be easily read to determine secondary structure. In the dot-bracket format, a string consisting of text characters the length of the primary sequence is written, with the period character (dots) used in positions of unpaired bases, and pairs of matching brackets used where basepaired positions exist (Figure 2.7, SS_cons row at alignment bottom). Unless multiple types of bracket characters were used, it can be impossible to distinguish between nested and unnested basepairs.
Figure 2.5:
Variant of the Cripavirus IRES stem-loop diagrams in Figure 2.4 by VARNA [186], trading aesthetic quality for informational clarity.

One other format outputted by the popular Vienna web server are mountain plots [181], initially defined in 1984 [190] (Figure 2.6). These show the primary sequence in a linear format on the x-axis, while the y-axis displays the number of enclosing basepairs (i.e. how deeply nested a basepair is). Starting at the first position, as you increase in position along the x-axis, you increase height on the y-axis whenever a basepair starts, descend when a basepair ends, and remain at the same height when unpaired bases are encountered. Thus, matching slopes at the same height correspond to helices and plateaus correspond to unpaired loops at the end of helices or bulges within helices.

More abstract format exist, but have not seen wide adoption such as trees [191] and graphs [192] in the strict mathematical sense.

2.2.5 Conserved RNA structure

For any experimentally determined or theoretically predicted RNA structure, a method of evaluation is to analyze the degree of structure conservation between homologous sequences [193]. Strong evidence for RNA structure conservation are pairs of compensatory mutations that retain the basepairing ability, but change the basepairing nucleotides (covariation). A quick method of visually surveying the quality of a given multiple sequence alignment and a corresponding secondary structure prediction is to highlight covarying pairs of alignment columns [194] (Figure 2.7). Finally, like standard gene prediction, identifying regions of high primary sequence conservation can also be useful in understanding the potential functional roles of different parts of the RNA
Figure 2.6:
Mountain plot (left) output by the RNAFOLD web server [181] for the predicted centroid [189] structure of the Cripavirus IRES sequence (right). Three lines correspond to the minimum free energy structure (red), the thermodynamic ensemble of RNA structures (green), and the centroid structure (blue). The entropy graph below shows how different the ensemble structure is from the MFE, indicating how likely the MFE basepair will not form at a given position. Colours on the stem-loop structure indicate basepairing probability from 0 (purple) to 1 (red).

During the development and evaluation of the comparative helix prediction method TRANSAT [195] in our research group, we had to develop a new method for RNA secondary structure visualization that was able to do the following:

1. Show conflicting basepairs (i.e. basepairs involving the same sequence position)

2. Display primary sequence conservation and basepair covariation with respect to the multiple sequence alignment

3. Simultaneously show two different structures (e.g. from different sources of annotation or prediction)

4. Be aesthetically pleasing and intuitive to grasp

sequence.
Figure 2.7: A screenshot of RALEE, the RNA ALignment Editor in Emacs [194]. Columns highlighted in the same shade represent stem-forming basepaired regions, with gaps in the highlighting showing loss of basepairing mutations. The structure is seen at the bottom, shown with matching brackets as basepairs and dots as unpaired regions.


Our requirement to visualize conflicting basepairs rules out all formats except circle, linear and dot plots. Finally, the additional need to simultaneously show several structures in conjunction with a corresponding multiple-sequence alignment, led us to choose a format where the RNA sequence of interest is shown along a straight line.

While various powerful visualization programs already exist, only a few are actively supported and visualize RNA secondary structure in a linear fashion [185, 186]. Of those that do, they lack the features that we require, as they were not designed to handle conflicting basepairs nor display multiple sequence alignments simultaneously with a structure. Finally, the need to create
such diagrams in a high-throughput and scripted manner, rather than restricted by a graphical user interface, made their adoption and modification difficult [185] [186].

In the following, we present a highly modified and new method employing a linear format which we call arc diagrams [196] to fulfill our above four requirements. In addition, we provide a web server R-CHIE which accepts four common secondary structure formats and secondary structure for the quick visualization of data with our method to generate publication quality figures. For further customization and local use, we also make a corresponding R package [197] called R4RNA available at www.e-rna.org which leverages the graphical and computational framework of the interactive and easily-scriptable language R.

2.3 Materials and methods

2.3.1 The R-CHIE web server

Part of the larger Meyer Lab web server of tools and methods at www.e-rna.org, the R-CHIE web server is located at http://www.e-rna.org/r-chie, which provides a simple interface for generating six different types of arc diagrams automatically with instructions and examples, accessible by all major browsers. Descriptions and usage of the six different types of diagrams are as follows:

Single Structure

This is the most basic type of arc diagram, essentially identical to the typical linear diagrams observed in other publications, with the exception of much more powerful graphical options (Figure 2.8). This arc diagram shows the RNA sequence of interest drawn as a horizontal line from 5’ to 3’, left to right, with arcs, drawn above the horizontal line. Each arc depicts a basepair of the RNA structure and connects the respective sequence positions involved in that basepair.

For predicted structures, it is not uncommon for individual structural features such as helices or basepairs to be assigned individual scores such as energetic contributions [180] or statistical significance [195]. In order to retain and visualize this valuable information, our method can assign different colours to individual arcs according to their corresponding scores, using palettes obtained from ColourBrewer [198], or those specified by the user. Alternatively, colouring arcs can also be done completely manually independent of value, e.g. when certain basepairs or structure features such as pseudoknots are to be especially highlighted. In addition, basepairs can also be filtered by their scores and a lower or upper threshold value can be imposed.
Figure 2.8: An example of a single structure arc diagram, showing the solved structure Cripavirus Internal Ribosomal Entry Site (family RF00458 from the RFAM database [182]). The horizontal line represents the nucleotide sequence with positions numbered in the background. Each arc represents basepair between two positions.

Double Structure

A double structure arc diagram is obtained by starting with a single structure diagram, and drawing a second structure below the horizontal sequence line (Figure 2.9). Any colouring and filtering options can be applied to the top and the bottom structure jointly or separately.

This type of arc diagram is especially useful when comparing two – perhaps radically different – alternative structures for the same sequence. It is also useful for comparing two similar structures, e.g. derived from two different structure prediction methods or comparing a predicted to an experimentally validated structure.

Overlapping Structure

The type of figures seen in the TRANSAT paper [195] allow us to quickly and visually evaluate the performance of a structure prediction method by comparing the predicted to the known structure (Figure 2.10). In order to do this best, we aim to simultaneously visualize the sensitivity and the positive predictive value, i.e. specificity, of the prediction method.

Similar to the double structure plot, arcs are seen both above and below the horizontal sequence line. Instead of showing all arcs corresponding to one structure above the horizontal line and all arcs corresponding to the other structure below, however, the first is interpreted as a predicted structure, the second as reference i.e. known structure, and the two structures are overlapped. For this, the algorithm identifies all predicted basepairs that overlap with those of the known structure (i.e. a true positive in the performance evaluation), and draws corresponding arcs above the line,
Figure 2.9: An example of a single structure arc diagram, showing the solved structure of Cripavirus IRES (bottom) and TRANSAT predicted basepairs (top). Predicted basepairs are coloured by P-value.

coloured by the score of the basepair in the predicted structure. Any predicted basepair that is not part of the known structure (i.e. a false positive) is drawn below the sequence line, also coloured by its score. Any known basepair that was not predicted (i.e. a false negative) is drawn above the sequence line in black. Basepairs that are neither part of the known nor the predicted structure (i.e. false negatives) are not shown at all.

With a single glance, this type of diagram shows both the sensitivity and specificity of the structure prediction and readily highlights new basepairs that are not part of the known structure that may warrant further investigation.

Creating two overlapping structures from predictions of two different algorithms against the same known structure and juxtaposing the resulting diagrams is also an interesting method of comparing and highlighting the differences of two algorithms, as was done extensively in the TRANSAT paper [195].
Figure 2.10: An example of a overlapping structure arc diagram, overlapping the TRANSAT predicted structure and the solved structure the Cripavirus IRES. The structure shown above the horizontal sequence is the known structure in black, coloured by p-value if correctly predicted by TRANSAT. The arcs below the line represent novel basepairs predicted by TRANSAT not found in the known structure. Such a diagram can give a qualitative description of a predicted structure’s performance, where high sensitivity would result in a high proportion of top helices being coloured, and high specificity would result in a majority of helices above the line.

Single Structure Covariation

Adding a multiple sequence alignment beneath a single structure arc diagram provides a powerful means of displaying both the secondary structure and corresponding evidence for basepair conservation and covariation (Figure 2.11 & 2.12). As comparative methods have been shown to be the state-of-the-art when it comes to predictive accuracy, sequence alignments are often used in RNA structure research. This type of arc diagram is especially useful for evaluating structures given an alignment and vice versa.

For this type of arc diagram, the arcs are drawn on top of the sequence line as usual while the
**Figure 2.11:** An example of a *single structure covariation diagram*, showing the solved structure Cripavirus IRES projected onto a multiple sequence alignment. Basepaired columns are coloured according to conservation status as indicated by the legend where **Conservation** is the absence of mutations, **(Double-Sided) Covariation** is a basepair where both sides have changed but remain valid, **(One-Sided) Covariation** is where only one side mutates but remains valid, **Invalid** basepairs have mutated to non-pairing nucleotides, and **Unpaired** and **Gap** are unpaired and gapped, respectively.

**Figure 2.12:** A variation of the plot shown in Figure 2.11, where species and basepairs have been labeled.

A multiple sequence alignment is shown below as a block of parallel black lines, each representing one sequence of nucleotides (with grey for gaps) from the multiple sequence alignment. Two alignment columns at the base of a single arc represent the two columns of basepairing nucleotides. The corresponding nucleotides are shown in green if they represent a valid canonical basepair, and
red if they are not. For bases coloured in green, if there is a compensatory mutation that differs from the most commonly observed basepair in that pair of alignment columns, it is highlighted in blue (dark blue for a double-sided mutation, light blue for a single-sided mutation), similar to existing programs [167, 194, 199].

Given a structure and a corresponding multiple sequence alignment, the web server automatically applies this colouring, allowing for a quick evaluation of how well (or poorly) the different structural features are supported by covariation and gap-patterns.

Given two different ways of aligning the same set of sequences, two arc diagrams of this type can also be used to highlight the effect that the alignment quality has on the corresponding structure prediction.

One small caveat is the technical inability of the covariation colouring to be displayed simultaneously for conflicting helices. When faced with conflicting helices, our algorithm makes a greedy decision to select and colour the first helix that it observes in the input. A user can therefore simply rank conflicting helices or basepairs in the input file to ensure that the most dominant features are being coloured (See dashed arcs in Figure 2.13).

Additional web server appearance options that users may adjust include displaying sequences as blocks instead of lines, including the nucleotide base on the block, and including the sequence descriptions left of each sequence. The specific colours used to highlight basepairs in the multiple sequence alignment can also be customized, and one may even completely ignore basepairs and colour the alignment based on nucleotides.

While the structure conservation for one basepair can be summarized in a single numerical value as done for some figures in the RFAM database [182], a coloured multiple sequence alignment as in these arc diagrams retains more detailed information. If desired however, although we offer the ability to colour arcs according to structure conservation, covariation, and percent canonical basepair.

**Double Structure Covariation**

Two multiple sequence alignment blocks, one for each structure, are inserted between the top and bottom arcs of a double arc diagram, highlighting the conservation and covariation for each structure (Figure 2.13).

An extension of the double structure arc diagram, the double structure covariation diagrams show not only the differences between two structures, but also allow the evaluation of the different basepairs in light of evolutionary evidence. The input multiple sequence alignment is dupli-
Figure 2.13: An example of a double structure covariation plot, showing TRANSAT predictions (top) and the known Cripavirus IRES structure (bottom). An extension of the single covariation plot, the only unique difference is the dashed basepairs, which indicate basepairs that conflict with other more highly ranked (according to P-value) basepairs and not displayed on the alignment.

To ensure that the colouring remains consistent, the determination of a covarying basepair is relative to the most commonly observed base pair in a given pair of basepaired alignment columns, as mentioned before. In the case where there is a tie for the most common base pair, priority is given to the basepair more commonly observed according to Structure Statistics from the Comparative RNA Web Site [200].
Overlapping Structure Covariation

A natural extension of the arc diagrams mentioned so far is to combine overlapping structure arc diagrams with covariation plots. These are similar to double structure covariation plots, but are drawn according to the same rules as overlapping structure arc plots (Figure 2.14).

Figure 2.14: An example of a overlapping structure covariation plot, combining the covariation plot with the overlapping structure diagram.

This type of diagram is of great use when evaluating new helices (drawn below the sequence line in overlapping structure arc plots) by providing evolutionary evidence (or lack of) for their existence. The same rules to resolving conflicting helices exist as outlined above for single structure covariation plots, as do the rules for ensuring consistent covariation plot colours.
2.3.2 Input

There are two main types of text inputs to the web server: those specifying secondary structures and those specifying multiple sequence alignments.

For RNA secondary structure, our method accepts most common output formats: dot bracket or Vienna format [201], MFOLD’s connect format [202], and Gutell’s bpseq format [200]. Additionally, we also accept a variant of Shapiro’s original region table [188] that we refer to as the helix format. This describes each helix as one line which contains the following fields: start position of outer basepair, end position of outer basepair, helix length (in terms of number of basepairs), helix score. Differing from Shapiro’s definition, we add a header line which includes the length of the sequence, along with any other comments one may want to retain, e.g. the primary sequence. This helix format provides an extremely compact means of representing complex RNA structures, and also allows for unambiguous specification of conflicting, pseudoknotted and overlapping basepairs.

For multiple sequence alignment, our program accepts standard FASTA format [203].

To control the appearance of the resulting figure showing the desired type of arc diagram, an standard options panel is available to automate and fine-tune colouring and filtering of basepairs on the fly.

2.3.3 Output

After jobs are submitted, they are queued in a dedicated computer cluster with up to 100 concurrent jobs processed from all our web servers. After generating the figure, which typically takes on the order of seconds, a static web page rendering the figure is displayed from which the figure can be downloaded as .png or .pdf format. To ensure reproducibility and help with customization, we also output the command sequence required to replicate the figure using a locally installed version of our program.

2.3.4 The R4RNA R package

The plotting functionality of the web server is driven by an R script built on top of an R package called R4RNA which we make available for offline and local use and which can be downloaded from http://www.e-rna.org/r-chie release for public use under the GPLv3 license.

Written in R [197] (which is freely downloadable at http://www.r-project.org/ for all major operating systems), the package is capable of producing the same plots as the server with a few interactive function calls, and allows for even more fine-tune control, automation, and customized diagrams, and output formats. This is especially convenient for Bioinformatics research groups that
can call functions of our R package within existing programs and analysis pipelines. Our software is well-documented, including a comprehensive manual and instructional examples in the vignette document, both of which are in the Appendix section of this thesis.

2.4 Results and discussion

We here present a new computational method for visualizing RNA secondary structures in conjunction with corresponding multiple sequence alignments which can either be used via a web server R-CHIE or offline and locally via a corresponding R package called R4RNA. Our method readily creates six different types of arc diagrams which cover numerous useful applications. These range from visualizing the evolutionary evidence for a given RNA secondary structure to comparisons of two RNA secondary structure and performance evaluations of RNA structure prediction methods. The key feature of all six types of arc diagrams is that details that are typically lost in a numerical evaluation are highlighted and can be visually interpreted in a straightforward and intuitive way.

Our method makes several major improvements with respect to existing methods that depict RNA secondary structures in a linear way. These include the colouring of structural features according to their score (e.g. free energy, p-value, log-likelihood), the joint display of an RNA structure with a corresponding multiple sequence alignment which highlights the evolutionary patterns that support the different structural features, and comparison plots which allow the quick visual inspection of sensitivity and specificity of a predicted structure with respect to a reference structure. In addition, all types of arc plots can display structural features that are mutually exclusive or would render the overall RNA structure pseudoknotted.

Our R-CHIE web server and the corresponding R4RNA R package can be freely accessed and downloaded from [http://www.e-rna.org/r-chie](http://www.e-rna.org/r-chie) In addition to several examples, we have also generated single structure covariation arc diagrams of all seed alignments in the RFAM database [182] which can be downloaded from our web page. Since our publication, our plots have been used to generate many published figures, in addition to being adopted by the latest version of RFAM as a visualization method for conservation information [204]. A few highlights include the display of riboswitch conformation covariation [205] (Figure 2.15), comparing two structures in light of chemical probing evidence [176] (Figure 2.16), and comparing multiple wild-type and mutant structures [206] (Figure 2.17).
Figure 2.15: The double covariation plot used by Zhu et al. [205] to display the structure and conservation of the two conformation of the SAM riboswitch.

Figure 2.16:
Double structure plot used by Rogler et al. [176] to show the structure of the RNA endoribonuclease RMRP as predicted by SHAPE chemical probing (blue) and previously published covariation-based results (orange). In the ideal scenario, SHAPE values correlate strong with unpaired regions and weakly with paired regions, which the authors believe to strongly suggest the existence of the top alternative structure.

Figure 2.17:

Single structure plots used in conjunction with a principal component decomposition scatter plot of suboptimal structures and stem-loop figures to show the results of mutants (purple and yellow) versus the wild-type structure (blue) of the human the Retinoblastoma 1 mRNA 5’UTR.

Chapter 3

Assessment of RNA-RNA Interaction Prediction Methods

In this chapter, I present a comprehensive assessment on prediction performance accuracy for currently available RNA-RNA interaction prediction methods. The chapter starts by describing the state-of-art in computational RNA-RNA interaction prediction, defines the specific problem of interest. I then proceed to evaluate the largest collection of existing tools, against one of the largest collection of experimentally validated interaction data. The work described here has been published as one of the only evaluation and benchmarks of its kind:


3.1 Introduction

A large percentage of the mammalian genome is transcribed into non-coding RNA (ncRNA) [31]. As these ncRNAs may play important regulatory roles in the cell, efforts have been made to functionally annotate these transcripts [32, 207]. Previous research on ncRNAs such as sRNA [208] and miRNA [209] have shown that the identification of RNA-RNA interactions (RRI) between candidate ncRNAs and their targets is a key step to understanding the role of the RNA. Identifying and validating these interactions experimentally however, can be slow and costly. To aid the identification of RNA-RNA interactions, a range of *in silico* methods have been proposed [118].

The prediction of RRIs can be viewed as a direct extension of RNA secondary structure predic-
tion, employing similar theories and algorithms. In both settings, solutions are obtained by determining the set of Watson-Crick and wobble basepairs that correspond to the functionally relevant structure/interaction. Specifically, given two RNA sequences consisting of nucleotides adenine (A), cytosine (C), guanine (G), and uracil (U), determine the optimal set of intermolecular hydrogen bond basepairs between the two sequences.

More complex versions of the problem exist, such as those solving the joint structure, consisting of both the intramolecular basepairs within a single sequence in addition to the intermolecular basepairs. There is also the highly related RNA-RNA target prediction problem, where given a single query RNA sequence, and a set of potential target RNA sequences, find the correct pairing target for the query RNA. Tools solving the basic RRI prediction problem are much more common than those tackling these variations, and correct prediction of the complex variations often rely on correctly predicting the RRI problem first. As such, we will focus on the basic intermolecular RNA-RNA interaction problem given two sequences. In contrast to many previous evaluations, we impose no restriction on the specific type or length of the input RNA, aiming to evaluate RRI tools in a general de novo scenario.

3.2 Algorithm strategies

We compare the predictive performance of 14 published computational methods (11 distinct program binaries) designed to predict interacting basepairs given two input RNA sequences. To better understand and compare these, we subdivide the RRI prediction algorithms based on their strategies into four types similar to those in other works [208, 210].

The first type concerns itself only with intermolecular basepairs, both during computation and also for the final predicted result. Such algorithms are typically the fastest, having no need to predict intramolecular basepairs that could interfere and restrict certain intermolecular interactions. Ignoring restrictions and interferences, however, is exactly why these tools may incorrectly predict certain interactions where the existing RNA secondary structure needs to be taken into account. Algorithmically, these types of tools usually derive the set of interacting basepairs that maximizes a certain value, commonly the stability of the entire interaction complex as quantified by the overall Gibbs free energy (ΔG) of stacking basepairs. We refer to these as “interaction-only” methods, RNA DUPLEX [181], RNAPLEX-c [211], RISEARCH [212], and GUUGLE [213] fall into this category. GUUGLE is unique amongst these tools, being the only one that does not compute Gibbs free energies to score optimal interactions, but instead returns all ungapped interactions above a user-specific length, which we include as an absolute baseline for predictive performance.
The second type of method predicts only intermolecular basepairs, but factors in intramolecular interactions during computation, addressing the weakness of the first type. These algorithms utilize the McCaskill partition function algorithm \cite{214, 215} on the single input sequences to predict the pairing likelihood of nucleotides at each position. Thus, the stability of the intermolecular interaction at a specific position is now affected by both the predicted stability of the stacking basepairs, and also how likely the position will be made inaccessible by existing intramolecular basepairs. We refer to these as “accessibility-based” methods which comprise RNAUP \cite{216}, INTARNA \cite{217}, and RNAPLEX-a \cite{218}.

The third type considers both inter- and intramolecular basepairs with restrictions during both computation and results, outputting in a joint structure. The most basic of these are termed “concatenation-based” algorithms, literally concatenating the two input sequences and running it through classical RNA secondary structure prediction algorithms such as Mfold \cite{152} and RNAfold \cite{153}. The main shortcoming of these methods stems from the classical RNA secondary structure algorithm’s inability to predict unnested basepairs or pseudoknots, which translates to the inability to predict interactions that form on interior loops in the joint structure. PAlRFold \cite{219} and RNAcofold \cite{181} fall into this category.

The fourth and final type is less well-defined and encompasses all non-concatenation methods that solve the joint structure, with little to no restrictions on interactions. The removal of restrictions often comes at the great expense of runtime performance, so these tools are typically restricted to relatively short input sequences. In this class, we have the program RActIP \cite{220}, made tractable for use on longer sequences by utilizing the technique of integer programming to optimize for runtime performance.

In addition to falling into one of the four categories, tools may optionally take multiple sequence alignments as input for each of the two input sequences. Based on successful RNA secondary structure prediction tools like PFold \cite{221} and RNAalifold \cite{222}, the addition of well-aligned and sufficiently divergent homologs provides additional information when predicting evolutionarily conserved basepairs. In theory, basepairs that are fully conserved or undergo compensatory mutations to retain the basepaired structure (i.e. covariation) are likely to be more functionally important than unconserved basepairs. RNAaliduplex \cite{181} is the multiple sequence alignment version of RNAduplex \cite{181}, classified as the first interaction-only type. The interaction-only and accessibility-based version of RNAplex can optionally take multiple sequence alignments as input, which we will denote as RNAplex-cA and RNAplex-aA, respectively. PETcofold \cite{210} belongs to the final complex joint structure category, given two multiple sequence alignment.
Table 3.1: RNA-RNA interaction tools evaluated with categories and features. Strategy indicates the broad strategy of the algorithm in terms of prediction and output, described in the Introduction. Suboptimal indicates whether the tool can return suboptimal results in addition to the minimum-free energy result. Conservation indicates whether it takes alignments as input. Interaction Length roughly describes the style of helices output, short helices ranging typically not surpassing a dozen or so basepairs, with long helices reaching up to several times of that in total basepair count. Local interaction are a single interaction with gaps and bulges typically no longer than a few basepairs, while global predictions may span the entire sequence, containing multiple instances of local interactions, separated by long regions lacking intermolecular basepairs.

### 3.3 Methods

#### 3.3.1 Energy-based RNA-RNA interaction prediction programs

Mentioned above, the programs used are summarized in Table 3.1, with more algorithmic details and exact settings to follow. On the table, in addition to splitting the tools into the four categories according to their strategy and usage of conservation, we also summarize whether they can output suboptimal results (instead of a MFE result) and the style of output they give.

Unless stated otherwise, inputs `target.fa` and `query.fa` consists of two standard FASTA files, each containing one header line followed by an RNA sequence.

Maximal interaction length for the sRNA-mRNA dataset was 60 and is set as such for many tools. This value is set to 25 for the snoRNA-rRNA dataset.
GUUGLE

GUUGLE [213], while not specifically designed to predict RNA-RNA interactions, can very rapidly recover all continuous helices even for very large input sequences. It creates a suffix tree out of the target sequence, which allows for the rapid lookup of complementary pairing locations when given a query sequence, with novel strategies to allow for the G-U wobble base. Its inability to recover helices with bulges and loops and the lack of an energy calculation makes it ill-equipped to find complex interactions, but we use it to serve as an absolute baseline for accuracy performance.

Version 1.2 obtained from https://bibiserv2.cebitec.uni-bielefeld.de/guugle?id=guugle_view_download

```
guugle -d 6 target.fa query.fa > output.txt
```
-d Obligatory option, minimum interaction length to be output

RNAPLEX-c

RNAPLEX-c is also part of the ViennaRNA Package 2.0, designed to be even faster than RNADUPLEX at predicting RNA-RNA interactions, ideally suited for genome-wide applications [211]. Algorithmically, this gain in speed is achieved by further simplifying the Zuker algorithm [152], restricting more loop types, loop size and approximating bulge and loop energies.

Part of the Version 2.1.9 ViennaRNA package obtained from http://www.tbi.univie.ac.at/RNA/

```
RNAplex -q query.fa -t target.fa -c 30 -l 60 -e 0 > output.txt
```
-c Per nucleotide extension cost, recommended settings from publication
-l Maximal length of interaction. 60 is minimum rounded known interaction length.
-e (Suboptimal toggle) Enables suboptimal mode, returning all results ΔG ≤ 0

RISEARCH

RISEARCH [212] is one of the most recent programs and has been shown to run a few times faster than RNAPLEX. Algorithmically, it reduces basepair prediction into a Smith-Waterman-like [224] local sequence alignment problem, with RNA secondary structure stacking energy calculation replaced by a dinucleotide scoring matrix lookup.

52
Version 1.0 obtained from [http://rth.dk/resources/risearch/](http://rth.dk/resources/risearch/). The latest Version 1.1 allows for thresholding by energy instead of score, which does not affect the results.

RIsearch -d 30 -s 0 -q query.fa -t target.fa > output.txt

-c Per nucleotide extension cost, recommended settings from publication

-s (Suboptimal toggle) Enables suboptimal mode, returning all results with a score less than 0

**RNAduplex**

RNAduplex is part of the ViennaRNA Package 2.0 [181], designed for the rapid prediction of RNA-RNA interactions. Algorithmically, it is a modification of Zuker’s classical RNA secondary structure algorithm [152], simplified to ignore intramolecular basepairs and branching structures, much like the miRNA-specific RNAHYBRID algorithm [225].

Part of the Version 2.1.9 ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download)

```
cat query.fa target.fa | RNAduplex -e 50 > output.txt
```

-e Suboptimal toggle Enables suboptimal mode, returning all results within 50 kcal/mol to the minimum-free-energy solution

**PairFold**

PairFold [219] is one of the earliest general tools for predicting the joint RNA secondary structure, which concatenates two input sequences and employs a slightly modified version Zuker algorithm [152] found in MFold. Special attention is paid to the concatenation site, which is contained within a “special loop” that is penalized for intermolecular interactions. Due to usage of the Zuker algorithm, it is unable to predict pseudoknotted basepairs for the concatenated sequence.

Part of the MultiRNAFold-2.0 package downloaded at [http://www.rnasoft.ca/download.html](http://www.rnasoft.ca/download.html)

Manuscript states that suboptimal results are available, but no settings to enable it were found in the exposed user interface.
pairfold query.fa target.fa > output.txt

Query and target sequences are not given as files, but as strings on the command line

**RNACOFOLD**

RNACOFOLD [223] is part of the ViennaRNA Package 2.0, takes two input sequences, concatenates the two, then runs a slightly modified version of RNAFOLD on the concatenated sequence, returning one joint structure with an overall stability score. Theoretically identical to PAIRFOLD, it too cannot predict pseudoknotted basepairs for the concatenated sequence.

Part of the Version 2.1.9 ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download)

concatenated_query_ampersand_target.fa | RNAcofold > output.txt

Input file required a single FASTA file with a single sequence consisting of the two input sequences concatenated and separated by an ampersand symbol.

**IntaRNA**

IntaRNA [217] can be seen as a direct improvement over RNAUP, reducing the runtime complexity with various optimizations, and introducing the concept of a seed region. Seed regions require interactions to contain a strongly complementary region, with user defined lengths and mismatch allowance. Designed specifically for predicting interactions between short sRNA query and longer mRNAs in bacterial systems, accessibility is computed over the entire query sequence using RNAUP and in a sliding window with RNAPLFOLD [153] over the target sequence.

Version 1.2.5 obtained from [http://www.bioinf.uni-freiburg.de/Software/#IntaRNA-download](http://www.bioinf.uni-freiburg.de/Software/#IntaRNA-download)

IntaRNA -w 140 -L 70 -l 60 -m query.fa -t target.fa -o > output.txt

- **w** Average the pair probabilities over windows of given size.

- **L** Set the maximum allowed separation of a base pair to span.

- **l** Max length of hybridized region, mainly for efficient computation

- **o** Enables detailed output
Window and length values set to be identical to RNAPLFOLD and RNPplex values seen above.

**RNPplex-a**

The updated version of RNPplex-c [218], which we term RNPplex-a can optionally take in an externally computed accessibility file output by RNAPLFOLD [181]. Accessibility profiles are computed for sliding windows by giving RNAPLFOLD FASTA files for both target and query sequences and converted to position and length-specific opening energies. Predicted interactions have a stability value formed by the summation of the hybridization energy, the opening cost on the query sequence, and opening cost on the target sequence.

RNPplex is part of the Version 2.0.7, while RNAPLFOLD is from ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download). Version 2.1.9 of RNPplex appears to have a bug that causes a lot of errors in the output file when accessibility is used, but is otherwise fine.

```bash
cat query.fa target.fa | RNAplfold -b -O -u 60 -W 140 -L 70
mkdir accessibility_dir; mv -f *_dp.ps *_openen* accessibility_dir;
RNAplex -q query.fa -t target.fa -a accessibility_dir -b -l 60 -e 0 > output.txt
```

- **-b** Output/input accessibility profiles in binary format
- **-O** Switch output from probabilities to their logarithms (approximate mean opening energy)
- **-u** Compute the mean probability that regions of length 1 to a given length are unpaired. 60 is minimum rounded known interaction length.
- **-W** Average the pair probabilities over windows of given size.
- **-L** Set the maximum allowed separation of a base pair to span.
- **-l** Maximal length of interaction. 60 is minimum rounded known interaction length.
- **-e** *(Suboptimal toggle)* Enables suboptimal mode, returning all results $\Delta G \leq 0$

Values for $-W$ and $-L$ were set relative to $-u$ using ratios that were obtained from the published recommended settings. In practice, FASTA sequence names had to be renamed from species names which were identical in the query and target file to two unique names so that would not clash in the FASTA-header-derived output filenames generated by RNAPLFOLD.
RNAup

RNAup [216] is one of the first accessibility-based methods, which treat RNA-RNA interaction formation as a two step process. The first is computing the probability that a region on the target sequence is unpaired (i.e. accessibility-profile), from which the energy required to open the site can be derived. This is followed by the hybridization step with the unpaired target site, potentially taking a penalty to stability if an inaccessible region needs to be opened.

Part of the Version 2.1.9 ViennaRNA package obtained from http://www.tbi.univie.ac.at/RNA/

\[
\text{cat query.fa target.fa | RNAup -b --interaction_pairwise > output.txt}
\]

-b Include the probability of unpaired regions in both RNAs

- --interaction_pairwise Activate pairwise interaction mode

A maximal interaction length settings -w was available, but enabling it to the expected number (maximal known interaction length) actually decreased performance and thus was not used.

Ractip

Ractip [220] has been shown to be the fastest method for simultaneously predicting the entire joint structure. A model of interactions and constraints are defined for general RNA-RNA interactions, and an optimal solution is solved for using integer programming. Like Petcofold, internal models allow it to predict complex structures such as kissing hairpins, not possible with concatenation-based methods.

Version 0.0.2 obtained from http://www.ncrna.org/software/ractip/

\[
\text{ractip query.fa target.fa > output.txt}
\]

No recommended settings were found, no alternative settings were attempted. Three probability settings and one accessibility model toggle were available.

3.3.2 Comparative RNA-RNA interaction prediction tools

Input files query_alignment.fa and target_alignment.fa are paired FASTA files, each containing the same number of FASTA sequences, with matching ordering of species.
**RNAPLEX-cA**

An updated version of RNAPLEX-c [218] can additionally take alignments as input, and like RNAALIDUPLEX, also incorporates a covariation score into the stability of the predicted duplex in the style of RNAALIFOLD. We shall refer to this as RNAPLEX-cA.

Part of the Version 2.1.9 ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download)

```
RNAPlex -q query_alignment.fa -t target_alignment.fa -c 30 -l 60 -e 0 -A > output.txt
```

- **c** Per nucleotide extension cost, recommended settings from publication
- **l** Maximal length of interaction. 60 is minimum rounded known interaction length.
- **e** *(Suboptimal toggle)* Enables suboptimal mode, returning all results \( \Delta G \leq 0 \)
- **A** Tells tool to compute interactions based on alignments

**RNAALIDUPLEX**

RNAALIDUPLEX is an alignment-based version of RNADUPLEX, also part of the same ViennaRNA Package 2.0. Using a similar technique as seen in RNAALIFOLD [222], RNAALIDUPLEX factors in basepair conservation information into energy-based predictions. For each basepair, in addition to the stability derived from the stacking energies, an additional basepair covariation score computed from the conservation of the basepair is added to the energy stability. Note that input files had to be converted to CLUSTAL format.

Part of the Version 2.1.9 ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download)

```
RNAaliduplex query_alignment.aln target_alignment.aln -e 50 > output.txt
```

- **e** *(Suboptimal toggle)* Enables suboptimal mode, returning all results within 50 kcal/mol to the minimum-free-energy solution

Instead of FASTA files, CLUSTALW files were required as input.
**RNAPLEX-aA**

RNAPLEX-a can additionally take input alignments, resulting in RNAPLEX-aA, using energy, accessibility, and conservation information in the style of RNAALIFOLD to make predictions.

RNAPLEX is part of the Version 2.0.7, while RNAPLFOLD is from ViennaRNA package obtained from [http://www.tbi.univie.ac.at/RNA/index.html#download](http://www.tbi.univie.ac.at/RNA/index.html#download). Version 2.1.9 of RNAPLEX appears to have a bug that causes a lot of errors in the output file when accessibility is used, but is otherwise fine.

```
cat query_alignment.fa target_alignment.fa | RNAplfold -b -O -u 60 -W 140 -L 70
mkdir accessibility_dir; mv -f *_dp.ps *_openen* accessibility_dir;
RNAplex -q query.fa -t target.fa -a accessibility_dir -b -l 60 -e 0 -A > output.txt
```

- **-b** Output/input accessibility profiles in binary format
- **-O** Switch output from probabilities to their logarithms (approximate mean opening energy)
- **-u** Compute the mean probability that regions of length 1 to a given length are unpaired. 60 is minimum rounded known interaction length.
- **-W** Average the pair probabilities over windows of given size.
- **-L** Set the maximum allowed separation of a base pair to span.
- **-l** Maximal length of interaction. 60 is minimum rounded known interaction length.
- **-e** *(Suboptimal toggle)* Enables suboptimal mode, returning all results \( \Delta G \leq 0 \)
- **-A** Tells tool to compute interactions based on alignments

Values for `-W` and `-L` were set relative to `-u` using ratios that were obtained from the published recommended settings. In practice, FASTA sequence names had to be renamed from species names which were identical in the query and target file to two unique names so that would not clash in the FASTA-header-derived output filenames generated by RNAPLFOLD.

**PETCOFOLD**

PETCOFOLD [210] is a hierarchical joint structure prediction tool that takes two FASTA alignments as input. In a two step pipeline, the program first predicts structures individually for each
of the two input alignments, thereby determining highly stable, conserved, and thus likely func-
tional, intramolecular basepairs using the PETFOLD [226] algorithm. This underlying PETFOLD
algorithm has been described to be the classical PfOLD [221] RNA secondary structure algorithm
with a full evolutionary model [227] but adds a weighted thermodynamic component generated
via the McCaskill algorithm using RNAFOLD with option \(-p\), not unlike the “accessibility” score.
These two alignments are then concatenated and likely intermolecular basepairs are then deter-
dined, with the likely basepairs from step one being prevented from binding in step two. This
process allows for intermolecular basepairs in loops (i.e. kissing hairpins), a structure that would
involve pseudoknotted basepairs that concatenation-based algorithms cannot handle.

Version 3.2 obtained from [http://rth.dk/resources/petcofold/download.php](http://rth.dk/resources/petcofold/download.php)

```
PETcofold -f query_alignment.fa -f target_alignment.fa --intermol --war
--extstem > output.txt
```

- `--intermol` Structure output of intermolecular base pairs
- `--war` FASTA format output
- `--extstem` Constrained stems get extended by inner and outer base pairs

### 3.3.3 Multiple sequence alignment generation programs

A subset of our tools are comparative and require high quality multiple sequence alignment to
perform optimally. These tools take multiple sequence alignments as input, with the objective
of using evolutionary information in the alignments to improve the accuracy performance of the
algorithm. The quality of the alignment is a large limiting factor to the performance of these tools,
so we evaluate the performance of the tools as a function of the alignments’ minimum percent
identity. Specifically, we start with the full unfiltered alignment, and then remove all sequences
with a percent identity (relative to the reference species) lower than the minimum threshold, and
run the resulting alignments with the algorithms selected. No filtering was done using minimum
sequence count or total tree length.

Our initial selection of aligners was based on recent assessments of multiple sequence aligners
[228, 229], where MAFFT (in “accurate mode” or L-INS-i) [230] and ProbConsRNA [231] were
selected. While these tools have been shown to perform well at aligning homologs with conserved
sequences, it is unknown if they can correctly align homologous conserved basepairing structure
which may exhibit covariation and thus lose sequence conservation. In order to alleviate this,
we also examine alignments from two structure-aware aligners LocARNA [232] and SPARSE [233]. The latest version of MAFFT also included two structure-aware alignment modes Q-INS-i and X-INS-i both of which we test. We conduct a very brief assessment of predictive performance on alignments created by the listed aligners, and made our final selection to use MAFFT Q-INS-i based on balance between accuracy and runtime performance, detailed in the Results section.

GotohScan

Version 2.0α was obtained from http://www.bioinf.uni-leipzig.de/software.html

GotohScan2a -d complete_genomes.fa -q reference_sequence.fa -o 0 -e 0.01 --quiet > output.csv

-d FASTA file containing completed genomes as separate entries
-o Output format selector, BLAST tabular format was chosen
-e Sets E-value, with downstream filtering in mind, a lenient value was chosen to maximize sensitivity
-quiet Don’t print status output of current query

For the bacterial dataset, the FASTA file containing 244 genomes is 1.1GB, making the search sufficiently swift if parallelized. For the fungal dataset, the 68 genomes are 4.2GB, which causes a significant slow down during searching. In practice, we split the genome file into 500 separate files of roughly equal size (taking care not to truncate any FASTA entries), then query each reference sequence against each one and find the union of the results.

BLAST was used in early trials, which was superior in terms of speed, but the inferior alignment quality made a non-negligible negative impact on tool performance and was thus abandoned.

MAFFT

Version 7.215 was obtained from http://mafft.cbrc.jp/alignment/software/

mafft-qinsi input.fa > output.fa

mafft-xinsi input.fa > output.fa
mafft-linsi input.fa > output.fa

Three modes were run during the testing phase of the aligner selection, and ultimately Q-INS-i was chosen as the only structure-aware mode that could successful align the largest sequences used in the study.

**LocARNA and SPARSE**

Version 1.8.1 of LocARNA was obtained from [http://www.bioinf.uni-freiburg.de/Software/LocARNA/](http://www.bioinf.uni-freiburg.de/Software/LocARNA/)

```
  mlocarna --quiet --keep-sequence-order --tgtdir output_directory input.fa

  mlocarna --sparse --tgtdir output_directory input.fa
```

SPARSE was distributed as an alignment mode in LocARNA and was run from the same binary.

**ProbConsRNA**

Version 1.1 of ProbConsRNA was downloaded from [http://probcons.stanford.edu/download.html](http://probcons.stanford.edu/download.html)

```
  probcons input.fa > output.fa
```

Note that the ProbConsRNA was used instead of the protein-sequence aligner ProbCons.

### 3.3.4 Related works and tools

Note that we focus on tools predicting general *non-biology-specific* RNA-RNA interactions, thus excluding the large collection of tools focused on predicting miRNA interactions and targets. While the general ideas of hybridization stability and accessibility apply to both the general and miRNA cases, modern miRNA prediction increasingly rely on miRNA-specific features that make their tools unsuited for predicting interactions outside of those for miRNAs. Reviews [209] and evaluations [234, 235] of miRNA tools have been covered extensively by other works.

Two notable related tools solving the interaction target prediction problem are RNAPredator [236] which utilizes RNAPLEX to predict the target partner of small bacterial sRNAs, and
COPRARNAs [237] which uses INTARNA to tackle the same sRNA target prediction problem. A recent assessment of target prediction tools for sRNA was done by Pain et al. [238], showing COPRARNAs as the best tool currently available for the task.

Finally, there were methods that fit the criteria of our evaluation but were excluded due to our inability to obtain or run them due to availability or practical reasons. These include GRNAS [239] (unavailable publicly), INTERNA [240] (algorithmically impractical), RIP [241] (unavailable publicly) and RIPALIGN [242] (algorithmically impractical).

3.4 Datasets

The evaluation of any new computational tool requires the compilation of a set of experimentally verified results. While RNA secondary structure tools have long benefited from curated and compiled datasets such as RNA STRAND [243] and RFAM [244], RNA-RNA interaction tool evaluation have so far only relied on ad hoc and varying datasets. Generally, tools have aimed to obtain a set of biologically functional interactions, consisting mostly of miRNA, bacterial small RNAs (sRNA), and snoRNAs. In this paper, we aggregate the sRNA and snoRNA interaction pairs that have been used in various papers, and here present what we believe is one of the largest, freely accessible and digitized collections of such RNA-RNA interactions. In contrast to previous datasets, we attempt to alleviate the biological bias of focusing on a single type of data, gathering both snoRNA and sRNA. Additionally, we also eliminate any bias on input length, taking the full length of target rRNA and mRNA sequences. We make the dataset available in tab-delimited format at doi:10.1093/nar/gkv1477 containing the full sequences and basepair information for future tool development and benchmarking efforts.

3.4.1 Bacterial sRNA

Small RNAs or sRNAs, are non-coding regulatory RNAs found in bacteria, shown to bind to the translational start sites of mRNAs, controlling the stability and translation of their targets. 40 to 400 nucleotides in length [124], sRNAs do not simply bind in zipper-like fashion to the mRNA across its entire length like the majority of miRNAs. Instead, sRNA-mRNA interactions vary significantly in stability and length, modulated by existing RNA secondary structures on both the sRNA and mRNA strands. Thus, the identification of the functionally relevant interaction serves as a challenging and relevant problem in RNA-RNA interaction prediction.

Functionally relevant sRNA-mRNA pairs are obtained from previously published experimental works, mostly derived from biochemical mapping experiments in *Escherichia coli* and *Salmonella*.
In earlier works a set of 18 interactions collected for INTARNA [217] was used by several works [212,245]. An expanded set tripling the interaction count was used for analysis of sRNA target binding regions [246,247], stated to be equivalent to sRNATarBase [248] published in parallel. Finally, the most recent and comprehensive set of over 100 interactions was compiled to evaluate the partner prediction problem in COPRARNA [237], a direct expansion of the aforementioned set.

Our sRNA dataset is a curated and digitized version of the interactions presented by COPRARNA, recovered from the supplementary material files and at times graphical figures in cited experimental publications to obtain the exact basepairing. The end product is a set of 109 sRNA-mRNA interactions (64 E. coli, 45 S. enterica) from 18 sRNAs against 82 mRNA targets. sRNA lengths range from 72 to 237 nucleotides, with a mean length of 123nt. The majority of interactions involve only one interaction site, but some pairs involve interactions at two disjoint sites. For these interactions (OxyS-fhlA and RprA-csgD in E. coli, GcvB-cycA, GcvB-tppB, and MicF-lpxR in S. enterica), each continuous segment is counted as one unique interaction for performance evaluation, resulting in these pairs having two solutions each. While it is technically possible to combine both sites into a long interaction containing a lengthy unpaired region in the middle, splitting it allows for both better predictive performance for the tools evaluated and also allows us to limit the maximum interaction length.

We used RefSeq [249] genomes for Escherichia coli str. K-12 substr. MG1655 (NC_000913.3) and Salmonella enterica subsp. enterica serovar Typhimurium str. LT2 (NC_003197) as our reference sequence. Sequences for sRNA and mRNA targets were extracted from genomes using gene names and associated GFF annotation files, along with 300 bases upstream of the translation start site. Once sequences were extracted, interactions from supplementary materials and manuscript figures were mapped onto the sequences and all interactions were confirmed to correspond to valid basepairs. The output data is stored in a computer-parsable CSV file, each line containing the full sRNA and mRNA sequences, along with the exact binding location and the basepair formation, available as Supplementary Files.

### 3.4.2 Fungal snoRNA

Small nucleolar RNAs or snoRNAs, are non-coding RNAs found in eukaryote and archaea, shown to stably bind to rRNAs, guiding essential chemical modifications at specific positions [117]. These RNAs are generally classified into C/D box snoRNAs that guide methylation and H/ACA snoRNAs that guide pseudouridylation. We focus on C/D box snoRNAs out of necessity, as H/ACA snoRNA interactions heavily depend on correctly folding intramolecular hairpins, making them
great for evaluating joint structures, but falling outside the scope of this work. For those interested in H/ACA predictions, we refer readers to the recent work RNAsnoop [250], which does a small comparison between three H/ACA prediction tools. A single C/D box snoRNA typically has one or two binding sites, ranging from 10 to 21 nt in length, forming highly complementary interactions with its target rRNA site. Despite the highly complementary interactions, the possible existence of multiple binding sites on the snoRNA, and the length of the rRNA targets (up to thousands of nucleotides) presents a very different problem than that posed by the sRNA set.

We chose to use yeast snoRNA-rRNA interactions from Saccharomyces cerevisiae, due to the completeness of the dataset and annotations available. Our dataset consists of 52 C/D box interactions obtained from Methylation Guide snoRNA Database [251], with additional interactions from the UMass Amherst Yeast snoRNA Database [252]. The 52 interactions are made by 43 unique snoRNAs and two rRNA targets. SnoRNA lengths range from 78 to 255 nucleotides with a mean length of 104nt, the full rRNA sequences are the 1800nt 18S rRNA and 3396nt 25S rRNA.

We use the Saccharomyces cerevisiae S288c genome from the Saccharomyces Genome Database [253], and extract snoRNA and rRNA sequences by gene name using the associated GFF genome annotation file. Interaction data from the Methylation Guide snoRNA Database [251] was reformatted and mapped onto the sequences, confirming the correctness of the basepairs. The final output is a computer parsable CSV file with each line representing an interaction, containing the full snoRNA and rRNA sequence, as well as the exact interaction sites and basepairing formation.

### 3.4.3 Multiple sequence alignments

Whereas minimum-free energy (MFE) methods only require a single sequence from each of the target and query RNAs, comparative methods require multiple sequence alignments (MSA) of homologs.

For the sRNA database, we obtained the list of 244 completed Enterobacteriaceae genomes from KEGG [254] and complete genomes listed from RefSeq. We used GotohScan [255] to find homologs for input reference sequences, and generated FASTA files of unaligned hits from the output. FASTA files were then aligned with MAFFT [230] in structurally-aware Q-INS-i mode with default settings. Finally, using percent identity (% ID) to the references species, we kept the top hit for any species that had more than one homolog hit. The above process resulted in MSA of sRNA and mRNA sequences, which were then used to make pairs of alignments containing the same species. For each interaction, the corresponding sRNA and mRNA alignments were taken, and the intersection of species was kept. The number of species after these intersections range
from 42 to 216, with a mean species count of 170. Pipeline scripting was done in Perl and R, with FASTA sequence and RNA structure manipulation done using the R4RNA package [256].

For the snoRNA database, we obtained the entirety of the RefSeq release 68 Fungi sequences Nov 2014, containing just under 3000 species. We followed the same procedure as described, also using MAFFT Q-INS-i on the full length rRNAs despite significantly longer runtimes. Species count for finalized alignments range from 5 to 44 with a mean count of 23.

### 3.4.4 Performance measures

We use the True Positive Rate (TPR, also called sensitivity) and Positive Predictive Value (PPV, also called selectivity [170]) to measure predictive performance. We only consider intermolecular basepairs, ignoring all intramolecular predictions. Given a set of predicted basepairs and a set of experimentally validated basepairs, each predicted basepair is either a True Positive (TP) if it also appears in the known set else it is a False Positive (FP). All basepairs in the known set that are not predicted as TP are False Negatives (FN), i.e. the prediction algorithm incorrectly predicts the basepair as non-pairing. Hence:

\[
TPR := \frac{TP}{TP+FN} \quad PPV := \frac{TP}{TP+FP}
\]

True negatives basepairs (TN) have traditionally been of little practical use for RNA secondary structure prediction evaluation, with the same applying in this work. Regardless, we describe its computation as it is required in some of our statistical analysis. We compute the number of TN basepairs as the “total” number of possible basepairs minus the number of TP basepairs as defined above. We estimate the total number of possible basepairs as \( \frac{n(n-1)}{2} \), where \( n \) is the length of the concatenated sequence, as estimated by [181]. This value is typically several magnitudes larger than the other values, and typically makes the specificity measure (also known as the true negative rate: \( TNR := \frac{TN}{TP+TN} \)) largely meaningless, as it is effectively 1 for all tools evaluated. For this reason, we use TPR and PPV, which are independent of the TN count.

Finally, we also use Matthews Correlation Coefficient (MCC) [257] as a rough summary of both TPR and PPV as defined as:

\[
MCC := \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\]

MCC ranges from 1 for predictions with maximum TPR and PPV, to -1 for very poor predictions, although in practice, the physical constraints of RNA basepairs result in a range between 0 and 1 for non-random predictions, and has been shown to be an approximation of the geometric
Considerations were made to subtract compatible basepairs as defined in [170] from the False Positive count, but doing so was found to be too lenient for tools which took a shotgun approach to predicting interactions.

3.5 Results

3.5.1 Minimum free energy results on sRNA dataset

We run the 10 energy-based tools against 109 sRNA-mRNA pairs, using the full length sRNA against truncated mRNA targets. Knowing the biology of sRNAs, it is not unreasonable for users to focus on a region around the translation start site. Previous studies have used similar windows, such as -150 and +100 bps relative to the start codon [247]. With all interactions falling within -130 and +104 bps, we begin with a conservative window of -150 and +150 bps. We run the analysis under two sets of options, the first being the most basic use case scenario of all default settings, followed by runs using optimal recommended settings when available.

Results for 109 pairs run on 10 tools are visualized as a heatmap in Figure 3.1 created using `ggplot` [259] in R, with default options on left, optimal options in middle and differences shown on the right. Hierarchical clustering of the results run with optimal settings clarifies an otherwise confusing set of results, and provides some immediate insight into tool similarity. As expected, results for accessibility-based INTARNA and RNALEX-a, interaction-only RISEARCH and RNAPLEX-c, and concatenation-based PAIRFOLD and RNAFOGOLD show highly similar performance profiles (Figure 3.2, left). RNA DUPLEX and RACTIP also cluster together, which was not apparent from their algorithmic strategies.

The mean of performance results is shown on Table 3.2 for the sRNA dataset, seen for results run with default options, optimal options, and the difference between the two runs where applicable. Setting correct options for RISEARCH and RNAPLEX-c are essential for obtaining competitive results, whereas all other tools gain only a small increase in performance (Figure 3.1 middle). For these two tools, the optimal setting involves setting the per nucleotide extension penalty to 0.3 kcal mol$^{-1}$—the average duplex energy between two random RNA bases [211]. For the two accessibility-based tools, optimal settings involve restricting the length of interactions to 60 nt, just larger than the longest interaction in the dataset.

According to mean MCC (Table 3.2), the best performing tool on the dataset is INTARNA (0.62) followed closely by RNALEX-a (0.58). The simple inclusion of accessibility information
may not completely explain this advantage over other tools, given that RNAUP too uses accessibility, yet only achieves a mean MCC of 0.39. Tools that perform poorly according to MCC, such as RNADUPLEX, appear to be severely penalized for predicting a large number of interacting basepairs resulting in a poor PPV.

Figure 3.1: Predictive accuracy measured by Matthews Correlation Coefficient for energy-based interaction prediction tools on the sRNA-mRNA dataset. Results for tools run on default (left) versus optimal (middle) options shown with differences (right), pairs and tools clustered hierarchically according to optimal results to group like results.
Figure 3.2: Clustering of tools according to energy-based MFE (left) and suboptimal (right) results on the sRNA-mRNA dataset.
<table>
<thead>
<tr>
<th></th>
<th>IntaRNA</th>
<th>RNAplex-a</th>
<th>GUUGle</th>
<th>Pairfold</th>
<th>RNAcofold</th>
<th>RactIP</th>
<th>RNAduplex</th>
<th>RIsearch</th>
<th>RNAplex-c</th>
<th>RNAup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCC (Optimal)</td>
<td>0.62</td>
<td>0.58</td>
<td>0.44</td>
<td>0.38</td>
<td>0.42</td>
<td>0.23</td>
<td>0.22</td>
<td>0.03</td>
<td>0.13</td>
<td>0.39</td>
</tr>
<tr>
<td>MCC (Default)</td>
<td>0.61</td>
<td>0.54</td>
<td>0.44</td>
<td>0.38</td>
<td>0.42</td>
<td>0.23</td>
<td>0.22</td>
<td>0.03</td>
<td>0.13</td>
<td>0.39</td>
</tr>
<tr>
<td>MCC (∆)</td>
<td>0.01</td>
<td>0.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPR (Optimal)</td>
<td>0.66</td>
<td>0.63</td>
<td>0.67</td>
<td>0.52</td>
<td>0.56</td>
<td>0.38</td>
<td>0.46</td>
<td>0.04</td>
<td>0.17</td>
<td>0.39</td>
</tr>
<tr>
<td>TPR (Default)</td>
<td>0.62</td>
<td>0.57</td>
<td>0.67</td>
<td>0.52</td>
<td>0.56</td>
<td>0.38</td>
<td>0.46</td>
<td>0.04</td>
<td>0.17</td>
<td>0.39</td>
</tr>
<tr>
<td>TPR (∆)</td>
<td>0.04</td>
<td>0.06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPV (Optimal)</td>
<td>0.61</td>
<td>0.56</td>
<td>0.39</td>
<td>0.29</td>
<td>0.33</td>
<td>0.15</td>
<td>0.11</td>
<td>0.03</td>
<td>0.11</td>
<td>0.48</td>
</tr>
<tr>
<td>PPV (Default)</td>
<td>0.64</td>
<td>0.52</td>
<td>0.39</td>
<td>0.29</td>
<td>0.33</td>
<td>0.15</td>
<td>0.11</td>
<td>0.03</td>
<td>0.11</td>
<td>0.48</td>
</tr>
<tr>
<td>PPV (∆)</td>
<td>-0.03</td>
<td>0.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>∆G (Optimal)</td>
<td>-13.03</td>
<td>-14.45</td>
<td>9.90^1</td>
<td>-130.47^2</td>
<td>-126.94^2</td>
<td>-64.26</td>
<td>-64.26</td>
<td>-2.30</td>
<td>-16.85</td>
<td>-16.74</td>
</tr>
<tr>
<td>∆G (Default)</td>
<td>-10.73</td>
<td>-14.11</td>
<td>9.90^1</td>
<td>-130.47^2</td>
<td>-126.94^2</td>
<td>-64.26</td>
<td>-64.26</td>
<td>-2.30</td>
<td>-16.85</td>
<td>-16.74</td>
</tr>
<tr>
<td>∆G (∆)</td>
<td>-2.30</td>
<td>-0.34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-2.30</td>
<td>-3.99</td>
<td></td>
</tr>
<tr>
<td>Basepairs (Optimal)</td>
<td>18.76</td>
<td>20.04</td>
<td>104.50</td>
<td>33.16</td>
<td>32.85</td>
<td>50.34</td>
<td>79.50</td>
<td>21.51</td>
<td>23.50</td>
<td>13.90</td>
</tr>
<tr>
<td>Basepairs (Default)</td>
<td>16.87</td>
<td>19.19</td>
<td>104.50</td>
<td>33.16</td>
<td>32.85</td>
<td>50.34</td>
<td>79.50</td>
<td>21.51</td>
<td>23.50</td>
<td>13.90</td>
</tr>
<tr>
<td>Basepairs (∆)</td>
<td>1.89</td>
<td>0.84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-6.84</td>
<td>-6.65</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3.2:** Mean accuracy performance for MFE predictions of energy-based tools on 109 sRNA-mRNA interactions. For each metric, we have results computed using optimal recommended settings, default settings, and the difference. Basepairs is the mean number of predicted intermolecular basepairs. Five tools lacked optimal settings, and RACTIP and GUUGLE did not compute ∆G free energies. GUUGLE output multiple results with no energy, so are more strictly “suboptimal” prediction. ^1 Minimal basepair threshold. ^2 Joint structure energies.
3.5.2 Suboptimal interaction results on sRNA dataset

While all the energy-based tools used produce a single minimum free-energy secondary structure by default, a majority of tools also allows the prediction of suboptimal results. In practice, this both allows for an increased sensitivity and also the ability to correctly predict pairs of interacting sequences with multiple binding sites. We determine the increases obtained by turning on suboptimal results for tools that have this option.

For this test, we use the same sRNA dataset and optimal options as used previously, with the exception of the new suboptimal results option enabled, set to allow for all suboptimal structures within reason to be returned. Specifically, when given the choice to set an energy threshold, we have the tools return all interactions with a predicted ΔG stability of \( \leq 0 \text{ kcal/mol} \). This is high enough to include the minimum free-energy structures and any suboptimal results that would be of practical interest, and low enough to keep output file sizes under control. It should be emphasized that this energy cutoff is not the one used for performance evaluation as follows. The definitions defined for MFE performance evaluation persist, but the predicted basepairs are now the union of intermolecular basepairs (i.e. no duplicates) from all suboptimal structures below a specific energy threshold. This specific energy threshold is unique to each tool for each interaction prediction, and is chosen to maximize the MCC value for that specific run. While knowing the energy threshold that maximizes the MCC up-front is impossible in the typical use case where the interactions are not known beforehand, we aim to derive the theoretical maximum of including suboptimal results, and thereby make potential recommendations on how to set such a threshold for future de novo runs.

Results for suboptimal results are visualized in Figure 3.3 (middle), retaining the same column and row ordering for easy comparison with the optimal MFE results in Figure 3.3 (left). Visually, the tools with the largest changes are RNA-DUPLEX, RISEARCH and RNA-PLEX-c, differences highlighted in Figure 3.3 (right). INTARNA and RNA-PLEX-a have relatively smaller gains to performance. Summarized in Table 3.3, we see that the three former tools roughly double TPR, but see little change to PPV rates. This increase in TPR likely stems from the increased total number of predicted basepairs, three to four times the number of bases for these three tools.

Measured by MCC, INTARNA (0.69) and RNA-PLEX-a (0.69) remain the two top performing tools, while RNA-PLEX-c (0.52) and RISEARCH (0.50) jump ahead to take third and fourth spot. Surprisingly, INTARNA has minimal change to the number of predicted basepairs even with suboptimal results enabled, meaning that the validated interactions are often its MFE prediction already. GUUGLE, which simply returns all valid ungapped interactions, does surprisingly well.
with an MCC of 0.44 if we simply take all predictions greater than 9 basepairs (mean 9.90). It obtains a TPR equal to RNAplex-a, but suffers from inadequate PPV, suggesting that Gibbs free energy serves as a good means to determine functional interactions from random ones. It is noted that while there is a shift in performance, the clustering of tools remains similar to before with only MFE results for this dataset (Figure 3.2, right).

In addition to a clustering of tools, the results of each tool also cluster to some extent. In Figures 3.4 and 3.5, we see MCC performance distributions of the energy-based tools on sRNA data returning MFE and suboptimal results (when applicable), respectively. We see a large number of results at 0 MCC when MFE results are returned, since predictions are either a hit or miss. With suboptimal options enabled, this peak at 0 mostly disappears for tools with the option. On this Figure 3.13, we show the TPR and PPV for each tool on each prediction, with a two dimensional density plot showing the rough clustering of results for each tool. Ignoring the large majority of points that end up with no predictions, we see a large concentration of accessibility-based predictions with a PPV of 1 and a range of TPR values. In contrast, tools without accessibility have a strong concentration of results with a TPR of 1, but a range of PPV values.

The Gibbs free energy ($\Delta G$) in Table 3.3 denotes the energy threshold used, below which bases are considered to be positively predicted. In practice, these could serve as guidelines for thresholds. The variance in energies between tools, however, makes setting clear guidelines difficult. The Rank denotes the average number of suboptimal results kept for each interaction to obtain the performances seen (i.e. MFE results effectively have Rank of 1). A lower rank doesn’t necessarily mean worse performance, since it might simply reflect a tool that successfully predicts the entire interaction via small separate interactions.
Figure 3.3: Predictive accuracy measured by Matthews Correlation Coefficient for energy-based interaction prediction tools on the sRNA-mRNA dataset. MFE (left) versus suboptimal (middle) results shown with differences (right), pairs and tools clustered hierarchically according to MFE results to group like results (Figure 3.1, middle). Suboptimal results only available in INTARNA, RISEARCH, RNA DUPLEX, and both versions of RNA PLEX.
Figure 3.4: MCC performance distribution of MFE results for energy-based tools using optimal settings (where applicable) on the sRNA-mRNA dataset.
Figure 3.5: MCC performance distribution of suboptimal results for energy-based tools using optimal settings (where applicable) on the sRNA-mRNA dataset.
<table>
<thead>
<tr>
<th></th>
<th>IntaRNA</th>
<th>RNAplex-a</th>
<th>GUUGle</th>
<th>Pairfold</th>
<th>RNAcofold</th>
<th>RactIP</th>
<th>RNAduplex</th>
<th>RIsearch</th>
<th>RNAplex-c</th>
<th>RNAup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCC (Suboptimal)</td>
<td>0.69</td>
<td>0.69</td>
<td>0.44</td>
<td></td>
<td></td>
<td></td>
<td>0.30</td>
<td>0.50</td>
<td>0.52</td>
<td></td>
</tr>
<tr>
<td>MCC (MFE)</td>
<td>0.62</td>
<td>0.58</td>
<td>0.38</td>
<td>0.42</td>
<td>0.23</td>
<td>0.22</td>
<td>0.35</td>
<td>0.33</td>
<td>0.33</td>
<td>0.39</td>
</tr>
<tr>
<td>MCC (Δ)</td>
<td>0.07</td>
<td>0.11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.08</td>
<td>0.14</td>
<td>0.19</td>
<td></td>
</tr>
<tr>
<td>TPR (Suboptimal)</td>
<td>0.76</td>
<td>0.86</td>
<td>0.76</td>
<td></td>
<td></td>
<td></td>
<td>0.86</td>
<td>0.75</td>
<td>0.77</td>
<td></td>
</tr>
<tr>
<td>TPR (MFE)</td>
<td>0.66</td>
<td>0.63</td>
<td>0.52</td>
<td>0.56</td>
<td>0.38</td>
<td>0.46</td>
<td>0.34</td>
<td>0.32</td>
<td>0.32</td>
<td>0.39</td>
</tr>
<tr>
<td>TPR (Δ)</td>
<td>0.10</td>
<td>0.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.40</td>
<td>0.42</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>PPV (Suboptimal)</td>
<td>0.72</td>
<td>0.62</td>
<td>0.39</td>
<td></td>
<td></td>
<td></td>
<td>0.13</td>
<td>0.39</td>
<td>0.43</td>
<td></td>
</tr>
<tr>
<td>PPV (MFE)</td>
<td>0.61</td>
<td>0.56</td>
<td>0.29</td>
<td>0.33</td>
<td>0.15</td>
<td>0.11</td>
<td>0.39</td>
<td>0.35</td>
<td>0.35</td>
<td>0.48</td>
</tr>
<tr>
<td>PPV (Δ)</td>
<td>0.11</td>
<td>0.06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.02</td>
<td>0.00</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td>ΔG (MFE)</td>
<td>-13.03</td>
<td>-14.45</td>
<td>-130.47</td>
<td></td>
<td></td>
<td></td>
<td>-64.26</td>
<td>-19.79</td>
<td>-25.11</td>
<td>-16.74</td>
</tr>
<tr>
<td>ΔG (Δ)</td>
<td>-0.31</td>
<td>0.68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.88</td>
<td>3.11</td>
<td>2.40</td>
<td></td>
</tr>
<tr>
<td>Bps (Suboptimal)</td>
<td>19.49</td>
<td>35.05</td>
<td>104.50</td>
<td></td>
<td></td>
<td></td>
<td>233.37</td>
<td>62.38</td>
<td>56.28</td>
<td></td>
</tr>
<tr>
<td>Bps (MFE)</td>
<td>18.76</td>
<td>20.04</td>
<td>33.16</td>
<td>32.85</td>
<td>50.34</td>
<td>79.50</td>
<td>14.67</td>
<td>16.85</td>
<td>13.90</td>
<td></td>
</tr>
<tr>
<td>Bps (Δ)</td>
<td>0.72</td>
<td>15.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>153.86</td>
<td>47.71</td>
<td>39.42</td>
<td></td>
</tr>
<tr>
<td>Rank (Suboptimal)</td>
<td>1.12</td>
<td>3.05</td>
<td>13.42</td>
<td></td>
<td></td>
<td></td>
<td>10.75</td>
<td>5.30</td>
<td>3.67</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3.3:** Mean accuracy performance for suboptimal predictions of energy-based tools on 109 sRNA-mRNA interactions. For each metric, we have results computed with suboptimal results enabled, disabled (MFE) and the difference. Basepairs is the mean number of predicted intermolecular basepairs. Five tools lacked the suboptimal option, RactIP and GUUGle did not compute ΔG free energies. ¹Minimal basepair threshold. ²Joint structure energies.
3.5.3 Effect of increasing target sequence size on sRNA dataset

Using optimal options and suboptimal results where applicable, we test the accuracy performance of energy-based tools on the sRNA dataset, but this time increasing the length of the target mRNA sequence. Previously, we had used the full-length sRNA sequence against a 300nt window around the translation start site, 150nt upstream, 150nt downstream. Here, we keep the 150nt upstream the same, but gradually increase the length of the coding sequence (CDS) downstream by increments of 100nt, until we are 1150nt upstream, having target sequences up to 1300nt, roughly the length of the average bacterial gene.

The resulting performance as measured by MCC is seen in Figure 3.6, showing a monotonically decreasing trend for all tools. All tools have difficulties maintaining PPV, resulting in an overall decrease in MCC as the search space increases. This is alleviated somewhat by tools that produce suboptimal results, as they are able to maintain a high TPR rate, which is often untrue for tools that only return MFE results. The rate of decrease varies between tools, with a few such as INTARNA and RNAPEX-a having relatively linear trends, while RISEARCH and RNAPEX-c having fast asymptotic trends. Extrapolating, it is likely that increasing the length even further will result in a further decrease in performance for all tools, which would have worrying implications for full-transcriptome searches.

3.5.4 MFE energy-based results on snoRNA dataset

In order to examine whether the performances observed in the sRNA dataset are generalizable to interactions of other types, we evaluate the performance of all tools on our second dataset, consisting of 52 C/D snoRNA-rRNA interactions. We proceed straight to evaluating the performance using optimal settings, adjusting the maximal interaction length to 25, just large enough to capture all known interactions in the dataset. We start by running MFE results following by suboptimal results.

We run the energy-based tools on two versions of the dataset using the same 52 snoRNA and rRNA pairs. We first determine the performance in an ideal scenario, knowing the general binding region of the snoRNA on the rRNA sequence, having the full length snoRNA interact with a 300nt subsequence of the target rRNA centered around the center of the binding site. We then test a more realistic de novo scenario, interacting the full snoRNA against the entirety of the target rRNA.

MFE results on the short and long dataset are seen on are seen in Table 3.4 and Figure 3.7, which repeat our observation that increasing the search space results in a significant drop in performance for all tools. In contrast to suboptimal results, the drop in MCC is caused by decreasing
performance in both TPR and PPV.
Figure 3.6: MCC (top), TPR (middle) and PPV (bottom) performance of energy-based tools as a function of length. Optimal options and suboptimal results are used where applicable.
Figure 3.7: Predictive accuracy measured by Matthews Correlation Coefficient for energy-based minimum-free energy prediction tools on the snoRNA-rRNA dataset. Results on the short dataset (left) versus the long dataset (middle) with differences (right).
<table>
<thead>
<tr>
<th></th>
<th>IntaRNA</th>
<th>RNAplex-a</th>
<th>GUUGle</th>
<th>Pairfold</th>
<th>RNAcofold</th>
<th>RactIP</th>
<th>RNAduplex</th>
<th>RIssearch</th>
<th>RNAplex-c</th>
<th>RNAup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCC (Short)</td>
<td>0.72</td>
<td>0.75</td>
<td>0.86</td>
<td>0.45</td>
<td>0.50</td>
<td>0.31</td>
<td>0.31</td>
<td>0.82</td>
<td>0.82</td>
<td>0.56</td>
</tr>
<tr>
<td>MCC (Long)</td>
<td>0.56</td>
<td>0.52</td>
<td>0.62</td>
<td>0.18</td>
<td>0.23</td>
<td>0.18</td>
<td>0.15</td>
<td>0.58</td>
<td>0.58</td>
<td>0.45</td>
</tr>
<tr>
<td>MCC (Δ)</td>
<td>0.16</td>
<td>0.23</td>
<td>0.24</td>
<td>0.26</td>
<td>0.27</td>
<td>0.13</td>
<td>0.16</td>
<td>0.24</td>
<td>0.24</td>
<td>0.11</td>
</tr>
<tr>
<td>TPR (Short)</td>
<td>0.77</td>
<td>0.82</td>
<td>0.97</td>
<td>0.72</td>
<td>0.75</td>
<td>0.61</td>
<td>0.70</td>
<td>0.87</td>
<td>0.84</td>
<td>0.63</td>
</tr>
<tr>
<td>TPR (Long)</td>
<td>0.61</td>
<td>0.56</td>
<td>0.96</td>
<td>0.31</td>
<td>0.40</td>
<td>0.36</td>
<td>0.32</td>
<td>0.62</td>
<td>0.60</td>
<td>0.51</td>
</tr>
<tr>
<td>TPR (Δ)</td>
<td>0.16</td>
<td>0.25</td>
<td>0.01</td>
<td>0.41</td>
<td>0.35</td>
<td>0.25</td>
<td>0.37</td>
<td>0.25</td>
<td>0.24</td>
<td>0.12</td>
</tr>
<tr>
<td>PPV (Short)</td>
<td>0.68</td>
<td>0.70</td>
<td>0.81</td>
<td>0.28</td>
<td>0.35</td>
<td>0.16</td>
<td>0.14</td>
<td>0.78</td>
<td>0.80</td>
<td>0.60</td>
</tr>
<tr>
<td>PPV (Long)</td>
<td>0.53</td>
<td>0.49</td>
<td>0.49</td>
<td>0.11</td>
<td>0.13</td>
<td>0.09</td>
<td>0.07</td>
<td>0.55</td>
<td>0.56</td>
<td>0.47</td>
</tr>
<tr>
<td>PPV (Δ)</td>
<td>0.16</td>
<td>0.21</td>
<td>0.32</td>
<td>0.18</td>
<td>0.22</td>
<td>0.07</td>
<td>0.07</td>
<td>0.23</td>
<td>0.24</td>
<td>0.13</td>
</tr>
<tr>
<td>ΔG (Δ)</td>
<td>-0.55</td>
<td>0.13</td>
<td>0.06</td>
<td>-917.46</td>
<td>-887.41</td>
<td>-6.07</td>
<td>-0.99</td>
<td>-0.87</td>
<td>-1.07</td>
<td></td>
</tr>
<tr>
<td>Bps (Short)</td>
<td>14.77</td>
<td>15.42</td>
<td>34.67</td>
<td>34.62</td>
<td>29.79</td>
<td>49.94</td>
<td>66.98</td>
<td>15.31</td>
<td>14.10</td>
<td>13.96</td>
</tr>
<tr>
<td>Bps (Long)</td>
<td>15.38</td>
<td>15.58</td>
<td>174.58</td>
<td>38.04</td>
<td>43.85</td>
<td>57.02</td>
<td>72.69</td>
<td>15.96</td>
<td>14.63</td>
<td>15.31</td>
</tr>
<tr>
<td>Bps (Δ)</td>
<td>-0.62</td>
<td>-0.15</td>
<td>-139.90</td>
<td>-3.42</td>
<td>-14.06</td>
<td>-7.08</td>
<td>-5.71</td>
<td>-0.65</td>
<td>-0.54</td>
<td>-1.35</td>
</tr>
</tbody>
</table>

**Table 3.4:** Mean accuracy performance measures MFE predictions of energy-based tools on 52 snoRNA-rRNA interactions. For each metric, we evaluate predictions using the full length snoRNA against the full length rRNA (long), or an optimal 300nt window around the binding site on the rRNA (short). Basepairs is the mean number of predicted intermolecular basepairs. ¹Minimal basepair threshold. ²Joint structure energies.
3.5.5 Suboptimal energy-based results on snoRNA dataset

On Table 3.5, we see a numerical summary of the effects of enabling the suboptimal results option when available. The MCC gains obtained for enabling suboptimal results on this snoRNA dataset (0.03 to 0.08) are much smaller compared to the effects seen with sRNA (Table 3.3, 0.07 to 0.19). This is due to the MFE results often being the known interaction, making the additional predictions gained from suboptimal results mostly unnecessary. As seen in the TPR and PPV columns, enabling suboptimal options results in a trade-off, increasing TPR at the expense of PPV, ultimately resulting in a higher MCC similar to the sRNA dataset. In Figures 3.8 and 3.9, we see MCC performance distributions of the energy-based tools on snoRNA data returning MFE and suboptimal results (when applicable).

The results of using the short ideal window versus full rRNA are shown in Figure 3.10 and summarized in Table 3.8, with average results for each tools across the dataset and metrics. As seen in the sRNA dataset, tools form pairs and cluster closely together Figure 3.11 compared to Figure 3.2.

For the short ideal case (Table 3.8 (Short) rows), the average MCC performance is higher for all tools in comparison to the sRNA dataset, likely due to the simpler and more uniform interactions in this dataset. The simpler interactions are reflected in a much higher TPR rate for all tools with six out of the ten energy-based tools achieving a TPR rate of $\geq 0.91$, detecting almost all known interactions. With the exception of GUUGLE, RISEARCH, and RNAPLEX-c which double their PPV rates from around 0.40 to 0.80, most tools only see relatively small improvements to PPV.

When we extend the target to full length rRNAs (Table 3.8 (Long) rows), we see a significant decrease in performance for all tools as the number of positive predictions increase, the majority of them false. Based on MCC, INTARNA, RNAUP, RNA Duplex, RACTIP and RNA Plex-a suffer a relatively smaller drop in MCC (-0.10 to -0.14), while the remaining tools suffer a larger decrease (-0.24 to -0.29). For the latter tools, interaction-only tools with suboptimal options (GUUGLE, RISEARCH and RNAPLEX-c) see little change in TPR, but experience significant decreases in PPV, explaining the MCC drop. Concatenation tools PAIRFOLD and RNA COFOLD see significant drops in TPR, PPV and MCC performances.

Based on the Gibbs free energy cutoffs and the predicted number of basepairs (Table 3.8 Bps columns), increasing the target search space does not significantly change the energy threshold, but the number of basepairs that pass this threshold increases. With the exception of RNAUP, tools that do not compute suboptimal results do extremely poorly with the increased search space. Of the remaining tools that do compute suboptimal results, accessibility seems to be key to preventing
huge losses in PPV, perhaps explaining why RNAUP remains competitive.
Figure 3.8: MCC performance distribution of MFE results for energy-based tools using optimal settings (where applicable) on the snoRNA-rRNA dataset.
Figure 3.9: MCC performance distribution of suboptimal results for energy-based tools using optimal settings (where applicable) on the snoRNA-rRNA dataset.
Figure 3.10: Predictive accuracy measured by Matthews Correlation Coefficient for energy-based interaction prediction tools on the snoRNA-rRNA dataset. Results on the short truncated target dataset (left) versus the long full-length target dataset (middle) with differences (right).
Figure 3.11: Clustering of tools according to energy-based MFE (left) and suboptimal (right) results on the snoRNA-rRNA dataset.
<table>
<thead>
<tr>
<th></th>
<th>IntaRNA</th>
<th>RNApex-a</th>
<th>GUUGle</th>
<th>Pairfold</th>
<th>RNAcofold</th>
<th>RactIP</th>
<th>RNAduplex</th>
<th>RIsearch</th>
<th>RNAplex-c</th>
<th>RNAup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCC (Suboptimal)</td>
<td>0.80</td>
<td>0.83</td>
<td>0.86</td>
<td></td>
<td></td>
<td></td>
<td>0.38</td>
<td>0.87</td>
<td>0.84</td>
<td></td>
</tr>
<tr>
<td>MCC (MFE)</td>
<td>0.72</td>
<td>0.75</td>
<td>0.45</td>
<td>0.50</td>
<td>0.31</td>
<td>0.31</td>
<td>0.82</td>
<td>0.82</td>
<td>0.56</td>
<td></td>
</tr>
<tr>
<td>MCC (∆)</td>
<td>0.08</td>
<td>0.08</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.07</td>
<td>0.06</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>TPR (Suboptimal)</td>
<td>0.91</td>
<td>0.95</td>
<td>0.97</td>
<td></td>
<td></td>
<td></td>
<td>0.97</td>
<td>0.98</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>TPR (MFE)</td>
<td>0.77</td>
<td>0.82</td>
<td>0.72</td>
<td>0.75</td>
<td>0.61</td>
<td>0.70</td>
<td>0.87</td>
<td>0.84</td>
<td>0.63</td>
<td></td>
</tr>
<tr>
<td>TPR (∆)</td>
<td>0.13</td>
<td>0.14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.27</td>
<td>0.11</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>PPV (Suboptimal)</td>
<td>0.75</td>
<td>0.76</td>
<td>0.81</td>
<td></td>
<td></td>
<td></td>
<td>0.16</td>
<td>0.81</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>PPV (MFE)</td>
<td>0.68</td>
<td>0.70</td>
<td>0.28</td>
<td>0.35</td>
<td>0.16</td>
<td>0.14</td>
<td>0.78</td>
<td>0.80</td>
<td>0.60</td>
<td></td>
</tr>
<tr>
<td>PPV (∆)</td>
<td>0.07</td>
<td>0.06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.02</td>
<td>0.03</td>
<td>-0.03</td>
<td></td>
</tr>
<tr>
<td>ΔG (Suboptimal)</td>
<td>-12.79</td>
<td>-14.12</td>
<td>12.58</td>
<td></td>
<td></td>
<td></td>
<td>-50.39</td>
<td>-21.44</td>
<td>-23.29</td>
<td></td>
</tr>
<tr>
<td>ΔG (MFE)</td>
<td>-12.39</td>
<td>-14.46</td>
<td>-125.35</td>
<td></td>
<td></td>
<td></td>
<td>-47.18</td>
<td>-20.45</td>
<td>-22.17</td>
<td>-17.10</td>
</tr>
<tr>
<td>ΔG (∆)</td>
<td>-0.40</td>
<td>0.35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-3.20</td>
<td>-0.99</td>
<td>-1.13</td>
<td></td>
</tr>
<tr>
<td>Bps (Suboptimal)</td>
<td>18.23</td>
<td>17.75</td>
<td>34.67</td>
<td></td>
<td></td>
<td></td>
<td>118.94</td>
<td>17.90</td>
<td>20.27</td>
<td></td>
</tr>
<tr>
<td>Bps (MFE)</td>
<td>14.77</td>
<td>15.42</td>
<td>34.62</td>
<td>29.79</td>
<td>49.94</td>
<td>66.98</td>
<td>15.31</td>
<td>14.10</td>
<td>13.96</td>
<td></td>
</tr>
<tr>
<td>Bps (∆)</td>
<td>3.46</td>
<td>2.33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.96</td>
<td>2.60</td>
<td>6.17</td>
<td></td>
</tr>
<tr>
<td>Rank (Suboptimal)</td>
<td>1.31</td>
<td>1.38</td>
<td>3.88</td>
<td></td>
<td></td>
<td></td>
<td>4.19</td>
<td>1.19</td>
<td>1.58</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3.5:** Mean accuracy performance measures suboptimal predictions of energy-based tools on 52 snoRNA-rRNA interactions. For each metric, we have results computed with suboptimal results enabled, disabled (MFE) and the difference. Basepairs is the mean number of predicted intermolecular basepairs. Five tools lacked the suboptimal option, RactIP and GUUGle did not compute ΔG free energies. ¹Minimal basepair threshold. ²Joint structure energies.
3.5.6 Comparative predictions for sRNA dataset

As seen in Table 3.6 and Figure 3.12, the minimum percent identity (% ID) has a large effect on the MCC, TPR and PPV values of the four tools evaluated. We observe a monotonically increasing trend for TPR as the minimum percent identity increases, suggesting that the experimentally determined basepairs are not extremely well conserved, and are only detected when a majority of divergent homologs are filtered out. PPV values fluctuate depending on the tool, with RNAplex-c and RNAaliduplex seeing a slight decreasing trend. With the exception of PETcofold, there appears to be a clear trade-off between TPR and PPV, with TPR increasing while PPV decreases as the minimum percent identity of the alignment increases.

Maximal MCC performances for tools are obtained at 70% ID for RNAaliduplex (0.31), 80% ID for PETcofold (0.43), 90% ID for RNAplex-cA (0.56) and 100% ID for RNAplex-aA (0.69). For the three tools that have direct energy-based counterparts, these MCC values are greater or equal to the performance values seen in Table 3.3, with an increase in performance of 0.01 RNAaliduplex, 0.04 RNAplex-cA, and 0.00 RNAplex-aA. Take note that optimal comparative MCC values were obtained at different minimum percent identity thresholds, with the RNAplex-aA threshold of 100% effectively being the energy-based methods as no divergent information was present in the alignment.

<table>
<thead>
<tr>
<th>% ID</th>
<th>PETcofold MCC</th>
<th>TPR</th>
<th>PPV</th>
<th>RNAaliduplex MCC</th>
<th>TPR</th>
<th>PPV</th>
<th>RNAplex-aA MCC</th>
<th>TPR</th>
<th>PPV</th>
<th>RNAplex-cA MCC</th>
<th>TPR</th>
<th>PPV</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>0.30</td>
<td>0.34</td>
<td>0.30</td>
<td>0.20</td>
<td>0.30</td>
<td>0.22</td>
<td>0.19</td>
<td>0.20</td>
<td>0.55</td>
<td>0.23</td>
<td>0.21</td>
<td>0.57</td>
</tr>
<tr>
<td>50</td>
<td>0.30</td>
<td>0.33</td>
<td>0.29</td>
<td>0.22</td>
<td>0.32</td>
<td>0.23</td>
<td>0.20</td>
<td>0.21</td>
<td>0.55</td>
<td>0.24</td>
<td>0.22</td>
<td>0.65</td>
</tr>
<tr>
<td>55</td>
<td>0.29</td>
<td>0.33</td>
<td>0.29</td>
<td>0.23</td>
<td>0.37</td>
<td>0.23</td>
<td>0.24</td>
<td>0.26</td>
<td>0.58</td>
<td>0.27</td>
<td>0.25</td>
<td>0.67</td>
</tr>
<tr>
<td>60</td>
<td>0.35</td>
<td>0.39</td>
<td>0.33</td>
<td>0.26</td>
<td>0.43</td>
<td>0.23</td>
<td>0.31</td>
<td>0.32</td>
<td>0.57</td>
<td>0.31</td>
<td>0.29</td>
<td>0.71</td>
</tr>
<tr>
<td>65</td>
<td>0.33</td>
<td>0.39</td>
<td>0.31</td>
<td>0.30</td>
<td>0.53</td>
<td>0.24</td>
<td>0.37</td>
<td>0.42</td>
<td>0.53</td>
<td>0.41</td>
<td>0.39</td>
<td>0.72</td>
</tr>
<tr>
<td>70</td>
<td>0.34</td>
<td>0.43</td>
<td>0.31</td>
<td>0.31</td>
<td>0.61</td>
<td>0.22</td>
<td>0.42</td>
<td>0.46</td>
<td>0.60</td>
<td>0.45</td>
<td>0.48</td>
<td>0.63</td>
</tr>
<tr>
<td>75</td>
<td>0.41</td>
<td>0.50</td>
<td>0.37</td>
<td>0.28</td>
<td>0.64</td>
<td>0.16</td>
<td>0.45</td>
<td>0.50</td>
<td>0.58</td>
<td>0.47</td>
<td>0.51</td>
<td>0.60</td>
</tr>
<tr>
<td>80</td>
<td>0.43</td>
<td>0.53</td>
<td>0.37</td>
<td>0.29</td>
<td>0.71</td>
<td>0.15</td>
<td>0.50</td>
<td>0.57</td>
<td>0.57</td>
<td>0.50</td>
<td>0.58</td>
<td>0.57</td>
</tr>
<tr>
<td>85</td>
<td>0.39</td>
<td>0.49</td>
<td>0.34</td>
<td>0.29</td>
<td>0.76</td>
<td>0.14</td>
<td>0.50</td>
<td>0.62</td>
<td>0.54</td>
<td>0.50</td>
<td>0.61</td>
<td>0.57</td>
</tr>
<tr>
<td>90</td>
<td>0.40</td>
<td>0.53</td>
<td>0.32</td>
<td>0.30</td>
<td>0.82</td>
<td>0.13</td>
<td>0.57</td>
<td>0.71</td>
<td>0.58</td>
<td>0.56</td>
<td>0.72</td>
<td>0.53</td>
</tr>
<tr>
<td>95</td>
<td>0.42</td>
<td>0.56</td>
<td>0.33</td>
<td>0.30</td>
<td>0.85</td>
<td>0.13</td>
<td>0.57</td>
<td>0.71</td>
<td>0.57</td>
<td>0.54</td>
<td>0.75</td>
<td>0.48</td>
</tr>
<tr>
<td>100</td>
<td>0.41</td>
<td>0.57</td>
<td>0.31</td>
<td>0.30</td>
<td>0.84</td>
<td>0.13</td>
<td>0.69</td>
<td>0.86</td>
<td>0.61</td>
<td>0.52</td>
<td>0.77</td>
<td>0.43</td>
</tr>
</tbody>
</table>

Table 3.6: Accuracy performance measures for comparative tools as a function of changing input multiple sequence alignment minimum percent identity on the sRNA dataset. Visually represented in Figure 2
Figure 3.12: Performance on comparative tools using alignments for the sRNA dataset, alignment sequences filtered by minimum percent identity.

3.5.7 Comparative predictions for snoRNA dataset

We test the effects of multiple sequence alignment inputs on the snoRNA dataset with truncated rRNA targets, again filtering alignments by minimum percent identity. As seen in Figure 3.14 and Table 3.7, this time we see that increasing percent identity actually results in a drop in MCC performance for three of the four MSA-based tools. Again, while we see a trade-off between increasing TPR and decreasing PPV as the minimum percent identity increases, the gains in TPR are quite minor, with the loss of PPV fairly significant as the minimum percent identity increases. These results seem to suggest that in this dataset, the benefits of an increase in PPV outweigh the penalties of decreased sensitivity, leading to an overall MCC that is superior to energy-based tools.

Maximal MCC performances for tools are obtained at 75% ID for RNAALIDUPLEX (0.67),
80% ID for PETCOFOLD (0.58), 80% ID for RNAplex-cA (0.83) and 80% ID for RNAplex-aA (0.82). For the three tools that have direct energy-based counterparts, these MCC values are greater or roughly equal to the performance values seen in Table 3.8 with an increase in performance of 0.29 RNAALIDUPLEX, 0.09 RNAplex-cA, and -0.01 RNAplex-aA. In contrast to the sRNA results, the increase in performance can be directly attributed to the conservation information, with an alignment of identical sequences (i.e. 100% minimum ID) resulting in inferior results.

Surprisingly, when comparing the two versions of RNAplex, the simpler RNAplex-cA that does not take in accessibility profiles does better than its accessibility-based comparative counterpart. With an optimal MCC value of 0.93, its accuracy is superior to all energy-based methods on the short snoRNA-rRNA dataset as seen on Table 3.8 (MCC (Short) row).

We follow up this evaluation by testing the same comparative tools on the snoRNA-rRNA dataset, but use the full-length rRNAs instead of the windowed binding sites. Simulating a de novo use case, we use no minimum percent identity filter, obtaining results seen in Table 3.9. As in the case of energy-based algorithms on Table 3.8 (MCC (Long) row), we see a drop in overall performance as measured by MCC. However, the usage of conservation information maintains a relatively higher PPV value. Most evident in RNAplex-cA, we observe a TPR rate of 0.94 and PPV of 0.81, resulting in an MCC measure of 0.85, eclipsing all other tools conservation or energy-based on the untruncated snoRNA-rRNA dataset.

![Figure 3.13](image-url)

**Figure 3.13:** Performance of MFE (left) and suboptimal results when applicable (right) for individual runs on the sRNA-mRNA dataset using energy-based tools.
<table>
<thead>
<tr>
<th>% ID</th>
<th>PETcofold MCC</th>
<th>PETcofold TPR</th>
<th>PETcofold PPV</th>
<th>RNAaliduplex MCC</th>
<th>RNAaliduplex TPR</th>
<th>RNAaliduplex PPV</th>
<th>RNAplex-aA MCC</th>
<th>RNAplex-aA TPR</th>
<th>RNAplex-aA PPV</th>
<th>RNAplex-cA MCC</th>
<th>RNAplex-cA TPR</th>
<th>RNAplex-cA PPV</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>0.54</td>
<td>0.67</td>
<td>0.48</td>
<td>0.67</td>
<td>0.94</td>
<td>0.51</td>
<td>0.82</td>
<td>0.86</td>
<td>0.85</td>
<td>0.93</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>70</td>
<td>0.54</td>
<td>0.67</td>
<td>0.48</td>
<td>0.67</td>
<td>0.94</td>
<td>0.51</td>
<td>0.82</td>
<td>0.86</td>
<td>0.85</td>
<td>0.93</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>75</td>
<td>0.55</td>
<td>0.68</td>
<td>0.48</td>
<td>0.67</td>
<td>0.94</td>
<td>0.51</td>
<td>0.82</td>
<td>0.86</td>
<td>0.85</td>
<td>0.93</td>
<td>0.95</td>
<td>0.92</td>
</tr>
<tr>
<td>80</td>
<td>0.55</td>
<td>0.68</td>
<td>0.50</td>
<td>0.66</td>
<td>0.95</td>
<td>0.50</td>
<td>0.82</td>
<td>0.87</td>
<td>0.84</td>
<td>0.93</td>
<td>0.95</td>
<td>0.92</td>
</tr>
<tr>
<td>85</td>
<td>0.52</td>
<td>0.69</td>
<td>0.48</td>
<td>0.62</td>
<td>0.95</td>
<td>0.44</td>
<td>0.80</td>
<td>0.86</td>
<td>0.83</td>
<td>0.92</td>
<td>0.95</td>
<td>0.90</td>
</tr>
<tr>
<td>90</td>
<td>0.58</td>
<td>0.77</td>
<td>0.47</td>
<td>0.53</td>
<td>0.96</td>
<td>0.33</td>
<td>0.81</td>
<td>0.89</td>
<td>0.81</td>
<td>0.92</td>
<td>0.96</td>
<td>0.89</td>
</tr>
<tr>
<td>95</td>
<td>0.54</td>
<td>0.80</td>
<td>0.37</td>
<td>0.40</td>
<td>0.97</td>
<td>0.18</td>
<td>0.82</td>
<td>0.95</td>
<td>0.75</td>
<td>0.85</td>
<td>0.94</td>
<td>0.82</td>
</tr>
<tr>
<td>100</td>
<td>0.54</td>
<td>0.84</td>
<td>0.35</td>
<td>0.38</td>
<td>0.97</td>
<td>0.16</td>
<td>0.81</td>
<td>0.95</td>
<td>0.72</td>
<td>0.81</td>
<td>0.94</td>
<td>0.75</td>
</tr>
</tbody>
</table>

*Table 3.7:* Accuracy performance measures for comparative tools as a function of changing input multiple sequence alignment minimum percent identity on the snoRNA dataset. Visually represented in Figure 3.14
Figure 3.14: Performance on comparative tools using alignments for the snoRNA dataset, alignment sequences filtered by minimum percent identity.
<table>
<thead>
<tr>
<th></th>
<th>IntaRNA</th>
<th>RNAplex-a</th>
<th>GUUGle</th>
<th>Pairfold</th>
<th>RNAcofold</th>
<th>RactIP</th>
<th>RNAduplex</th>
<th>RIssearch</th>
<th>RNAplex-c</th>
<th>RNAup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCC (Long)</td>
<td>0.70</td>
<td>0.71</td>
<td>0.62</td>
<td>0.18</td>
<td>0.23</td>
<td>0.18</td>
<td>0.24</td>
<td>0.58</td>
<td>0.60</td>
<td>0.45</td>
</tr>
<tr>
<td>MCC (Short)</td>
<td>0.80</td>
<td>0.83</td>
<td>0.86</td>
<td>0.45</td>
<td>0.50</td>
<td>0.31</td>
<td>0.38</td>
<td>0.87</td>
<td>0.84</td>
<td>0.56</td>
</tr>
<tr>
<td>MCC (Δ)</td>
<td>-0.10</td>
<td>-0.12</td>
<td>-0.24</td>
<td>-0.26</td>
<td>-0.27</td>
<td>-0.13</td>
<td>-0.14</td>
<td>-0.29</td>
<td>-0.24</td>
<td>-0.11</td>
</tr>
<tr>
<td>TPR (Long)</td>
<td>0.90</td>
<td>0.95</td>
<td>0.96</td>
<td>0.31</td>
<td>0.40</td>
<td>0.36</td>
<td>0.97</td>
<td>0.98</td>
<td>0.97</td>
<td>0.51</td>
</tr>
<tr>
<td>TPR (Short)</td>
<td>0.91</td>
<td>0.95</td>
<td>0.97</td>
<td>0.72</td>
<td>0.75</td>
<td>0.61</td>
<td>0.97</td>
<td>0.98</td>
<td>0.97</td>
<td>0.63</td>
</tr>
<tr>
<td>TPR (Δ)</td>
<td>-0.00</td>
<td>-0.00</td>
<td>-0.01</td>
<td>-0.41</td>
<td>-0.35</td>
<td>-0.25</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>-0.12</td>
</tr>
<tr>
<td>PPV (Long)</td>
<td>0.62</td>
<td>0.58</td>
<td>0.49</td>
<td>0.11</td>
<td>0.13</td>
<td>0.09</td>
<td>0.08</td>
<td>0.42</td>
<td>0.47</td>
<td>0.47</td>
</tr>
<tr>
<td>PPV (Short)</td>
<td>0.75</td>
<td>0.76</td>
<td>0.81</td>
<td>0.28</td>
<td>0.35</td>
<td>0.16</td>
<td>0.16</td>
<td>0.81</td>
<td>0.76</td>
<td>0.60</td>
</tr>
<tr>
<td>PPV (Δ)</td>
<td>-0.13</td>
<td>-0.17</td>
<td>-0.32</td>
<td>-0.18</td>
<td>-0.22</td>
<td>-0.07</td>
<td>-0.08</td>
<td>-0.39</td>
<td>-0.30</td>
<td>-0.13</td>
</tr>
<tr>
<td>ΔG (Δ)</td>
<td>-0.60</td>
<td>-1.21</td>
<td>-0.06</td>
<td>917.46</td>
<td>887.41</td>
<td>2.12</td>
<td>-0.07</td>
<td>-0.05</td>
<td>1.07</td>
<td>-0.60</td>
</tr>
<tr>
<td>Bps (Long)</td>
<td>52.44</td>
<td>34.87</td>
<td>174.58</td>
<td>38.04</td>
<td>43.85</td>
<td>57.02</td>
<td>723.35</td>
<td>75.50</td>
<td>97.27</td>
<td>15.31</td>
</tr>
<tr>
<td>Bps (Short)</td>
<td>18.23</td>
<td>17.75</td>
<td>34.67</td>
<td>34.62</td>
<td>29.79</td>
<td>49.94</td>
<td>118.94</td>
<td>17.90</td>
<td>20.27</td>
<td>13.96</td>
</tr>
<tr>
<td>Bps (Δ)</td>
<td>34.21</td>
<td>17.12</td>
<td>139.90</td>
<td>3.42</td>
<td>14.06</td>
<td>7.08</td>
<td>604.40</td>
<td>57.60</td>
<td>77.00</td>
<td>1.35</td>
</tr>
<tr>
<td>Rank (Long)</td>
<td>4.50</td>
<td>3.98</td>
<td>21.42</td>
<td></td>
<td></td>
<td></td>
<td>32.21</td>
<td>4.54</td>
<td>7.19</td>
<td></td>
</tr>
<tr>
<td>Rank (Short)</td>
<td>1.31</td>
<td>1.38</td>
<td>3.88</td>
<td></td>
<td></td>
<td></td>
<td>4.19</td>
<td>1.19</td>
<td>1.58</td>
<td></td>
</tr>
<tr>
<td>Rank (Δ)</td>
<td>3.19</td>
<td>2.60</td>
<td>17.54</td>
<td></td>
<td></td>
<td></td>
<td>28.02</td>
<td>3.35</td>
<td>5.62</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3.8:** Mean accuracy performance measures for suboptimal predictions of energy-based tools on 52 snoRNA-rRNA interactions. For each metric, we evaluate predictions using the full length snoRNA against the full length rRNA (long), or an optimal 300nt window around the binding site on the rRNA (short). Basepairs is the mean number of predicted intermolecular basepairs. ¹Minimal basepair threshold. ²Joint structure energies.
### Table 3.9: Comparative results on the snoRNA dataset, evaluated by running the full length snoRNA query against the full length rRNA sequence (long) or a 300nt window around the binding site (short).

<table>
<thead>
<tr>
<th></th>
<th>PETcofold</th>
<th>RNAaliduplex</th>
<th>RNAplex-cA</th>
<th>RNAplex-aA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MCC (Long)</strong></td>
<td>0.25</td>
<td>0.62</td>
<td>0.85</td>
<td>0.73</td>
</tr>
<tr>
<td><strong>MCC (Short)</strong></td>
<td>0.54</td>
<td>0.67</td>
<td>0.93</td>
<td>0.79</td>
</tr>
<tr>
<td><strong>MCC (Δ)</strong></td>
<td>-0.29</td>
<td>-0.05</td>
<td>-0.07</td>
<td>-0.06</td>
</tr>
<tr>
<td><strong>TPR (Long)</strong></td>
<td>0.29</td>
<td>0.94</td>
<td>0.94</td>
<td>0.83</td>
</tr>
<tr>
<td><strong>TPR (Short)</strong></td>
<td>0.67</td>
<td>0.94</td>
<td>0.94</td>
<td>0.82</td>
</tr>
<tr>
<td><strong>TPR (Δ)</strong></td>
<td>-0.38</td>
<td>0.00</td>
<td>-0.00</td>
<td>0.01</td>
</tr>
<tr>
<td><strong>PPV (Long)</strong></td>
<td>0.47</td>
<td>0.47</td>
<td>0.81</td>
<td>0.74</td>
</tr>
<tr>
<td><strong>PPV (Short)</strong></td>
<td>0.49</td>
<td>0.51</td>
<td>0.92</td>
<td>0.86</td>
</tr>
<tr>
<td><strong>PPV (Δ)</strong></td>
<td>-0.03</td>
<td>-0.05</td>
<td>-0.11</td>
<td>-0.12</td>
</tr>
<tr>
<td><strong>ΔG (Long)</strong></td>
<td>-23.96</td>
<td>-19.22</td>
<td>-12.53</td>
<td></td>
</tr>
<tr>
<td><strong>ΔG (Short)</strong></td>
<td>-24.06</td>
<td>-19.27</td>
<td>-11.08</td>
<td></td>
</tr>
<tr>
<td><strong>ΔG (Delta)</strong></td>
<td>0.09</td>
<td>0.05</td>
<td>-1.45</td>
<td></td>
</tr>
<tr>
<td><strong>Bps (Long)</strong></td>
<td>7.44</td>
<td>51.90</td>
<td>19.10</td>
<td>29.50</td>
</tr>
<tr>
<td><strong>Bps (Short)</strong></td>
<td>18.06</td>
<td>33.44</td>
<td>13.56</td>
<td>12.94</td>
</tr>
<tr>
<td><strong>Bps (Δ)</strong></td>
<td>-10.62</td>
<td>18.46</td>
<td>5.54</td>
<td>16.56</td>
</tr>
<tr>
<td><strong>Rank (Long)</strong></td>
<td>2.48</td>
<td>1.50</td>
<td>3.25</td>
<td></td>
</tr>
<tr>
<td><strong>Rank (Short)</strong></td>
<td>1.48</td>
<td>1.12</td>
<td>1.17</td>
<td></td>
</tr>
<tr>
<td><strong>Rank (Δ)</strong></td>
<td>1.00</td>
<td>0.38</td>
<td>2.08</td>
<td></td>
</tr>
</tbody>
</table>

### 3.5.8 Effect of different aligners on predictive performance

Using the snoRNA-rRNA dataset, we run sequence-based aligners ProbConsRNA [231] and MAFFT (L-INS-i mode) [230], and structurally-aware aligners LOCARNA [232], SPARSE [233], and two modes of MAFFT (Q-INS-i and X-INS-i). For each aligner, we give as input the full-length unfiltered unaligned snoRNA and rRNA alignments as inputs. After obtaining alignments we trim the target rRNA to a window surrounding the known interaction site and ensure each pair of alignments has the same species. Finally, we progressively filter the alignment according to minimum percent identity every 10%. We then run comparative interactions on the resulting alignments and evaluate performance in terms of MCC, TPR and PPV.
All tools successfully aligned snoRNA within a reasonable amount of time but only PROBCONS-sRNA, and two modes of MAFFT (L-INS-i and Q-INS-i) successfully completed the two rRNA alignments (18S and 25S) within a week of continuous runtime. For the tools that failed their rRNA alignments, MAFFT Q-INS-i rRNA alignments were used. This test was to be repeated on the sRNA-mRNA dataset, but multiple tools were unable to complete the mRNA alignments within a week of runtime.

Accuracy performance of the four comparative tools on the different alignments are measured in MCC (Table 3.15), TPR (Table 3.16) and PPV (Table 3.17). According to MCC, no tools seems clearly superior to all the others, while SPARSE and PROBCONS RNA clearly produce inferior results on specific tools. The three MAFFT modes perform extremely similarly. Considering both performance accuracy and runtime speed, the choice of MAFFT Q-INS-i is arguably the best choice given the selection of alignment algorithms.
Figure 3.15: Accuracy performance as measured by MCC of comparative interaction predictions tools on alignments constructed by different alignment algorithms and subsequently filtered by minimum percent identity.
Figure 3.16: Accuracy performance as measured by TPR of comparative interaction predictions tools on alignments constructed by different alignment algorithms and subsequently filtered by minimum percent identity.
**Figure 3.17:** Accuracy performance as measured by PPV of comparative interaction predictions tools on alignments constructed by different alignment algorithms and subsequently filtered by minimum percent identity.
Combining energy-based and comparative results

A common way to increase TPR or PPV is to combine results of multiple tools. Due to the number of tools we have, the potential number of combinations is unrealistic to fully explore. However, we take some time to test this technique on the three energy-based tools with comparative counterparts.

In Table 3.10, we show the predictive performance of RNA DUPLEX, RNA PLEX-c and RNA PLEX-a paired with comparative counterparts RNA ALIDUPLEX, RNA PLEX-cA and RNA PLEX-aA. For each pair of tools, we show the performance of the MFE algorithm, the MSA algorithm (MFE + alignment input), the union of results, and the intersection of results.

As expected, the union of results increases the TPR to values greater than the MSA or MFE results individually, but results in a large decrease in PPV. The intersection increases the PPV to values greater than the MSA or MFE results individually, and results in a decrease of TPR. Based of MCC results, taking the union or intersection can be highly beneficial at times, resulting in values greater than MFE or MSA results. However, the results are inconsistent and it is hard to recommend a consistent setting for specific tools.

However, if TPR or PPV are of particular interest in predicting basepair interactions, these results suggest that taking the union or intersection of results is not a bad approach.

Basepair covariation in datasets and background

To gain some understanding into the different effects that alignments had on the sRNA vs snoRNA datasets, we computed basepair covaration score and basepair conservation for the known inter- and intramolecular basepairs in the datasets. Figure 3.18 shows the binned distribution of basepairs according to their covariation scores (where -2 is unconserved, 0 is perfectly conserved, and 2 is covarying with compensatory mutations, as defined by [260]) and conservation (1 is perfect conservation of nucleotide, 0 is no conservation). Specifically, we take the multiple sequence alignment as previously described for the dataset, and compute the two scores for every intermolecular basepair between the snoRNA and rRNA, sRNA and mRNA, and the intramolecular basepairs between rRNA and rRNA derived from the solved structures from the Comparative RNA Website [200]. We also repeat the same scoring with random helices predicted on the same rRNA alignment to obtain a background, and random helices predicted on a shuffled (with MULTIPERM [261]) rRNA alignment for a fully random control. We filter all alignments used to 80% minimum percent identity in both datasets and controls, which is where most tools have been shown to perform best across both datasets.

The results from the plots are summarized in Table 3.11. From these plots, we clearly see...
that the majority of basepairs show strong conservation (1) and no covariation (0). Overall, we see a much stronger positive covariation score in intramolecular rRNA-rRNA results, with the intermolecular basepairs in snoRNA-rRNA and sRNA-rRNA showing less covariation. The intermolecular interactions show surprisingly similar covariation and conservation scores, both with

---

**Table 3.10:** Predictive performance of **RNA DUPLEX**, **RNA PLEX-c** and **RNA PLEX-a** paired with comparative counterparts **RNA ALIDUPLEX**, **RNA PLEX-cA** and **RNA PLEX-aA**. For each pair of tools, we show the performance of the MFE algorithm, the MSA algorithm (MFE + alignment input), the union of results, and the intersection of results. TP, FN, FP, TPR, PPV and MCC defined in main paper, where MCC here is estimated by the geometric mean of the TPR and PPV.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Covariation</th>
<th>Conservation</th>
<th>Basepairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>rRNA-rRNA</td>
<td>0.06±0.34</td>
<td>0.87±0.19</td>
<td>1504</td>
</tr>
<tr>
<td>snoRNA-rRNA</td>
<td>-0.03±0.13</td>
<td>0.98±0.06</td>
<td>664</td>
</tr>
<tr>
<td>sRNA-mRNA</td>
<td>-0.03±0.14</td>
<td>0.97±0.08</td>
<td>1879</td>
</tr>
<tr>
<td>Non-functional rRNA-rRNA</td>
<td>-0.22±0.22</td>
<td>0.82±0.13</td>
<td>1500</td>
</tr>
<tr>
<td>Shuffled rRNA-rRNA</td>
<td>-0.60±0.21</td>
<td>0.64±0.13</td>
<td>1500</td>
</tr>
</tbody>
</table>

Table 3.11: Covariation and conservation mean ± standard deviation scores for known and control inter- and intramolecular basepairs

slightly negative covariation scores and extremely high conservation scores.

3.6 Discussion

3.6.1 Settings and overfitting

As shown in Table 3.2, the effect of differing settings is significant for multiple tools on various datasets. Given no prior information and guidance, determining optimal settings is a non-trivial task. Even with recommended settings from authors, we noticed that settings that were optimal for one dataset cannot be assumed to work well with another. While the tools evaluated can technically be applied to any RNA sequence, the need for biologically-specific settings adds an extra layer of challenge for users.

In theory, it would be possible to perform an exhaustive search through the multi-variable setting space to find the setting values that maximize performance accuracy for each tool on each dataset. It is debatable, however, whether such dataset-specific settings would still perform well on other datasets and de novo user data as such parameters would potentially be extremely overfit and largely meaningless on other datasets and research settings.

We showed that enabling the suboptimal options for tools that support it consistently increases the overall predictive performance (Table 3.3). As touched upon previously, however, in practice the user would then have to deal with a ranked list of results instead of the single output. We have shown that the number of ranked results to use for optimal performance and the optimal Gibbs energy threshold cutoff varies greatly depending on both the tool and dataset in question, making specific suggestions difficult.
Figure 3.18: Covariation and conservation of basepairs in manuscript datasets compared to positive and negative backgrounds. Mean values ± the standard deviation for covariation and conservation are shown. rRNA-rRNA (positive control) shows the basepair distribution seen in conserved intramolecular basepairs. snoRNA-rRNA and sRNA-mRNA show basepair distributions of the experimentally validated (from single species) intermolecular basepairs used in the dataset. Non-functional and Shuffled rRNA-rRNA (negative controls) show basepair distribution for valid intramolecular basepairs on rRNA that are not in the known functional structure, and valid intramolecular basepairs on the rRNA predicted after shuffling the sequence. The number of basepairs for the negative controls were selected to be roughly approximate to the other sets.
3.6.2 Performance effects of conservation

According to benchmark evaluations in RNA secondary structure prediction, compensatory mutations in multiple sequence alignments can greatly aid the accurate prediction of basepairs [170].

For RNA-RNA interactions prediction, the inclusion of conservation information by giving alignments seems to bring mixed results depending on the tool and dataset. For interaction-only tools like RNAS-PLEX-c, the addition of conservation information increases the specificity, resulting in an overall MCC performance increase. When used in conjunction with accessibility-based methods (e.g. RNA-PLEX-a) however, additional alignment information does not seem to significantly increase the performance, and may even decrease performance like due to the alignments of questionable quality. Out of all the tools evaluated, the highest MCC performance (0.93) was achieved by RNAS-PLEX-cA on the snoRNA dataset with a relatively divergent input alignment (65% ID), showing that in the ideal case, conservation information can provide the best predictions. The number of variables that need to be correctly determined for this optimal result (i.e. alignment settings, percent identity threshold, settings, suboptimal results to keep), could make it impractical in a de novo setting.

Previous studies have observed that sRNA binding sites exhibit a high sequence conservation but low basepair conservation, further stating that covariance can only help a subset of interactions [247]. However, other studies also on sRNA have shown that under ideal circumstances [246], and sophisticated alignment methods [218], conservation can serve as a beneficial feature.

In addition to issues with the alignments due to the biology of sequence, the tools used to obtain homologs and align them can greatly effect the predictions, as shown through our usage of six different alignment methods. Choosing a proper aligner is a non-trivial task, with a need to consider computational restraints, RRI prediction tool, and whether an algorithmically more complex algorithm is actually worth a potential increase in performance. While our alignments could undoubtedly be improved with expert knowledge and manual curation, our work hopefully shows the issues with a high-throughput comparative de novo RNA-RNA interaction screen.

Compared to what is known from RNA secondary structure studies, the performance observed for comparative tools on the datasets at different minimum percent identity settings is perplexing. Possible avenues of explanation include things attributable to the user, such as alignment input and tool settings of which there are already a non-trivial amount to control. Additionally, it is known that many of the evolutionary models and scores employed in these intermolecular basepaire prediction algorithms were trained on intramolecular basepairs [210], for which we have suggested may be under different selective pressures and evolve differently. For all comparative algorithms
used, they combine thermodynamic and evolutionary components, with tunable weights for each component trained on specific datasets. These weights undoubtedly play a role in how tools react to changes in alignment quality, but require a non-trivial amount of work to optimize for specific datasets.

Taken together, the benefits of conservation information highly depend on the type of interaction (and possibly even the specific transcript pair) in question, complicated by the significant effect that homolog and alignment quality have on the results. Even under ideal circumstances, however, previous work has shown that accessibility aids correct prediction more than conservation [246] and that using both results in only a slightly higher performance. In practice, the effort and curation required to generate high quality alignments required is non-trivial making comparative potentially less appealing than energy-based methods.

### 3.6.3 Target size and interaction search space

Consistently observed in all tools across all datasets tested, increasing the length of the input sequences leads a decrease in predictive performance. For tools that enable multiple suboptimal results, there is little change to TPR, but difficulties in maintaining a high PPV results in an overall low MCC value. For tools that only produce a MFE result, both TPR and PPV suffer.

The inability of these tools to scale properly as input size increases is most problematic when applying these tools to predict potential interactions on a transcriptome-wide scale. These observations agree with existing interaction target prediction tools for sRNAs such as COPRA RNA and RNA PREDATOR shown to have PPV values of 44% and 28%, and TPR values of 23% and 32% respectively [237, 238].

While we demonstrate on the untruncated snoRNA dataset that conservation may be a strong feature to include to for an increased PPV rate, the difficulties discussed likely affect its usefulness in practice. COPRA RNA uses homolog information in its computations, and it is uncertain whether it is the limitations of the algorithm, the alignment quality, or the biology that limit its performance.

### 3.6.4 Runtime and memory performance

We show CPU runtimes and physical memory usages for the energy-based tools outputting suboptimal results (where applicable) when running on the sRNA-mRNA dataset with increasing long mRNA sequences in Figures 3.19 and 3.20, respectively.

With the exception of RNAUP, all tools ran in a few seconds to under a minute. Notably, GUUGLE, RISEARCH and RNADUPLEX returned results effectively immediately up to the maximal
input length of 1150 basepairs. INTARNA, RNAPLEX-c, RNAPLEX-a and RNACOFOLD saw roughly linearly increasing runtimes up to roughly 10 seconds. PAIRFOLD and RACTIP had polynomial runtimes with the longest jobs finishing in under a minute. RNAUP had runtimes several times larger than all other tools, with longer jobs taking several minutes.

For physical memory, most tools could comfortable run under a hundred or so megabytes, with RNAUP being the exception taking several times more. GUUGLE, RISEARCH and RNADUPLEX again used negligible amounts of memory. Interesting, both versions of RNAPLEX showed a constant memory usage, and it is unclear whether this is a consequence of the scanning-like algorithm, or a pre-allocation of memory whose limit we have yet to encounter. RNACOFOLD, PAIRFOLD, RACTIP and INTARNA use increasing amounts of memory relative to each other, each with increasing memory usage as a function of input length.

Extrapolating from these performances, it is likely that several tools would see little problem when applied to larger genome-wide searches, while others would need to be modified or perhaps used in a secondary pass in a larger pipeline.

3.7 Conclusion

RNA-RNA interaction prediction has increasingly become a field of intense interest, driven by advances in sequencing technology, uncovering a vast number of novel non-coding RNAs. The potential of using RNA-RNA interactions in identifying ncRNA targets may help us determine its networks and functions in the cell.

Fast and accurate full genome computational RNA interaction target searches are a sought after goal, which we believe starts with a strong foundation of being able to accurately predict interactions sites given two transcripts. In this work, we have conducted the most comprehensive assessment of general RNA-RNA interaction prediction tools to date. For this, we have compiled a comprehensive benchmark dataset, consisting of two biologically different types of functional and experimentally confirmed RRIs: bacterial sRNA-mRNA interactions that regulate translation, and yeast snoRNA-rRNA interactions that guide nucleotide modifications. Instead of artificially truncating input sequences around their know interaction sites, we provide the full query and target transcript sequence to simulate a realistic setting for de novo discoveries. We make our dataset publicly available for future research and development of new tools.

Evaluating all tools against all interactions in our dataset, we test not only the predictive accuracy of tools, but also the effects of various common settings seen in multiple tools. Of the four increasingly complex prediction strategies we grouped our tools into, those that did accessibility-
Figure 3.19: Boxplot of CPU runtime of tools on increasing input target mRNA sequences (query sRNA sequence length kept constant), using energy-based tools outputting sub-optimal results where applicable.

based predictions generally fared the best, with INTARNA consistently performing well across all datasets, RNASPLEX-a performing closely on many occasions, and RNAUP being an exception to this observation.
Figure 3.20: Boxplot of physical memory usage of tools on increasing input target mRNA sequences (query sRNA sequence length kept constant), using energy-based tools outputting suboptimal results where applicable.

The effects of adding evolutionary conservation information to predictions is highly mixed, ranging from detrimental effects on the sRNA dataset to impressive gains in performance on the untruncated snoRNA dataset. We further observe that the addition of conservation information to
accessibility information often results in an overall decrease in performance. This is unexpected given that the best methods for predicting RNA secondary structure work in a comparative way (by harnessing information on evolutionary conservation of base-pairs) and should also be seen as a warning as many of the current methods for predicting general RNA-RNA interaction deliberately employ similar ideas.

With the field’s goal of applying RNA-RNA interaction prediction to full-genome searches of RNA targets, we conduct a controlled experiment by increasing the target sequence length. As expected, we observed large drops in prediction accuracy, resulting in implications for large-scale searches.

The comparatively new field of general RNA-RNA interaction prediction thus needs a range of novel ideas compared to RNA secondary structure prediction to address the challenges shown by our benchmark tests.

Current prediction accuracies that pale in comparison to general RNA secondary structure prediction algorithms. It may be that we have reached the theoretical limits that a generalized non-biology-specific prediction algorithm can achieve, and that further performance gains can only be achieved by developing tools specific to a biological class of interactions. Various existing tools for miRNA prediction already pursue this, taking advantage of binding motifs and highly specific interactions lengths. There are also tools such as PLEXY [262], which can accurately predict C/D snoRNA binding sites by using nucleotide sequence motifs to narrow down the window of prediction. Alternatively, recent advancements have been made in high-throughput RNA structure and interaction probing [263]. These corresponding enzymatic probes and pairing constraints produced have the potential to greatly assist predictions, replacing pure in silico accessibility profiles with experimental binding evidence. Regardless, we hope that our assessment and the accompanying dataset will help improve the current state-of-art in RNA-RNA interaction prediction.
Chapter 4

Cotranscriptional RNA Folding and Prediction

In this chapter, we examine the biological *in vivo* process of RNA folding, and contrast it to the *in vitro* process currently captured in computational predictions. Specifically, we discuss base-pair formation beginning from the very moment the RNA polymer forms during transcription, or *cotranscriptionally*. We review the experimental and theoretical evidence for cotranscriptionally folding, then discuss existing and possible methods for capturing this to improve the computational prediction of basepairs.

This chapter has been published as a review:


4.1 Introduction and motivation

The primary products of all DNA genomes are RNA transcripts consisting of linear sequences of four different types of ribonucleic acids (abbreviated A, C, G and U and chemically different from the similarly abbreviated DNA building blocks A, C, G and T). When a gene of the genome is activated, a corresponding transcript is synthesized in a linear fashion with its 5'-end emerging first and its 3'-end emerging last. Primary transcripts vary greatly in length from a few nucleotides (nt) to $10^4$ nt and longer. They may be processed in a number of ways, *e.g.* splicing and RNA editing, which may happen while the transcript is being made. The functional role of some transcripts is exerted by RNA structure which is formed when pairs of complementary nucleotides of the
RNA sequence (C-G, A-U, G-U) form basepairs. In contrast to proteins, where we typically need to know its three-dimensional structure in order to study its potential functional roles, it often suffices to only know the RNA secondary structure in order to investigate its potential functional role(s). This RNA secondary structure is defined by the pairs of basepaired sequence positions in the RNA. RNA structure can either be global, i.e. span most of the transcript, or more local, i.e. be confined to a sub-sequence of the transcript. During its life in the cell, a single transcript may assume more than one functionally relevant RNA structure, e.g. riboswitches which can assume two mutually exclusive structures which are both functional.

Many computational methods for RNA structure prediction, in particular earlier and non-comparative methods, implicitly focus on predicting global RNA structures only. They are typically applied to analyze the non-coding portion of a given transcriptome as this is where globally structured RNA genes are suspected. RNA structural features, however, are also known to play important functional roles in regulating protein-coding transcripts (e.g. splicing, localization, degradation, translation initiation), yet this typically involves only local RNA structures which only some of the computational methods for RNA secondary-structure prediction can adequately model [264,265].

Recent advances in nucleotide sequencing technologies have enabled the routine sequencing of entire transcriptomes, with methods such as strand-specific RNA-seq, enabling the discovery of novel transcripts en masse. Experimental methods for RNA structure determination such as X-ray crystallography and NMR can provide atomic resolution three dimensional solutions, but remain relatively costly and comparatively slow. Computational methods for predicting RNA secondary structures based on RNA sequence information alone are therefore key to assigning potential functional roles to the transcriptome and identifying worthwhile targets for experimental validation.

When available, computational structure prediction can be aided by results from RNA footprinting experiments. Such experiments can estimate the pairing status of individual nucleotide positions in a single sequence with chemical probes, but cannot identify the pairing-partner involved in a basepair. Such methods, when paired with next-generation sequencing technologies, in protocols such as Frag-seq, PARS, and SHAPE-seq, show great promise in generating high-throughput RNA secondary structure probe maps [50]. Nonetheless, footprinting results still require algorithms to derive the overall most likely solution, again emphasizing the need for reliable and computationally efficient computational methods.

There exists by now ample experimental evidence that RNA structure formation starts cotranscriptionally, while the RNA is transcribed from the genome. The process of cotranscriptional structure formation is key to determining the resulting functional RNA structure(s) in vivo and that
this process can be influenced by a range of intrinsic as well as extrinsic factors. Yet, nearly all state-of-the-art methods for computational RNA secondary structure prediction ignore the structure formation process and focus exclusively on the end result, i.e. a single, final RNA structure. There already exist a few computational methods that aim to explicitly simulate the cotranscriptional folding pathway by capturing key features of the folding environment in vivo. As their prediction accuracy has so far been evaluated on only a few select sequences of typically short length, however, they are currently viewed as folding pathway prediction methods rather than RNA secondary structure prediction methods.

We argue that ignoring the formation process often yields decent structure predictions, especially for short and globally structured transcripts (< 200 nt), but that in order to increase the prediction accuracy for longer transcripts and to reach a conceptually better understanding, we ought to aim to take some effects of cotranscriptional folding into account.

In the following, we first review the variety of mechanisms that have been shown to influence cotranscriptional folding in vivo. This summarizes primarily experimental, but also some theoretical evidence for cotranscriptional folding. We then provide an overview of the currently existing methods for RNA secondary structure prediction. This part of the review is not aimed at providing a detailed description of every existing method for RNA secondary structure prediction, but rather at highlighting the different underlying concepts employed by these methods. At this point, we also cover methods for predicting RNA folding pathways which already capture some effects of cotranscriptional folding. To conclude, we propose a range of ideas how cotranscriptional folding could be captured in computational methods for RNA secondary structure prediction in order to further improve their prediction accuracy.

4.2 Experimental and theoretical evidence for cotranscriptional folding

4.2.1 Directionality of transcription

One of the most obvious differences between the in vivo and the typical in vitro setting is that RNA transcripts in vivo emerge sequentially starting with the 5’-end, whereas in vitro experiments start with an already synthesized molecule. The directionality of the molecule’s synthesis in vivo may thus lead to structural asymmetries during its cotranscriptional folding which may in turn influence the resulting functional RNA structure(s).
4.2.2 Transcription, transcription speed and variations thereof

Whether or not folding can happen during synthesis depends, amongst other things, on how the timescale of RNA synthesis compares to that of RNA structure formation. The speed of transcription not only depends on the underlying organism, but also on the polymerase responsible for generating the transcript in question. It ranges from 200 nucleotides per second (nts$^{-1}$) in phages, to 20–80 nts$^{-1}$ in bacteria and 10–20 nts$^{-1}$ for human polymerase II [141]. On the other hand, RNA folding is known to occur on a wide range of time scales; some RNAs fold in 10–100 ms [266], whereas kinetically trapped conformations can persist for minutes or hours [266-268]. Experiments in the early 1980s have shown that RNA structure formation can happen during transcription [269, 270], i.e. cotranscriptionally, and that folding in vivo can happen on the same timescale as RNA synthesis [271]. The latter was first shown for the cotranscriptional and structure-dependent self-splicing of the Tetrahymena group I intron [271].

Since then, several in vitro experiments have confirmed that RNA folding can happen cotranscriptionally and that the speed of transcription not only affects the overall folding rate, but also transient structures as well as the final structure [272-274]. Lewicki et al. [275] and Chao et al. [276] showed that altering the natural speed of transcription can yield misfolded and functionally inactive transcripts. Experimental studies of the Tetrahymena self-splicing intron are consistent with the view that a set of identical RNA molecules partitions into an active and an inactive pool, and that this partitioning is highly influenced by the cotranscriptional folding environment, including the RNA transcription rate [277].

For a given transcript, the speed of transcription is not necessarily constant. Transcriptional pausing can serve as an additional mechanism for fine-tuning cotranscriptional folding [278-280]. This pausing happens at specific transcript positions and for well-defined time intervals (ranging from 10$^{-6}$ s to 10 s). In bacteria, pausing can be due to interactions between the emerging RNA and the polymerase and/or polymerase-associated protein factors [281-283]. The flavin mono-nucleotide (FMN) dependent riboswitch in Bacillus subtilis [279] is a beautiful example of how these features can be combined into a cotranscriptional feedback loop where the binding of a metabolite selects one of two possible cotranscriptional folding pathways whose resulting RNA structure determines whether transcription is terminated or not.

4.2.3 Self-interactions including transient RNA structures

One of the key features of any RNA sequence is that it can interact with itself via basepairs between complementary nucleotides to form RNA structure. During cotranscriptional folding, al-
ready formed structures can un-pair and yield to other structures, in which case we refer to them as a transient structures. In other cases, it is energetically unfavourable for an existing structure to yield to a new conformation, thereby forming a kinetic trap. Transient structural features thus have the potential to significantly influence the cotranscriptional folding pathway and the resulting functional RNA structure(s), see Figure 4.2. Most of our current knowledge of transient structures, which we also refer to as cis RNA-RNA interactions, stems from dedicated experimental studies of select folding pathways which explore how RNA structure changes as a function of time.

Folding pathways of RNA transcripts in vitro have been the subject of intense study for a long time. Initial experiments primarily studied how already synthesized and fully denatured RNA molecules fold, whereas more recent studies examine cotranscriptional folding pathways in vitro and, most recently, also in vivo [284, 285]. As any of these experiments are technically sophisticated, our current view derives from a few well-studied test cases such as the hairpin ribozyme [286–290] and the Tetrahymena intron [277, 291]. These ribozymes are comparatively easy to study in vivo as their cleavage relies on distinct structural features whose products are easier to detect than the corresponding functional structures.

Cotranscriptional folding—whether in vitro and in vivo—tends to happen sequentially [288, 290] as basepairs at the 5’-end of the RNA can form first, whereas basepairs involving the 3’-end can only form once transcription is complete. This folding often involves transient RNA structure elements, i.e. structural features that are only present for a specific time-span [270, 292]. These can direct the structure formation via one or several folding pathways towards the desired structural configuration(s). These transient features may also play distinct functional roles. They may, for example, be required for template activity during (+)-strand synthesis in some viruses [292] or may serve as protein-binding sites during transcription [293]. These examples once again illustrate that any given RNA transcript may have more than a single functionally relevant RNA structure during its lifetime in the cell.

Cotranscriptional folding and other reaction rates in vivo typically differ from those in vitro with folding rates in vivo being typically [288, 290], but not always [286] higher than in vitro. One example is the cotranscriptional folding of the Tetrahymena ribozyme in vitro which is twice as fast as the refolding of the fully synthesized and denatured molecule, but slower than the cotranscriptional folding in vivo [273]. Cotranscriptional folding pathways in vivo need not be unique [291] and tertiary interactions can determine which of several possible folding pathways is chosen [294]. Factors such as transcription speed and flanking sequences can also influence which pathway dominates [277]. One of the few existing in vivo studies of cotranscriptional folding pathways [295] indirectly examined the structural folding intermediates of the Tetrahymena ribozyme at $10^{-5}$ s
time resolution using X-ray synchrotron radiation footprinting and chemical accessibility probing and found folding intermediates that are similar to those in vitro.

The tryptophan (trp) operon is a group of genes found in bacteria that act in the biosynthesis pathway of the amino acid tryptophan. The trp operon leader encodes a short peptide that is rich in tryptophan codons near the 5’-end of the RNA [296]. Regulation of the trp operon is carried out in part by the trp operon leader through a mechanism that relies on the simultaneous transcription of a DNA gene and translation of the resulting RNA in bacteria. The trp operon leader is a riboswitch that assumes two mutually exclusive structural configurations that form cotranscriptionally: the attenuator which prevents further transcription of the trp operon, and the anti-terminator which permits transcription [296]. When tryptophan levels are high, the ribosome proceeds rapidly through the operon leader, and interferes with the anti-terminator hairpin. When tryptophan levels are low, the ribosome stalls while translating the leader peptide, and allows the anti-terminator hairpin to form, and thus the trp operon is activated.

In addition to these experimental results, the bioinformatics community has conducted a range of computational studies to investigate cotranscriptional structure formation. Computational simulations of cotranscriptional folding pathways, e.g. [297], show that the basic features of cotranscriptional folding and their beneficial effects on RNA structure formation can be investigated in silico. Using a kinetic Monte Carlo Markov Chain (MCMC) to study the folding of the hepatitis delta virus ribozyme (87 nt length), Isambert and Siggia [297] show that cotranscriptional folding at the natural transcript speed of 50 nts\(^{-1}\) is significantly more efficient than when starting with a fully denatured sequence or when using the increased transcript speed of 1000 nts\(^{-1}\) that is typically used in in vitro experiments. By combining computational simulations of RNA folding pathways with phylogenetic structure analyses, Schoemaker and Gultyaev [298] investigate the effect of sRNA binding on ribosomal RNA (rRNA) structure formation during cotranscriptional folding and find that it significantly facilitates structure formation.

A bioinformatics analysis of 361 structural RNA genes [140] showed that these genes not only encode information on their known functional structure, but also on transient features of their respective cotranscriptional RNA folding pathways. For this, Meyer and Miklós examined helices (defined as contiguous stretches of adjacent basepairs) that could potentially out compete helices of the known structure. They found statistically significant 5’-to-3’ asymmetries between these competing helices and the respective helices of the known structure. More specifically, they identified two different types of transient structures: those that can yield to the functional structure and help its cotranscriptional formation and those that are more likely to act as kinetic traps during cotranscriptional folding. They showed that the former are preferentially encoded in the underlying
RNA sequences, whereas the latter are suppressed.

More recently, Zhu et al. [143] conducted a computational study of six RNA families with known transient and alternative structures in order to test whether evolutionarily related sequences not only assume similar final structures, but also share common transient structures during their respective cotranscriptional folding pathways. They find that some transient structures have been evolutionarily conserved on a level that is similar to those of the final structure. Moreover, they find that evolutionarily related sequences encounter similar transient structure features during their respective, predicted cotranscriptional folding pathways and that these features often coincide with known transient features.

To conclude, naturally occurring transcripts not only encode their functional RNA structure, but also information on how to get there via transient features that help define the corresponding cotranscriptional folding pathway.

4.2.4 Interactions with other molecules

One key difference between the in vivo and in vitro setting is that the cellular environment typically contains a wealth of additional molecules. In vivo, these may interact with the RNA transcript and thereby influence its structure formation and the resulting RNA structure, see Figure 4.2C. These molecules may comprise of proteins, RNA transcripts, metabolites, ligands and different types of ions. Any intermolecular interaction between two distinct RNA molecules, i.e. any trans RNA-RNA interaction, has the potential to prevent the thus bound RNA nucleotides from engaging in other interactions including RNA structure (i.e. cis RNA-RNA interactions). This may either stabilize or destabilize existing RNA structure features which may in turn influence the cotranscriptional folding pathway and the resulting RNA structures.

Due to the methodological challenges of investigating RNA folding in vivo and in real time, we currently have only limited insight into folding pathways in vivo [273, 291, 294, 295]. Numerous recent in vitro experiments that replicate specific aspects of the complex in vivo environment and rapid progress regarding in vivo methodologies [284, 299] are likely to change this.

So, which interactions between RNA transcripts and other molecules have been experimentally confirmed to be functionally important for RNA structure formation?

Ligand-RNA interactions One of the most obvious examples where RNA structure formation is influenced by trans interactions are so-called riboswitches. The change of one distinct RNA structure to another one is usually triggered by the binding of a metabolite or ion, but can also be
induced by a temperature change, at least in bacteria (thermoswitches) \cite{300-302}. The two distinct structural conformations of a riboswitch are typically located in the 5'-UTRs of messenger RNAs (mRNAs) and are mutually exclusive as they engage two overlapping sub-sequences. The structural change triggers a change of the gene’s expression by altering either its transcription, translation or splicing \cite{81,303}. Nechooshtan et al. \cite{304} identified a pH-responsive riboregulator up-stream of the alx ORF. For a high pH, the translationally active RNA structure is formed during transcription which involves two well-defined transcriptional pausing sites. Frieda and Block \cite{305} succeeded in directly observing the cotranscriptional folding of the pbuE adenine riboswitch. Using an optical assay which allowed them to monitor folding transitions in individual transcripts in real time, they showed that the transcriptional outcome of the riboswitch is kinetically controlled. Perdrizet et al. \cite{306} present strong evidence that the btuB riboswitch in Escherichia coli depends on the precise transcriptional pausing of its polymerase to guide its folding into its native structure.

**Protein-RNA interactions** In order for many large RNAs to fold in vitro into their functional structure without any other trans-acting molecules (apart from water), it is necessary to raise the concentration of metal ions (e.g. of Mg$^{2+}$) significantly above normal levels in vivo \cite{307,308}. Several in vitro experiments have shown that the ion concentration can be lowered if specific proteins are added that stabilize the RNA structure \cite{309-313} and that can bind folding intermediates \cite{310}. This has also been confirmed by several in vivo experiments \cite{314-316}.

RNA-binding proteins often play different functional roles depending on the binding interface they use to interact with different partners. One example is Cyt 18 in Neurospora crassa which not only aids RNA folding, but also acts as a splicing factor and an aminoacyl-tRNA synthetase \cite{314,317}. Most of these proteins bind an RNA in a sequence- or structure-specific way \cite{310,311,318-325}. There are also proteins, however, that interact with RNAs in a less specific way such as RNA helicases which help anneal and unwind RNAs while requiring ATP \cite{326-330} and hnRNP proteins which bind single-stranded stretches of pre-mRNAs and thereby aid splicing \cite{331}. Some protein-RNA interactions are required to happen at very specific times. One key example are ribosomal RNAs which are modified, processed and the corresponding ribosomes pre-assembled cotranscriptionally and in a tightly co-regulated way as shown in several in vivo experiments \cite{332-335}. There is also recent experimental evidence that cotranscriptional splicing is coupled to transcriptional pausing in yeast \cite{336} and that, interestingly, cotranscriptional splicing can also be coupled to translation as shown in vivo for the thymidylate synthase intron of the T4-phage \cite{337}. Therefore, RNA-binding proteins involved in splicing may thus act cotranscript-
Chaperone-RNA interactions

Chaperones are molecules, usually proteins, that assist a molecule’s correct folding by refolding misfolded structure features. Based on this definition, the trans-interaction partners of a given RNA transcript described above are not chaperones as they guide the correct co-folding pathway rather than help already misfolded RNA transcripts refold correctly. Many detailed experiments have shown that RNA transcripts can misfold in vitro and that it takes these molecules minutes to many hours or longer to escape these structural traps [338–340]. This may be attributed to several alternative folding pathways of the in vitro folding landscape which tends to be more rugged than the cotranscriptional folding landscape in vivo [341–343], but can also be due to individual RNA structure elements that keep the structure trapped.

There is some evidence that RNA structures can also misfold in vivo [291, 337] and that there exist dedicated cellular mechanisms for dealing with misfolded RNA structures, e.g. by sequestering and degrading them as shown for the Tetrahymena intron [291]. Most RNA chaperones identified so far are proteins that resolve misfolded RNA structures by binding stretches of double-stranded RNA with low-affinity and in a sequence-unspecific way. Other RNA chaperones bind single-stranded RNA and facilitate the transition from the incorrect to the correct structural conformation by lowering specific kinetic barriers [344].

Chaperone-assisted folding has been extensively studied for proteins, whereas comparatively little is known about the extent and mechanisms underlying chaperone-assisted RNA folding. What we know is that most of these proteins play a wide range of other functional roles in addition to being RNA chaperones and that they share no obvious similarities in terms of sequence and structure motifs [285]. Additionally, unlike protein-chaperones, RNA chaperons typically do not require any ATP to encourage refolding [312, 344–346].

Trans RNA-RNA interactions, i.e. interactions with other transcripts

Trans RNA-RNA interactions, i.e. interactions with other transcripts, involve the same elementary building blocks as RNA structure or cis RNA-RNA interactions, namely basepairs between pairs of complementary nucleotides. This implies that trans RNA-RNA interactions involve two single-stranded stretches of RNAs. They differ in that regard from protein-RNA interactions which may involve single-stranded or double-stranded RNA (and may happen in a sequence-specific or unspecific way).

If a single-stranded stretch of RNA sequence is to be bound in a sequence specific way, it should be much easier in terms of evolution to come up with a corresponding, near-complementary
RNA sequence than to devise an RNA-binding protein that would bind in an equally sequence-specific way. One would therefore expect trans RNA-RNA interactions to be more much abundant than sequence-specific protein-RNA interactions with single-stranded RNAs [118, 347].

Functionally important trans RNA-RNA interactions include the well-known class of microRNA-mRNA interactions which alter gene expression on mRNA level [348], interactions between snoRNAs and ribosomal RNAs which edit rRNAs before ribosome assembly [117] and snRNA-mRNA interactions which are key during mRNA splicing [115]. Both mRNA splicing and ribosome assembly can occur cotranscriptionally.

Large-scale transcriptome studies of higher organisms such as mouse and human show that a large fraction of the transcriptome does not encode any proteins, e.g. [349]. These non-coding transcripts are diverse with regard to length, expression patterns and levels and functional roles, if known. This has given rise to a wealth of different names for these transcripts which are commonly collectively referred to as non-coding RNAs (ncRNA).

One well-studied ncRNA example is the short DsrA sRNA (small RNA) in Escherichia coli which alters the structure of the rpoS mRNA upon binding, thereby enabling its translation. In order for this trans RNA-RNA interaction to happen, the structure of the ncRNA DsrA first needs to be destabilized by binding the Sm-like protein Hfq [350–353]. Several other examples of structure-mediated translation regulation via trans RNA-RNA interactions between a short ncRNA and an mRNA have been found, primarily in bacteria [354, 355]. The short ncRNA is often an anti-sense transcript of the corresponding mRNA, the trans RNA-RNA interaction typically involves a short stretch of near-complementarity and a protein is often required as third ingredient for the regulatory mechanism to be functional. Yet another example of a functionally relevant trans RNA-RNA interaction is the formation of the 30S ribosomal subunit in bacteria which requires the transient interaction with the leader sequence of the rRNA-operons [356].

Another well-studied example is the hok/sok toxin-antitoxin system in Escherichia coli which provides a mechanism for preservation of the R1 plasmid after cell division [357], see Figure 4.1. This system consists of three overlapping genes. The host-killing hok gene induces cell death upon translation of its protein. The mok (modulation of killing) gene overlaps hok on the same mRNA transcript, and translation of the mok reading frame must occur in order for translation of hok to occur. The sok (suppression of killing) gene encodes a short anti-sense RNA that binds and prevents translation of mok, and thus indirectly, also the translation of hok. In cells that possess the R1 plasmid, the unstable sok RNA is produced in high quantities, and prevents cell death caused by the longer lived hok RNAs. Following mitosis, the sok RNA is rapidly degraded in any daughter cells that lack the R1 plasmid, allowing the hok gene to induce cell death. The mechanism of
the hok/sok system depends on several structural features of the hok mRNA. Alternative structural
configurations reduce the degradation rate of the hok mRNA, and several transient hairpins at the
5'-end prevent binding of sok RNA during transcription [357].

4.2.5 Summary
The overall view that emerges is that the cotranscriptional folding pathways are determined both
by intrinsic features encoded in the RNA sequence itself such as transient and final structural fea-
tures, and by extrinsic features such as the speed of the transcribing polymerase, trans interaction
partners (e.g. proteins, ligands, RNA transcripts and other trans-interaction partners). In vivo, both
types of features are combined in the appropriate cellular context and determine the functional
RNA structure(s) being formed.

A range of experimental evidence supports the notion of fairly well-defined co-folding path-
ways in vivo. These pathways are on the one hand robust enough to guide the formation of the
correct functional RNA structure under typical cellular conditions, but are—if required—flexible
enough to yield different structural and functional outcomes, if the cellular environment signifi-
cantly changes [279].

4.3 Capturing cotranscriptional folding in methods for
RNA secondary structure prediction

4.3.1 Existing methods for RNA secondary structure prediction
A wide variety of computational methods already exist for predicting RNA structural features.
Most RNA structure prediction methods that can technically handle long, naturally occurring tran-
scripts such as rRNAs only aim to capture the RNA secondary structure rather than its tertiary
structure. Fortunately, many functional features can already be studied on this level of abstraction.
In the following, we therefore focus on methods for RNA secondary structure prediction (rather
than also covering methods for predicting tertiary RNA structure which are currently limited to
sequences of around 100 nt length).

Existing methods for predicting RNA secondary structure can be broadly grouped into two
categories. Those that take a single RNA sequence as input and those that work in a comparative
way by taking a set of homologous RNA sequences as input. There also exists a different class of
prediction methods that explicitly predict cotranscriptional folding pathways in terms of RNA sec-
ondary structure changes over time. They aim to capture the structure formation process in vivo and
are typically limited to analyzing transcripts of a few hundred nucleotides length. These methods are currently viewed as folding pathway prediction methods rather than RNA secondary structure prediction methods.

Comparative methods for RNA secondary structure prediction currently provide the state-of-art in terms of prediction accuracy, in particular for long RNA sequences. Apart from one recently introduced new method CoFold [162], none of the currently existing non-comparative or comparative methods for predicting RNA secondary structures, however, explicitly capture cotranscriptional folding or its overall effects.

In the following, we review the existing methods and propose ways of capturing some effects of cotranscriptional folding explicitly in order to further improve their prediction accuracy.

**Non-comparative, MFE methods for RNA secondary structure prediction**  Historically, non-comparative methods which take a single RNA sequence as input came first. These employ the so-called minimum-free energy (MFE) approach which aims to identify the (usually pseudoknot free) RNA secondary structure that minimizes the overall free Gibbs energy of the transcript. They include well-known methods such as MFOLD, RNAFOLD and related programs [152, 153, 202, 358]. These methods mirror the *in vitro* setting, where a fully synthesized RNA has infinite time to settle into its thermodynamically most favorable configuration. They implicitly assume that the functionally relevant secondary structure is the thermodynamically most stable one. Predictions are generated by efficiently searching the search space of all possible (usually, pseudoknot free) RNA secondary structure for the structure with the lowest overall MFE. This is typically done using a dynamic programming algorithm.

Several methods based on the suboptimal folding algorithm introduced by Wuchty et al. [359] have been developed which explicitly consider an ensemble of RNA secondary structures close to the minimum free energy. RNAsubopt, a program included in the ViennaRNA package [153, 201], provides a list of low energy secondary structures above a user-defined energy cutoff above the minimum free energy. Sfold [189, 360, 361] employs a statistical approach to sample RNA secondary structures from the ensemble of RNA secondary structures at thermodynamic equilibrium, where the probability that the algorithm picks a particular structure is proportional to the structure’s probability in the structural ensemble. While these methods consider structures that differ from the MFE configuration, they still assume that the RNAs are in thermodynamic equilibrium. Moreover, they ignore the kinetic nature of cotranscriptional formation, and the effect it may have on the resulting structure or ensemble of structures.
In 1996, Morgan and Higgs [156] investigated a set of long RNAs (comprising 16S rRNAs, 23S rRNAs and RNAseP) and found significant discrepancies between the evolutionarily conserved RNA structure features and the respective predicted MFE structures. They concluded that these differences “cannot simply be put down to errors in the free energy parameters used in the model” [156] and hypothesized that these may be due to effects of kinetic folding in vivo.

In order to test this hypothesis, Proctor and Meyer recently introduced the new RNA secondary structure prediction method COFOLD [162] which is the first to combine thermodynamic with kinetic considerations. They incorporate one overall effect of kinetic folding into a minimum free energy prediction method: the reachability of potential pairing partners during cotranscriptional folding. COFOLD demonstrates a significant performance improvement over minimum free energy methods alone, particularly for longer RNA sequences of more than 1000 nt for which one usually observes a marked decrease in prediction accuracy. Capturing this overall effect of cotranscriptional folding yields RNA secondary structures with similar, but slightly higher free energies compared to the MFE structure. These results promise that there may be great value in accounting for other effects of cotranscriptional folding to improve non-comparative methods for RNA secondary structure prediction.

Comparative methods for RNA secondary structure prediction Rapidly increasing amounts of genome sequencing data for a variety of organisms have given rise to a conceptually new approach to RNA secondary structure prediction that takes as input a set of homologous RNA sequences rather than a single RNA sequence of interest, e.g. [166, 221, 260, 264, 265, 362-372]. Even though these comparative methods differ considerably regarding their underlying algorithms, they all aim to identify the consensus RNA secondary structure that has been conserved during evolution. The underlying working hypothesis is that RNA structures that are functionally relevant should also be conserved. This assumption usually holds as RNA structures tend to be more conserved than the underlying primary sequences. Depending on the evolutionary distances among the input sequences, however, this approach may fail to detect species-specific structure features that have only developed recently.

Overall, comparative methods for RNA secondary structure prediction currently provide the state-of-art in terms of prediction accuracy. They tend to significantly outperform non-comparative methods [170], but typically require a high-quality input alignment provided by the user to reach their optimal performance (see, e.g. [364, 366, 367, 370, 372] for methods that do not require a fixed input alignment).
All of these methods generate predictions by first identifying pairs of covarying alignment columns to detect conserved basepairs and then combining these into a single (and, usually, global) consensus RNA secondary structure. For this, they either employ (1) a modified MFE framework which also accounts for conservation of basepairs and aims for overall energy minimization, (2) a probabilistic framework such as stochastic context-free grammars (SCFGs) combined with likelihood maximization, (3) a non-deterministic, yet probabilistic approach such as Bayesian Markov Chain Monte Carlos (MCMCs) that samples from a posterior distribution which is subsequently combined with a post-processing step to extract a consensus structure, or (4) a combination of heuristic, ad-hoc procedures.

### 4.3.2 Existing methods for predicting RNA folding pathways

In parallel to the development of the RNA secondary structure prediction methods, several methods have been developed that aim to explicitly simulate cotranscriptional structure formation as function of time. All of these methods, e.g. RNAKINETICS [160, 373, 374], KINFOLD [375], KINEFOLD [159, 297, 376] and KINWALKER [161], take as input a single RNA sequence and employ a range of different statistical models, approximations and heuristics to arrive at their predictions. Typically, they utilize stochastic simulation that extends the input RNA sequence at regular intervals, and simulates helix formation and disruption events over a simulated time scale. The probability that each event occurs is proportional to its theoretical chemical rate of change. They have, however, conceptual difficulties dealing with long sequences (over a few hundred nt) and their performance has until recently [143] been only benchmarked for a few select sequences. They are thus currently viewed as folding pathway prediction methods rather than RNA secondary structure prediction methods.

The recent study by Zhu et al. [143] utilizes three of these existing methods to show that evolutionarily related RNA sequences share common transient structural features during their predicted folding pathways, and that these features often coincide with known transient structures. The authors propose an analysis pipeline that applies several folding pathway prediction methods in a comparative manner by combining folding predictions across evolutionarily related RNA sequences. Moreover, this study provides solid evidence that some transient helices have been conserved during evolution.
4.3.3 Ideas for capturing cotranscriptional folding in methods for RNA secondary structure prediction

The key effect of cotranscriptional folding is to make the formation of the final structure depend on its wider context, both along the sequence and in terms of time.

The key feature common to all existing non-comparative and comparative methods for RNA secondary structure prediction is that they search the space of all possible (typically pseudoknot free) RNA secondary structure for the optimal structure without having any notion of a folding pathway or a time-wise ordering of events, see Figure 4.2. The recently introduced method CoFold [162] is an exception, yet it currently only models a single overall effect, namely the reachability of basepairing partners during cotranscriptional folding which effectively amounts to a re-weighing of different regions of the structure search space. The search of the structure space usually involves a scoring function whose overall value is being optimized during the search. The overall score for any candidate RNA structure is typically expressed as the sum or product of scores for individual structural building blocks that, taken together, cover the entire sequence. These elementary scores and the way in which they are combined by the scoring function during optimization, however, only depends on the local building blocks of the sub-sequence under consideration, but neither on their location within the sequence nor the RNA structure context of the surrounding sequence, see Figure 4.2. Most optimization algorithms are dynamic programming algorithms that combine optimal structures for adjacent sub-sequences into one optimal structure for the resulting merged sub-sequence. The order of these steps, however, does not replicate the events during cotranscriptional folding. In particular, no region of the theoretical structural search space is marked as unlikely, if the corresponding structure feature could not readily form cotranscriptionally in vivo, see Figure 4.2.

One of the intrinsic features that are known to influence the formation of RNA structure in vivo are transient structures as discussed earlier. As these features are encoded in the RNA sequence itself, they could in principle be detected by any method for RNA secondary structure prediction and subsequently used to bias the optimization process yielding the final RNA structure. Their detection could be implemented via a straightforward dynamic programming procedure that swiftly identifies all candidate helices (of some minimum length or stability) in the given input RNA sequence [140]. The conceptual problem is that these helices would naturally comprise both candidate transient helices as well as candidate helices of the final RNA secondary structure. These helices could be used in the optimization procedure in order to influence the local decision making (how to combine optimal structures for two sub-sequences into a single optimal structure for the
merged sub-sequence). This would be one conceptual way of taking the wider structure context into account during the optimization procedure yielding the predicted final RNA structure. In the spirit of [140], these modifications could for example penalize any candidate structure that has strong competing transient helices upstream which could jeopardize its cotranscriptional formation.

Whereas the identification of candidate helices and relevant competing helices for a single sequence may be complicated due to the relatively large search space, comparative methods may generate a more accurate and smaller set of evolutionarily conserved competing helices to consider, such as those output by the comparative helix finding algorithm TRANSAT [195]. If transient RNA structural features turn out to be evolutionarily conserved on a similar level to those of the final RNA structure, which is what recent results by Zhu et al. [143] indicate, however, this may actually lower the prediction accuracy of comparative RNA secondary structure prediction methods as they may erroneously incorporate these conserved transient helices into the predicted final RNA secondary structure. Whether or not this is the case and a cause for concern remains to be shown.

In addition to the ideas employed by CoFOLD [162] discussed above, the directionality of transcription could also be captured by rendering the scores assigned to the structural building blocks dependent on their position within the transcript, whether they are nearer to the 5’-end or the 3’-end.

It is less obvious how one should account for the speed of transcription, let alone variations of transcription speed and transcriptional pausing. At least for now, there is too little experimental information to hope to identify transcriptional pausing sites computationally. A change in overall transcription speed alters the ratio between the speed of transcript synthesis and the rate of structure formation. This has been experimentally shown to influence cotranscriptional folding pathways and their structural outcome. On the structure prediction side, the speed of transcription could be captured by altering the effective distances between structural features. This is exactly what the free parameter in CoFOLD [162] is for. By changing its value, one can effectively account for different (yet constant) transcription rates and thereby optimize the program’s performance for different species. If the transcription speed is high with respect to the rate of structure formation, the emerging transcript has less time and hence fewer opportunities to explore the surrounding structure space. This has the overall effect of enlarging effective distances, whereas a low transcription speed should have the overall effect of reducing effective distances.

Trans interactions of the transcript with other molecules comprise a biologically diverse set of interactions between the transcript and various other molecules. All of the existing methods for predicting RNA secondary structure including methods for folding pathway prediction assume an
isolated RNA sequence as input and ignore any potential trans interaction partners (the bulk effects of water and some ions is taken into account by most folding pathway prediction methods). If and how these trans interactions influence the cotranscriptional structure formation not only depends on the type of interaction (RNA-RNA, RNA-protein etc.), but also very much on the timing of the interaction with respect to the structure formation. For example, a protein that binds the emerging transcript early on and for a short time has a very different influence on structure formation than a protein that binds the final RNA structure only.

Early and persistent types of trans interactions could be captured in RNA secondary structure prediction methods by preventing the bound sub-sequence from engaging in other interactions, in particular other RNA structural features. Technically, this is fairly easy to achieve via a slight modification of the default optimization procedure by assigning a large penalty to all structure solutions that do not keep the bound sub-sequence single or double-stranded. This feature is already implemented by all RNA secondary structure prediction methods that allow known RNA structural features to be taken into account, e.g. [152, 221, 265]. This assumes, however, that details about the interaction site (sub-sequence, ssRNA versus dsRNA) are known up-front which is often not the case.

Any trans interactions of a more transient nature, however, are hard to capture computationally by any of the existing methods for RNA secondary structure prediction as this would require them to have some notion of time-ordered steps which they currently do not have.

Suggestions for further improving methods for folding pathway prediction

The existing folding pathway prediction methods already mimic the in vivo folding as they fold the RNA sequence cotranscriptionally at a constant transcription speed (which needs to be specified by the user). This is, however, only a first approximation of the complex in vivo situation. As these methods explicitly predict folding pathways, they already model cis RNA- RNA interactions and in particular transient RNA structural features. At least for now, these methods do not predict variations of transcription speed and do not capture potential trans interactions with other molecules from the in vivo environment.

If details about trans interactions are known up-front (timing, binding site, ssRNA versus dsRNA), these could be fairly easily captured by preventing the known binding site from engaging in other interactions. This has already been done for select examples and allowed us to computationally investigate the effect of trans interactions on cotranscriptional RNA structure formation [298].
4.4 Summary

With 90% of the human genome being transcribed [30–32], the investigation of transcriptomes and how they are regulated has never been more important. RNA structure is one important feature by which transcripts can influence their fate in the cell. There is by now ample experimental and solid theoretical evidence that RNA structure formation already starts during transcription and that events during the cotranscriptional folding determine which functional RNA structure(s) are being formed. Yet, as of now the process of structure formation is completely ignored by almost all state-of-the-art methods for RNA secondary structure prediction. We argue that capturing some aspects of the structure formation process in predictive models could significantly improve these methods and provide evidence for this in form of a new method [162]. These initial results are very encouraging as they show that a significant improvement in prediction accuracy can already be gained by modeling a single overall effect of cotranscriptional folding and without making the underlying prediction algorithm much more complex. Beyond this, we propose detailed ideas of how different aspects of cotranscriptional folding in vivo could also be captured in silico.

One of the most simple and encouraging messages from the mounting (and sometimes dauntingly complex) experimental results is certainly the realization that the transcript in the cell does not explore all of the structure search space.
Figure 4.1: RNA structure features for the reference sequence from *Escherichia coli* plasmid R1 encoding the hok and mok proteins. The horizontal line depicts the plasmid’s sequence with its nucleotides color-coded according to the legend on the top left. Underneath the sequence line, black arrows indicate the protein-coding regions of the hok and mok proteins. The grey arrow shows the sequence region that is complementary to the sok antisense RNA which is part of a different transcript. Each arc above the horizontal line represents a basepair between the two corresponding positions along the sequence and is color-coded according to the structure conformation to which it belongs (active, inactive or transient, see the legend on the top right). Below the horizontal sequence line, black lines indicate the location of known sequence motifs (tac, translational activator element; ucb, upstream complementary box; dcdb, downstream complementary box; mok SD, mok Shine-Dalgarno sequence; hok SD, hok Shine-Dalgarno sequence; fbi, fold-back inhibitory element). This arc-diagram was first published by Steif and Meyer [357] and generated using the R-CHIE web-server [256].

Figure 4.2: Examples of *cis* and *trans* interactions during cotranscriptional folding. A: Hypothetical RNA sequence, capable of forming helices $h_1$ to $h_4$, at sites A to E. B: Transcription of the sequence across time points $t_1$ to $t_5$, with the sequential lengthening of the 3'-end. The transcription process limits the available sites for helix formation, imposing an order on helix formation. If an early formed helix is stable, it can serve to block the formation of subsequent helices by occupying specific sites. C: Sites may also be occupied due to interactions with other molecules, in this case a protein binding site (PBS), occupies site A, leading to a very different result. D: If early helices are relatively unstable, they can be seen as transient helices that yield to new helices. This mechanisms can aid the robust formation of desired structure features. Note that some of the conformations shown above correspond to the ones introduced and defined by Meyer and Miklós [140]. These are: B $h_1$ (ii) and $h_3$ (ic) are 3' Trans, where $h_1$ is stable, preventing the formation of $h_3$ and $h_1$ (ii) and $h_2$ (ic) are 3' Cis, where $h_1$ is stable, preventing the formation of $h_2$. D $h_1$ (ci) and $h_2$ (ii) are 5' Cis, where $h_1$ is an intermediate for $h_2$ and $h_2$ (ci) and $h_3$ (ii) are 5' Cis, where $h_2$ is an intermediate for $h_3$. 

128
Chapter 5

Conclusion

Advances in high-throughput technologies have revealed an abundance of novel RNA species, and solving the structure of these species may aid researchers in characterizing and understanding them. With computational algorithms, we can swiftly predict basepairing structures, partners and interactions for these RNAs, highlighting candidates for further study and validation. In this thesis, I make contributions to the computational aspect of this problem.

In Chapter 2, I present a novel R package for RNA basepair processing, allowing for the easy manipulation of RNA sequence and basepair data. While algorithmically trivial, the availability of such a package can greatly aid the often tedious and mundane chore of writing ad hoc parsers and converters for the growing number of RNA basepair formats unique to every new prediction tool. Built upon the foundation of the R package R4RNA, I have already created the tool R-CHIE, capable of visualizing complex RNA basepair data in arc diagram formats, creating intuitive comparison plots not possible with typical methods of RNA secondary structure visualization. The linear format of the arc diagram also makes it ideal for overlaying on top of multiple sequence alignments, allowing users to see basepair conservation for each species at each position. Since its publication, the work has been well received, seeing adoption by the largest RNA structure database for visualizing basepair conservation status [204], displaying alternative structures [205, 357], comparing multiple structure [206], and contrasting two unique structures relative to chemical probing data [176].

In Chapter 3, I conduct an assessment of the current state-of-the-art for computational RNA-RNA interaction basepair prediction. To our knowledge, it is the largest and most comprehensive of its kind at the time of writing. With growing interest in determining ncRNA functionality, we feel it is important to establish the predictive accuracy of RNA-RNA interaction predictions
given sequences with known interacting basepairs. For predicting short intermolecular RNA-RNA interactions, we have shown that tools considering standard stacking energy stabilities in addition to accessibility score penalties consistently perform the best. Contrary to RNA secondary structure prediction, the usage of conservation information does not result in a sufficient gain in predictive performance compared to the aforementioned method, and may even decrease the performance if used with accessibility. The inability of all tools to maintain a high performance as input length increases continues to be a major issue, with worrisome implications when applying these methods to whole-genome interaction partner searches. Overall, we believe that in their current states, computational RNA-RNA interaction can serve as a rough tool for predictions that are better than random guesses, but fall short of replacing experimental methods.

In Chapter 4, we present a review on cotranscriptional folding in RNA basepair prediction, highlighting the biological and computational status of this process. We present the first review outlining the experimental and statistical evidence the field has for cotranscriptional folding, the implications it has on RNA basepair formation, and how computational methods have and can use this information. Biologically, we provide known examples of both cis- and trans- interactions with the RNA as its folds during transcription, including metabolites, ions, proteins, chaperones, upstream nucleotides, and nucleotides of other RNA sequences. Computationally, we review the current methods RNA secondary structure prediction, starting with classical energy-based methods, comparative and hybrid methods taking sequence alignments, and the small set of tools simulating dynamic RNA folding pathways. Finally, we conclude with some ideas of incorporating RNA co-transcriptional knowledge into RNA prediction algorithms, with the main example being CoFOLD [162]. Created by co-author Jeff Proctor, CoFOLD incorporates some overall effects of cotranscriptional folding directly into the structure prediction process thereby significantly increasing the prediction accuracy especially for long RNA sequences.

As a whole, the thesis positions itself at the junction between what is known, and what we can do in the field of computational RNA basepair prediction. It is often foolish to proceed down a path of research without knowing the current state of affairs, and my thesis hopes to serve as a beacon for future works to come.
Bibliography


132


134


141


146


148


151


155


156


Appendix A

Supporting Materials

A.1 Chapter 1 appendix material

A.1.1 R4RNA function manual
Package ‘R4RNA’

October 13, 2015

Type Package
Title An R package for RNA visualization and analysis
Version 0.99.2
Date 2015-10-13
Author Daniel Lai, Irmtraud Meyer
Maintainer Daniel Lai <redacted@example.com>
Depends R (>= 3.2.0)
Description Plots are diagrams for RNA secondary structure and alignments
License GPL-3

biocViews Alignment, MultipleSequenceAlignment, Preprocessing,
Visualisation, DataImport, DataRepresentation

URL http://www.e-rna.org/r-chie/

R topics documented:

R4RNA-package ......................................................... 2
Alignment Statistics .................................................... 3
Basepair Frequency ...................................................... 4
Basepair/Helix Conversion ............................................. 5
Coerce to Helix ......................................................... 6
Colour Helices ......................................................... 7
Convert Helix Formats ............................................... 9
Covariation Plots ...................................................... 10
Create Blank Plot ...................................................... 13
Example Data .......................................................... 14
Find Unknotted Groups .............................................. 15
Helix Type Filters ..................................................... 16
Log10 Space Operations .............................................. 17
Plot Helix Structures ................................................. 18
Read FASTA .......................................................... 20
Read Structure File .................................................... 20
Structure Mismatch Score .......................................... 22
Write FASTA .......................................................... 23
Write Helix ............................................................ 24

Index 25
R4RNA-package

An R package for RNA visualization and analysis

Description

An R package for RNA visualization and analysis

Details

Package: R4RNA
Type: Package
Version: 0.99.2
Date: 2015-10-13
License: GPL-3

Author(s)

Maintainer: Daniel Lai,
Irmtraud Meyer

Examples

# Read input data
predicted <- readHelix(system.file("extdata", "helix.txt", package = "R4RNA"))
known <- readVienna(system.file("extdata", "vienna.txt", package = "R4RNA"))
sequence <- readFasta(system.file("extdata", "fasta.txt", package = "R4RNA"))

plotHelix(predicted)
pval.coloured <- colourByValue(predicted, log = TRUE, get = TRUE)
plotDoubleHelix(pval.coloured, known, scale = FALSE)
plotOverlapHelix(pval.coloured, known)

cov.coloured <- colourByCovariation(known, sequence, get = TRUE)
plotCovariance(sequence, cov.coloured)

plotDoubleCovariance(cov.coloured, pval.coloured, sequence,
                     conflict.filter = "grey")
plotOverlapCovariance(pval.coloured, known, sequence, grid = TRUE,
                      conflict.filter = "grey", legend = FALSE, any = TRUE)

# List of all functions
ls("package:R4RNA")
# use example() and help() for more details on each function
Alignment Statistics

Description
Functions to compute covariation, percent identity conservation, and percent canonical basepairs given a multiple sequence alignment and optionally a secondary structure. Statistics can be computed for a single base, basepair, helix or entire alignment.

Usage

```r
baseConservation(msa, pos)
basepairConservation(msa, pos.5p, pos.3p)
basepairCovariation(msa, pos.5p, pos.3p)
basepairCanonical(msa, pos.5p, pos.3p)

helixConservation(helix, msa)
helixCovariation(helix, msa)
helixCanonical(helix, msa)

alignmentConservation(msa)
alignmentCovariation(msa, helix)
alignmentCanonical(msa, helix)
alignmentPercentGaps(msa)
```

Arguments

- `helix` A helix data.frame
- `msa` A multiple sequence alignment, such as those returned by `readFasta`
- `pos, pos.5p, pos.3p` Positions of bases or basepairs for which statistics shall be calculated for.

Details

Conservation values have a range of \([0, 1]\), where 0 is the absence of primary sequence conservation (all bases different), and 1 is full primary sequence conservation (all bases identical).

Canonical values have a range of \([0, 1]\), where 0 is a complete lack of basepair potential, and 1 indicates that all basepairs are valid.

Covariation values have a range of \([-2, 2]\), where -2 is a complete lack of basepair potential and sequence conservation, 0 is complete sequence conservation regardless of basepairing potential, and 2 is a complete lack of sequence conservation but maintaining full basepair potential.

`helix` values are average of base/basepair values, and the `alignment` values are averages of helices or all columns depending on whether the `helix` argument is required.

`alignmentPercentGaps` simply returns the percentage of nucleotides that are gaps in a sequence for each sequence of the alignment.
Basepair Frequency

Value

baseConservation, basepairConservation, basepairCovariation, basepairCanonical, alignmentConservation,
alignmentCovariation, and alignmentCanonical return a single decimal value.
helixConservation, helixCovariation, helixCanonical return a list of values whose length
equals the number of rows in helix.
alignmentPercentGaps returns a list of values whose length equals the number of sequences in
the multiple sequence alignment.

Author(s)

Jeff Proctor, Daniel Lai

Examples

data(helix)
baseConservation(fasta, 9)
basepairConservation(fasta, 9, 18)
basepairCovariation(fasta, 9, 18)
basepairCanonical(fasta, 9, 18)
helixConservation(helix, fasta)
helixCovariation(helix, fasta)
helixCanonical(helix, fasta)
alignmentConservation(fasta)
alignmentCovariation(fasta, helix)
alignmentCanonical(fasta, helix)
alignmentPercentGaps(fasta)

Basepair Frequency

Calculates the frequency of each basepair

Description

Calculates the frequency of each basepair in a given helix structure. Internally, breaks helices into
basepairs, and returns a structure of unique basepairs, where the values is its frequency, regardless
of original value.

Usage

basepairFrequency(helix)

Arguments

helix A helix data.frame

Value

A helix data.frame of unique basepairs of length 1, with the frequency of appearance as its value,
sorted by decreasing value.
Basepair/Helix Conversion

Author(s)
Daniel Lai

See Also
colourByBasepairFrequency

Examples

data(helix)
basepairFrequency(helix)

Description
Given a helix data frame, expands a helix of arbitrary length into helices of length 1 (i.e. basepairs). Also does the reverse operation of clustering consecutive basepairs (or helices), and merging/collapsing them into a single helix.

Usage

expandHelix(helix)
collapseHelix(helix, number = FALSE)

Arguments

helix A helix data frame.
number Indicates presence of a column in the helix data frame titled exactly 'number’, which will be used to unique identify basepairs belonging to the same helix. Only basepairs from the same helix as identified by the number will be collapsed together.

Details
During the expansion, basepairs expanded from a single helix will all be assigned the value of the originating helix (the same goes for all other columns besides i, j, and length). During collapsing, only helices/basepairs of equal value will be grouped together. The ordering of collapsed helices returned will be sorted by value (increasing order). For any other columns besides i, j, length and value, values will be obtained from the corresponding columns of the outer most basepair.

Value
Returns a helix data frame.

Author(s)
Daniel Lai
Examples

# Create helix data frame
helix <- data.frame(2, 8, 3, 0.5)
helix[2,] <- c(5, 15, 4, -0.5)
helix <- as.helix(helix)
helix$colour <- c("red", "blue")

# Before expansion
print(helix)
# After expansion
print(expanded <- expandHelix(helix))
# Collapse back (sorted by value)
print(collapseHelix(expanded))

---

Coerce to Helix  Coerce to a Helix Data Frame

Description

Functions to coerce a structure into a helix data frame, and to check whether a structure is a valid helix data frame. A helix data frame is a data frame, so any structure coercible into a data.frame can become a helix data frame.

Usage

as.helix(x, length)
is.helix(x)

Arguments

x Structure to coerce. Should be a structure coercible into a standard R data.frame structure for as.helix. Should be a string for parseBracket. May be anything for is.helix.
length The length of the RNA sequence containing the helices.

Details

as.helix takes in a data.frame and coerces it into a helix data frame acceptable by other R4RNA functions. This mainly involves setting specific column names and casting to specific types.

Value

is.helix returns a boolean.

as.helix returns helix data frame with valid input.

Author(s)

Daniel Lai
Colour Helices

Examples

# Not a valid helix data frame
helix <- data.frame(c(1, 2, 3), seq(1/zero.noslash, 2/zero.noslash, length.out = 3), 5, runif(3))
is.helix(helix)
warnings()

# Formatted into a helix data frame
helix <- as.helix(helix)
is.helix(helix)

---

Assign colours to helices

Description

Functions to generate colours for helices by various rules, including integer counts, value ranges, percent identity covariation, conservation, percentage canonical basepair, basepair frequency, and non-pseudoknotted groups.

Usage

colourByCount(helix, cols, counts, get = FALSE)
colourByValue(helix, cols, breaks, get = FALSE, log = FALSE, include.lowest = TRUE, ...)
colourByBasepairFrequency(helix, cols, get = TRUE)
colourByUnknottedGroups(helix, cols, get = TRUE)
colourByCovariation(helix, msa, cols, get = FALSE)
colourByConservation(helix, msa, cols, get = FALSE)
colourByCanonical(helix, msa, cols, get = FALSE)
defaultPalette()

Arguments

- **helix**: A helix data frame to be coloured.
- **cols**: An array of characters (or numbers) representing a set of colours to colour helix with. When missing, a default set of colours from defaultPalette() will be used. Valid input include hex codes, colour names from the colours function, and integer numbers. The colours will be interpreted as being from best to worst.
- **counts**: An array of integers the same length as cols, dictating the number of times each corresponding colour should be used. When missing, the function will divide the number of helices evenly over each of the colours available.
- **breaks**: An integer number of intervals to break the ‘value’ column of helix into, or a list of numbers defining the interval breaks. If missing, the range of ‘helix$value’ will automatically be split evenly into intervals for each colour available.
- **get**: If TRUE, returns the input helix with a col column, else simply returns an array of colours the same length as the number of row in helix. The exceptions are colourByBasepairFrequency and colourByUnknottedGroups which will return a different helix if TRUE, and a list of colours that will not match the input helix if FALSE.
Colour Helices

log If TRUE, will breaks values into even log10 space intervals, useful when values are p-values.
include.lowest Whether the lowest interval should include the lowest value, passed to cut
... Additional arguments passed to cut, potentially useful ones include right (whether intervals should be inclusive on the right or left) and dig.lab (number of digits in interval labels).
msa A multiple sequence alignment, such as those returned by readFasta

Details
colourByCount assigns colours independent of the helix input's value column, and instead operates over the number of helices (i.e. rows).
colourByValue uses cut to assign each of the helices to an interval based on its value.
colourByCovariation, colourByConservation, and colourByCanonical, colour helices according to compensatory mutations (or covariation), percentage identity conservation, and percentage canonical basepair respectively, relative to the multiple sequence alignment provided.
colourByBasepairFrequency colours each basepair according to the number of times it appear in the input, regardless of its value.
colourByUnknottedGroups greedily partitions the basepairs into non-pseudoknotted groups, and assigns a colour to each.

Value
All "colourBy" functions return a list of colours when get = FALSE, and a helix with a col column if get = TRUE. In both bases, the returned object has attributes "legend" and "fill", showing the mapping between interval (in legend) and colour (in fill), which can as eponymous arguments legend.
defaultPalette returns the default list of colours.

Author(s)
Daniel Lai

See Also
plotHelix
logseq
basepairFrequency
unknottedGroups

Examples
data(helix)

known$col <- colourByCount(known)
plotHelix(known)

plotHelix(colourByValue(helix, log = TRUE, get = TRUE))
cov <- colourByCovariation(known, fasta, get = TRUE)
plotCovariance(fasta, cov)
Convert Helix Formats

Description

Converts dot bracket vienna format to and from helix format. It should be noted that the allows structures of vienna is a subset of those allowed in the helix format. Thus, conversion from vienna to helix will yield the identical structure, while conversion from helix to vienna may result in the loss of certain basepairs (mainly those that are conflicting). Pseudoknots are supported in both directions of conversion with limitations.

Usage

viennaToHelix(vienna, value = NA, palette = NA)
helixToVienna(helix)
helixToConnect(helix)
helixToBpseq(helix)

Arguments

vienna A string containing only a vienna dot bracket structure, with balanced brackets. Allowable brackets are (, <, [, {, A, B, C, and D (where upper-case alphabets are paired with lower-case alphabets).
value A numerical value to assign to all helices.
palette A list of colour names for up to 8 colours that will be used to colour brackets of type (, <, [, {, A, B, C, and D, respectively.

Details

viennaToHelix will ignore any non dot-bracket characters prior to parsing, so the resultant length will be shorter than expected if invalid characters are included.

If the colour palette is less than the number of supported brackets, it will simply cycle through the list. To explicitly prevent the colouring/display of specific bracket type, colour it “NA”.

For helixToVienna, pseudoknotted basepairs will be assigned different bracket types. As there are only 8 supported bracket types, any basepair pseudonotted deeper than 8 levels will be excluded from the output. Additionally, vienna format is unable to respresent conflicting basepairs, so conflicting basepairs will also be excluded. For both types of exclusion, those at the bottom of the helix data.frame will always be excluded in favour of keeping helices higher on the data.frame table.

helixToConnect and helixToBpseq will convert a non-conflicting helix data.frame into connect or bpseq format respectively, provided the helix structure has a “sequence” attribute containing a single nucleotide sequence of the structure.
Covariation Plots

Value

`viennaToHelix` returns a helix data.frame. `helixToVienna` returns a character string of basepairs in the Vienna helix format. `helixToConnect` and `HelixTpBpseq` return data.frames in the connect and bpseq formats, respectively.

Author(s)

Daniel Lai

Examples

```r
# viennaToHelix demonstrating ALL valid bracket symbols
dot_bracket <- ".....<[[.....ABCD.....]>....dcba....."
parsed <- viennaToHelix(dot_bracket, -31.5)
print(parsed)

vienna <- helixToVienna(parsed)
print(vienna)

# Colouring the brackets by bracket type
colour <- c("red", "orange", "yellow", "green", "lightblue", "blue", "purple", "black")
double.rainbow <- viennaToHelix(dot_bracket, /zero.noslash, colour)
plotHelix(double.rainbow)
```

Covariation Plots

Plot nucleotide sequence coloured by covariance

Description

Given a multiple sequence alignment and a corresponding secondary structure, nucleotides in the sequence alignment will be coloured according to the basepairing and conservation status, where green is the most commonly observed valid basepair in the column, dark blue being valid covariation (i.e. mutation into another valid basepair), cyan is one-sided mutation that retains the basepair, and red is a mutation where the basepair has been lost.

Usage

```r
plotCovariance(msa, helix, arcs = TRUE, add = FALSE, grid = FALSE, text = FALSE, legend = TRUE, species = 0, base.colour = FALSE, palette = NA, flip = FALSE, grid.col = "white", grid.lwd = 0, text.cex = 0.5, text.col = "white", text.font = 2, text.family = "sans", species.cex = 0.5, species.col = "black", species.font = 2, species.family = "mono", shape = "circle", conflict.cutoff = 0.01, conflict.lty = 2, conflict.col = NA, pad = c(0, 0, 0, 0), y = 0, x = 0, ...
```

```r
plotDoubleCovariance(top.helix, bot.helix, top.msa, bot.msa = top.msa, add = FALSE, grid = FALSE, species = 0, legend = TRUE, pad = c(0, 0, 0, 0), ...
```

```r
plotOverlapCovariance(predict.helix, known.helix, msa, bot.msa = TRUE, overlap.cutoff = 1, miss = "black", add = FALSE, grid = FALSE, species = 0, legend = TRUE, pad = c(0, 0, 0, 0), ...
```
Arguments

- **msa, top.msa, bot.msa**
  Multiple sequence alignment as an array of named characters, all of equal length. Typically output of `readFasta`
  `top.msa` and `bot.msa` are specific to `top.helix` and `bot.helix` respectively, and may be set to NA to have no multiple sequence alignment at all.

- **helix, top.helix, bot.helix, predict.helix, known.helix**
  A helix data.frame with a structure corresponding to `msa`. See `plotDoubleHelix` and `plotOverlapHelix` for detailed explanations of `top.helix`, `bot.helix`, `predict.helix`, and `known.helix`.

- **arcs**
  TRUE if the structure should be plotted as arcs. Arcs may be styled with styling columns, see example and `plotHelix` for details.

- **add**
  TRUE if graphical elements are to be added to an existing device, else a new plotting device is created with `blankPlot`.

- **grid**
  TRUE if the multiple sequence alignment is to be drawn as a grid of bases, else the multiple sequence alignment is drawn as equidistant horizontal lines.

- **text**
  Only applicable when grid is TRUE. TRUE if the grid is to be filled with nucleotide character.

- **legend**
  TRUE if legend are to be shown.

- **species**
  If a number greater than 0 is given, then species names for the multiple sequence alignment will be printed along the left side. This name is typically the entire header lines of FASTA entries from `readFasta`, and can be manually manipulated using the `names` function. The number specifies the start position relative to the left edge of the multiple sequence alignment.

- **base.colour**
  TRUE if bases are to be coloured by nucleotide instead of basepair conservation.

- **palette**
  A list of colour names to override the default colour palette. When `base.colour` is TRUE, the first 6 colours will be used for colouring bases A, U, G, C, -(gap), and ? (everything else), respectively. When `base.colour` is FALSE, the first 7 colours will be used for colouring conserved basepairs, covarying basepairs, one-sided conserved basepairs, invalid basepairs, unpaired bases, gaps, and bases/pairs with ambiguous bases, respectively. If the palette is shorter than the expected length, the palette will simply cycle. “NA” is a valid colour, that will effectively plot nothing.

- **flip**
  If TRUE, the entire plot will be flipped upside down. Note that this is not a perfect mirror image about the horizon.

- **grid.col, grid.lwd**
  The colour and line width of the borders displayed when grid is TRUE.

- **text.cex, text.col, text.font, text.family**
  cex, col, family and font for the text displayed via the text option. Use `help("par")` for more information the paramters.

- **species.cex, species.col, species.font, species.family**
  cex, col, family and font for the species text displayed via the species option. Use `help("par")` for more information the paramters.

- **shape**
  One of "circle", "triangle", or "square", specifying the shape of the arcs.

- **conflict.lty, conflict.col, conflict.cutoff**
  Determines the line type (style) and colour to be used for conflicting basepairs. By default, conflicting helices are drawn as dotted lines (lty = 2) and whatever colour was originally assigned to it (col = NA). Conflicting helices may be
coloured by setting `conflict.col` to some R-compatible colour name. If both arguments are set to `NA`, then no attempt to exclude conflicting helices will be made when colouring covariance plot columns, which in most cases will render the plot nonsensical. When the input has helices with multiple basepairs, and only part of the helix is conflicting, the `conflict.cutoff` determines above what percentage of basepairs have to be conflicting before a helix is considered conflicting, with the default set at 1 conflicting).

`miss` The colour for unpredicted arcs in overlapping diagrams, see `plotOverlapHelix` for more information.

`overlap.cutoff` Decimal between 0 and 1 indicating the percentage of basepairs within a helix that have to be overlapping for the entire helix to count as overlapping. Default is 1, or 100

`pad` A four integer array passed to `blankPlot`, specifies the number of pixels to pad the bottom, left, top and right sides of the figure with, respectively.

`x, y` Coordinates for the left bottom corner of the plot. Useful for manually positioning and overlapping figure elements.

`...` In `plotCovariance`, these are additional arguments passed to `blankPlot`, useful arguments include `lwd`, `col`, `cex` for line width, line colour, and text size, respectively. `help(par)` for more.

For `plotDoubleCovariance` and `plotOverlapCovariance`, these are additional arguments passed to `plotCovariance` (and thus indirectly also to `blankPlot`).

**Value**

Not intended to return a value, will plot to GUI or file if specific.

**Author(s)**

Daniel Lai

**See Also**

- `plotHelix`
- `plotDoubleHelix`
- `plotOverlapHelix`
- `colourByCovariation`
- `colourByConservation`
- `colourByCanonical`

**Examples**

```r
data(helix)

# Basic covariance plot
plotCovariance(fasta, known, cex = 0.8, lwd = 1.5)

# Grid mode
plotCovariance(fasta, known, grid = TRUE, text = FALSE, cex = 0.8)

# Global style and nucleotide colouring
plotCovariance(fasta, known, grid = TRUE, text = FALSE, base.colour = TRUE)
```
# Styling individual helices with styling columns
known$col <- c("red", "blue")
plotCovariance(fasta, known, lwd = 2, cex = 0.8)

# Use in combination with colourBy functions
cov <- colourByCovariation(known, fasta, get = TRUE)
plotCovariance(fasta, cov)
legend("topleft", legend = attr(cov, "legend"),
     fill = attr(cov, "fill"), title = "Covariation")

---

**Description**

Creates a blank plotting canvas with the given dimensions, along with functions to find best values for the canvas dimensions.

**Usage**

```r
blankPlot(width, top, bottom, pad = c(0, 0, 0, 0), scale = TRUE,
          scale.lwd = 1, scale.col = "#DDDDDD", scale.cex = 1, debug = FALSE,
          png = NA, pdf = NA, factor = ifelse(!is.na(png), 8, 1/9),
          no.par = FALSE, asp = 1, ...
)
maxHeight(helix)
```

**Arguments**

- `width`: A number indicating the horizontal width of the blank plot.
- `top, bottom`: The maximum and minimum values vertically to be displayed in the plot.
- `pad`: An array of 4 integers, specifying the pixels of whitespace to pad beyond the dimensions given by top, bottom, and width. Four number corresponding to padding on the bottom, left, top and right, respectively. Default is `c(0, 0, 0, 0)`.
- `scale`: If TRUE, inserts a scale on the plot.
- `scale.lwd, scale.col, scale.cex`: Allows manual modification of the scale’s line width and colour, respectively.
- `png, pdf`: If one or the other is set to a filename, a file in png or pdf format will be produced respectively. If both are set to non-NA values, png will have priority.
- `factor`: The scaling factor used to produce plots of png or pdf format. Should be set so after multiplication of the top, bot, etc arguments, good document dimension in pixels with png and inches for pdf will be produced.
- `debug`: If TRUE, frames the boundaries of the intended plotting space in red, used to determine if inputs produce expected output area. Also outputs to STDIN dimensions of the plot.
- `no.par`: Suppresses the internal call to par in the function if set to TRUE, useful for using par arguments such as mfrow, etc.
- `asp`: Controls and aspect ratio of the plot, defaultly set to 1, set to NA to disable completely.
Additional arguments passed to `par` when `no.par` is FALSE, common ones include ‘lwd’, ‘col’, ‘cex’ for line width, line colour, and text size, respectively. `help(par)` for more. When `no.par` is set to TRUE, this option does nothing, and manually calling `par` is required prior to the calling of this function.

`helix` A helix data.frame

**Details**

`blankPlot` creates a blank plot with the given dimensions, with minimal margins around the plot and no axis or labels. If more control is required, using `plot` directly would be more efficient.

`maxHeight` returns the height that the highest helix would require, and can be used to determine `top` and `bottom` for `blankPlot`.

**Value**

`maxHeight` returns a numeric integer.

**Author(s)**

Daniel Lai

**See Also**

`plotHelix`

**Examples**

```r
# Create helix and obtain height
bsp <- as.helix(data.frame(1, 37, 12, 0.5))
height <- maxHeight(helix)
print(height)

# Use height to create properly sized plot
width <- attr(helix, "length")
blankPlot(width, height, 0)

# Add helix to plot
plotHelix(helix, add = TRUE)
```

**Description**

This data set contains two sets of helices and a multiple sequence alignment. The two sets of helices are `helices` and `known` which are helices predicted to occur for RNA sequence RF00458 by the program TRANSAT, and experimentally proposed structure of the same sequence, respectively. `fasta` is the seed homologues for the multiple sequence alignment obtained from the RFAM database.

**Usage**

`data(helix)`
Find Unknotted Groups

Format

helix and known are 4 column data frames, where columns i and j denote the left-most and right-most basepairs, the length is the number of consecutive basepairs the helix contains, and the value is assigned to each helix on a row.

fasta is an array of named characters of length 7.

References


Find Unknotted Groups  Partition basepairs into unknotted groups

Description

Breaks down input helices into basepairs, and assigns each basepair to a numbered group such that basepairs in each group are non-pseudoknotted relative to all other basepairs within the same group. The algorithm is greedy and thus will not find the best combination of basepairs to minimize the number of groups.

Usage

unknottedGroups(helix)

Arguments

helix A helix data.frame.

Value

An array of integers dictating the groups of each helix. Will only correspond to the input helix structure if the input had helices of length 1 (e.g. output of expandHelix).

Author(s)

Daniel Lai

See Also

colourByUnknottedGroups
expandHelix

Examples

data(helix)
known$group <- unknottedGroups(known)
print(known)
Description
Given a helix data frame, checks if helices are conflicting, duplicating, or overlapping, and returns an array of numeric values, where 0 is FALSE and 1 is TRUE. Values in between 0 and 1 occur when a single helix has multiple basepairs with different values, the number observed in this case is the mean of the basepair values within the helix. See details for exact definition of the three types of events.

Usage

```r
isConflictingHelix(helix)
isDuplicatingHelix(helix)
isOverlappingHelix(helix, query)
```

Arguments

- `helix`: A helix data frame
- `query`: For `isOverlappingHelix`, a helix data structure against which `helix` will be checked for overlap against.

Details

Helices of length greater than 1 are internally expanded into basepairs of length 1, after which the following conditions are evaluated:

A **conflicting** basepair is one where at least one of its two positions is used by either end of another basepair.

A **duplicating** basepair is one where both of its positions are used by both ends of another basepair.

An **overlapping** basepair is one in `helix` where both of its positions are used by both ends of another basepair in the `query` structure.

In the case of conflicting and duplicating basepairs, for a set of basepairs that satisfies this condition, the basepair situation highest on the data frame will be exempt from the condition. i.e. Say 5 basepairs are all duplicates of each other, the top 1 will return FALSE, while the bottom 4 will return TRUE. This assumes some significant meaning to the ordering of rows prior to using this function. This is to be used with `which` to filter out basepairs that satisfy these conditions, leaving a set of basepairs free of these events.

If the original input had helices greater than length 1, then after applying all of the above, TRUE is treated as 1, FALSE as 0, and the average of values from each basepair is taken as the value for the helix in question.

Value

Returns an array of numerics corresponding to each row of `helix`, giving the average conditional status of the helix, where 0 signifying all basepairs are FALSE, and 1 where all basepairs are TRUE.

Author(s)

Daniel Lai
Examples

data(helix)
conflicting <- isConflictingHelix(helix)
duplicating <- isDuplicatingHelix(helix)

# Nonsensical covariation plot
plotCovariance(fasta, helix)

# Plot nonconflicting helices
plotCovariance(fasta, helix[(!conflicting & !duplicating), ])

# Similar result
plotCovariance(fasta, helix, conflict.col = "lightgrey")

Description

Sequence, floor and ceiling operations in log 10 space.

Usage

logseq(from, to, length.out)
logfloor(x)
logceiling(x)

Arguments

from, to  Positive non-zero values to start and end sequence, respectively.
length.out The number of elements the resulting sequence should contain. If absent, function will attempt to generate numbers factors of 10 apart.
x          A value to round.

Value

logseq returns an array numbers evenly distanced in log10-space.
logfloor and logceiling return a value that is 10 raised to an integer number.

Author(s)

Daniel Lai

Examples

logseq(1e-10, 1e3)
logseq(1e-10, 1e3, length.out = 10)
logceiling(2.13e-6)
logfloor(2.13e-6)
Plot Helix Structures

Plots helices in arc diagram

Description

Plots a helix data frame as an arc diagram, with styling possible with properly named additional columns on the data frame.

Usage

```r
plotHelix(helix, x = /zero.noslash, y = /zero.noslash, flip = FALSE, line = FALSE, arrow = FALSE, add = FALSE, shape = "circle", ...)
plotDoubleHelix(top, bot, line = TRUE, arrow = FALSE, add = FALSE, ...)
plotOverlapHelix(predict, known, miss = "black", line = TRUE, arrow = FALSE, add = FALSE, overlap.cutoff = 1, ...)
plotArcs(i, j, length, x = /zero.noslash, y = /zero.noslash, flip = FALSE, shape = "circle", ...)
plotArc(i, j, x = /zero.noslash, y = /zero.noslash, flip = FALSE, shape = "circle", ...)
```

Arguments

- `helix, top, bot, predict, known` Helix dataframes, with the four mandatory columns. Any other column will be considered a styling column, and will be used for styling the helix. See example for styling usage. See Details for exact usage of each helix.
- `x, y` The coordinate of the left bottom corner of the plot, useful for manually positioning figure elements.
- `flip` If TRUE, flips the arcs upside down about the y-axis.
- `line` If TRUE, a horizontal line representing the sequence is plotted.
- `arrow` If TRUE, an arrow is played on the right end of the line.
- `add` If TRUE, graphical elements are added to the active plot device, else a new plot device is created for the plot.
- `shape` One of "circle", "triangle", or "square", specifying the shape of the arcs.
- `miss` The colour for unpredicted arcs in overlapping diagrams, see details for more information.
- `overlap.cutoff` Decimal between 0 and 1 indicating the percentage of basepairs within a helix that have to be overlapping for the entire helix to count as overlapping. Default is 1, or 100
- `i, j` The starting and ending position of the arc along the x-axis
- `length` The total number of arcs to draw by incrementing `i` and decrementing `j`. Used to draw helices.
- `...` Any additional parameters passed to `par`
Details

plotHelix creates a arc diagram with all arcs on top, plotDoubleHelix creates a diagram with arcs on the top and bottom. plotOverlapHelix is slight trickier, and given two structures predict and known, plots the predicted helices that are known on top, predicted helices that are not known on the bottom, and finally plots unpredicted helices on top in the colour defined by miss.

plotArc and plotArcs are the core functions that make everything work, and may be used for extreme fine-tuning and customization.

Value

Not intended to return a value, will plot to GUI or file if specific.

Author(s)

Daniel Lai

See Also

colourByCount

Examples

data(helix)

# Plot helix plain
plotHelix(known)

# Apply global appearance options
plotHelix(known, line = TRUE, arrow = TRUE, col = "blue", lwd = 1.5)

# Add extra column with styling options
known$lty <- 1:4
known$lwd <- 1:2
known$col <- c(rgb(1, 0, 0), "orange", "yellow", "#00FF00", 4, "purple")
plotHelix(known)

# Manually colour helices according to value
helix$col <- "red"
helix$col[which(helix$value < 1e-3)] <- "orange"
helix$col[which(helix$value < 1e-4)] <- "green"
helix$col[which(helix$value < 1e-5)] <- "blue"
plotHelix(helix)

# Automatically creating a similar plot with legend
coloured <- colourByValue(helix, log = TRUE, get = TRUE)
plotHelix(coloured, line = TRUE, arrow = TRUE)
legend("topleft", legend = attr(coloured, "legend"),
       fill = attr(coloured, "fill"), title = "P-value", text.col = "black")

# Plot both helices with styles
plotDoubleHelix(helix, known)

# Overlap helix
plotOverlapHelix(helix, known)
Read FASTA

Description
Reads in FASTA format multiple sequence text files into a list of named characters, with names derived from the description line of each FASTA entry.

Usage
readFasta(file, filter = FALSE)

Arguments
file
FASTA format file containing at least one sequence, where each sequence has a description line beginning with the > character.

filter
When true, filters out any sequences with any other characters besides: A, C, G, T, U, and - for gaps. Also converts all letter to uppercase and all T’s to U’s, and N’s to -’s.

Value
Returns an array of named characters, each element a sequence read, with the description as its name.

Author(s)
Daniel Lai

Examples
file <- system.file("extdata", "fasta.txt", package = "R4RNA")
fasta <- readFasta(file)
head(fasta)

Read secondary structure file

Description
Reads in secondary structure text files into a helix data frame.

Usage
readHelix(file)
readConnect(file)
readVienna(file, palette = NA)
readBpseq(file)
Read Structure File

Arguments

file A text file in connect format, see details for format specifications.
palette Used to colour basepairs by bracket type. See viennaToHelix for more details.

Details

Helix: Files start with a header line beginning with # followed by the sequence length, followed by a four-column tab-delimited table (with column names), where each row corresponds to a helix in the structure. The four columns are i and j for the left-most and right-most basepair positions respectively, the length of the helix (converging inwards from i and j, and finally an arbitrary value assigned to the helix.

Vienna: Dot-bracket notation from Vienna package programs, where each structure consists of matched brackets for basepairs and periods for unbased pairs. Valid brackets are (, , [, , <, A, B, C, D matched with ), , ], >, a, b, c, d, respectively. An energy value can be appended to the end of any dot-bracket structure. The function will accept slight variations of the format, including those with FASTA-like headers (in which case line breaks are allows), and those without FASTA-like headers (in which case line breaks are NOT allowed), with both types allowing for a preceding (NOT following) nucleotide sequence for the structure. Multiple entries of the same length may be in a single file, which will be returned as a single helix structure, with respectively energy values (if specified).

Connect: Output from mfold and other programs, this format is expected to be a text file beginning with a header line that starts with the sequence length, with an optional Energy/dG value, followed by a six-column tab-delimited table where columns 1 and 5 denote the position that are basepaired (unpaired when column 5 is 0). Other columns are ignored, but for completeness, column 2 is the nucleotide, column 3 and 4 are the positions of the bases left and right of the base specified in column 1 respectively (with 0 denoting non-existance), and column 6 a copy of column 1. Multiple entries of the same length may be in a single file, which will be returned as a single helix structure. All helices will be assigned the energy value extracted from their respective structure header lines.

Bpseq: Format used by the Gutell Lab’s Comparative RNA Website. The file may optionally begin several header lines (e.g. Filename, Organism, Accession, etc.), followed by a 3-column tab-delimited table for the structure, where column 1 is the base position, base 2 is the nucleotide base, and column 3 is the paired position (0 if unpaired). Certain pieces of header information will be parsed and returned as attributes of the output data frame. Multiple structures can be within a single file, returned as a single helix data frame, with attributes set to those of the first entry.

Value

Returns a helix format data frame.

Author(s)

Daniel Lai, Jeff Proctor

Examples

```r
file <- system.file("extdata", "helix.txt", package = "R4RNA")
helix <- readHelix(file)
head(helix)

file <- system.file("extdata", "connect.txt", package = "R4RNA")
connect <- readConnect(file)
head(connect)
```
Structure Mismatch Score

Scores how a basepair structure fits a sequence

Description

Calculates a score that indicates how badly a set of basepairs (i.e. a secondary structure) fits with a sequence. A perfect fit is a structure where all basepairs form valid basepairs (A:U, G:C, G:U, and equivalents) and has a score of 0. Each basepair that forms a non-canonical pairing or pairs to gaps increases the score by 1, and each base-pair with a single-sided gap increases the score by 2.

Usage

structureMismatchScore(msa, helix, one.gap.penalty = 2, two.gap.penalty = 2, invalid.penalty = 1)

Arguments

- msa: An array of strings representing sequences of interest, typically the output from `readFasta`
- helix: A helix data.frame
- one.gap.penalty: Penalty score for basepairs with one of the bases being a gap
- two.gap.penalty: Penalty score for basepairs with both bases being a gaps
- invalid.penalty: Penalty score for non-canonical basepairs

Value

Returns an array of mismatch scores.

Author(s)

Jeff Proctor, Daniel Lai
Examples

```r
data(helix)
mismatch <- structureMismatchScore(fasta, known)

# Sort by increasing mismatch
sorted_fasta <- fasta[order(mismatch)]
```

Write FASTA

`writeFasta(msa, file = stdout(), wrap = NA)`

Description

Writes out a FASTA format file from a list of named characters, where the sequences are from the elements, and the descriptions are from the names. Does not attempt to break the sequence into multiple lines.

Usage

```r
writeFasta(msa, file = stdout(), wrap = NA)
```

Arguments

- `msa` A list of characters representing each sequence, and names for each element containing the description of each sequence. Defaults to the console.
- `file` A character string pointing to the path of a file, or a connection.
- `wrap` An integer to determine the number of characters in each row before the line wraps/breaks. If NA, then no wrapping will occur.

Value

No value returned. Will write to STDOUT or file if specified.

Author(s)

Daniel Lai

Examples

```r
fasta <- c(sequence = "AAAAACCCCCUUUUU", structure = "((((....))))")
writeFasta(fasta)
```
### Description

Write out a helix data frame into a text file into the four-column tab-delimited format with proper header and column names.

### Usage

```r
writeHelix(helix, file = stdout())
```

### Arguments

- **helix**: A helix data frame.
- **file**: A character string pointing to a file path, or a file connection. Defaults to the console.

### Value

No value returned, will write to STDOUT or specific file location.

### Author(s)

Daniel Lai

### Examples

```r
# Create helix data frame
helix <- data.frame(2, 8, 3, 0.5)
helix[, 2] <- c(5, 15, 4, -0.5)
helix <- as.helix(helix)
writeHelix(helix)
```
Index

+ Topic IO
  Read FASTA, 20
  Read Structure File, 20
  Write FASTA, 23
  Write Helix, 24
+ Topic aplot
  Basepair Frequency, 4
  Covariation Plots, 10
  Create Blank Plot, 13
  Find Unknotted Groups, 15
  Plot Helix Structures, 18
+ Topic color
  Colour Helices, 7
  Log10 Space Operations, 17
+ Topic datasets
  Example Data, 14
+ Topic file
  Read FASTA, 20
  Read Structure File, 20
  Write FASTA, 23
  Write Helix, 24
+ Topic logic
  Helix Type Filters, 16
+ Topic manip
  Basepair/Helix Conversion, 5
  Coerce to Helix, 6
  Convert Helix Formats, 9
+ Topic math
  Alignment Statistics, 3
  Structure Mismatch Score, 22
+ Topic package
  R4RNA-package, 2
  baseConservation (Alignment Statistics), 3
  Basepair Frequency, 4
  Basepair/Helix Conversion, 5
  basepairCanonical (Alignment Statistics), 3
  basepairConservation (Alignment Statistics), 3
  basepairCovariation (Alignment Statistics), 3
  basepairFrequency, 8
  basepairFrequency (Basepair Frequency), 4
  blankPlot, 11, 12
  blankPlot (Create Blank Plot), 13
  Coerce to Helix, 6
  collapseHelix (Basepair/Helix Conversion), 5
  Colour Helices, 7
  colourByBasepairFrequency, 5
  colourByBasepairFrequency (Colour Helices), 7
  colourByCanonical, 12
  colourByCanonical (Colour Helices), 7
  colourByConservation, 12
  colourByConservation (Colour Helices), 7
  colourByCount, 19
  colourByCount (Colour Helices), 7
  colourByCovariation, 12
  colourByCovariation (Colour Helices), 7
  colourByUnknottedGroups, 15
  colourByUnknottedGroups (Colour Helices), 7
  colourByValue (Colour Helices), 7
  Convert Helix Formats, 9
  Covariation Plots, 10
  Create Blank Plot, 13
cut, 8
defaultPalette (Colour Helices), 7
  Example Data, 14
  expandHelix, 15
expandHelix (Basepair/Helix Conversion), 5
Fasta (Example Data), 14
Find Unknotted Groups, 15
helix (Example Data), 14
Helix Type Filters, 16
helixCanonical (Alignment Statistics), 3
helixConservation (Alignment Statistics), 3
helixCovariation (Alignment Statistics), 3
helixToBpseq (Convert Helix Formats), 9
helixToConnect (Convert Helix Formats), 9
helixToVienna (Convert Helix Formats), 9
is.helix, 6
is.helix (Coerce to Helix), 6
isConflictingHelix (Helix Type Filters), 16
isDuplicatingHelix (Helix Type Filters), 16
isOverlappingHelix (Helix Type Filters), 16
known (Example Data), 14
legend, 8
Log1/zero.noslash Space Operations, 17
logceiling (Log1/zero.noslash Space Operations), 17
logfloor (Log1/zero.noslash Space Operations), 17
logseq, 8
logseq (Log1/zero.noslash Space Operations), 17
maxHeight (Create Blank Plot), 13
names, 11
par, 14
parseBracket, 6
parseBracket (Coerce to Helix), 6
plot, 14
Plot Helix Structures, 18
plotArc (Plot Helix Structures), 18
plotArcs (Plot Helix Structures), 18
plotCovariance (Covariation Plots), 10
plotDoubleCovariance (Covariation Plots), 10
plotDoubleHelix, 11, 12
plotDoubleHelix (Plot Helix Structures), 18
plotHelix, 8, 11, 12, 14
plotOverlapCovariance (Covariation Plots), 10
plotOverlapHelix, 11, 12
plotOverlapHelix (Plot Helix Structures), 18
R4RNA (R4RNA-package), 2
R4RNA-package, 2
Read FASTA, 20
Read Structure File, 20
readBpseq (Read Structure File), 20
readConnect (Read Structure File), 20
readFasta, 3, 8, 11, 22
readFasta (Read FASTA), 20
readHelix (Read Structure File), 20
readVienna (Read Structure File), 20
Structure Mismatch Score, 22
structureMismatchScore (Structure Mismatch Score), 22
unknottedGroups, 8
unknottedGroups (Find Unknotted Groups), 15
viennaToHelix, 21
viennaToHelix (Convert Helix Formats), 9
Write FASTA, 23
Write Helix, 24
writeFasta (Write FASTA), 23
writeHelix (Write Helix), 24
A.1.2 R4RNA vignette
R4RNA: A R package for RNA visualization and analysis

Daniel Lai
March 21, 2012

Contents

1 R4RNA
  1.1 Reading Input ....................................................... 1
  1.2 Basic Arc Diagram .................................................. 2
  1.3 Multiple Structures ................................................ 2
  1.4 Filtering Helices .................................................... 3
  1.5 Colouring Structures ............................................... 3
  1.6 Overlapping Multiple Structures ................................... 4
  1.7 Visualizing Multiple Sequence Alignments .......................... 5
  1.8 Multiple Sequence Alignements with Annotated Arcs ............... 6
  1.9 Additional Colouring Methods ..................................... 6
    1.9.1 Colour By Covariation (with alignment as blocks) ............ 6
    1.9.2 Colour By Conservation (with custom alignment colours) ....... 7
    1.9.3 Colour By Percentage Canonical Basepairs (with custom arc colours) .......... 7
    1.9.4 Colour Pseudoknots (with CLUSTALX-style alignment) ............ 8

2 Session Information .................................................... 8

1 R4RNA

The R4RNA package aims to be a general framework for the analysis of RNA secondary structure and comparative analysis in R, the language so chosen due to its native support for publication-quality graphics, and portability across all major operating systems, and interactive power with large datasets.

To demonstrate the ease of creating complex arc diagrams, a short example is as follows.

1.1 Reading Input

Currently, supported input formats include dot-bracket, connect, bpseq, and a custom “helix” format. Below, we read in a structure predicted by TRANSAT, the known structure obtained form the RFAM database.

> library(R4RNA)
> message("TRANSAT prediction in helix format")
> transat_file <- system.file("extdata", "helix.txt", package = "R4RNA")
> transat <- readHelix(transat_file)
> message("RFAM structure in dot bracket format")
> known_file <- system.file("extdata", "vienna.txt", package = "R4RNA")
> known <- readVienna(known_file)
> message("Work with basepairs instead of helices for more flexibility")
> message("Breaks all helices into helices of length 1")
1.2 Basic Arc Diagram

The standard arc diagram, where the nucleotide sequence is the horizontal line running left to right from 5’ to 3’ at the bottom of the diagram. Any two bases that base-pair in a secondary structure are connect with an arc.

```
> plotHelix(known, line = TRUE, arrow = TRUE)
> mtext("Known Structure", side = 3, line = -2, adj = 0)
```

![Known Structure Diagram]

1.3 Multiple Structures

Two structures for the same sequence can be visualized simultaneously, allowing one to compare and contrast the two structures.

```
> plotDoubleHelix(transat, known, line = TRUE, arrow = TRUE)
> mtext("TRANSAT Predicted Structure", side = 3, line = -5, adj = 0)
> mtext("Known Structure", side = 1, line = -2, adj = 0)
```
1.4 Filtering Helices

Base-pairs can be associated with a value, such as energy stability or statistical probability, and we can easily filter out basepairs according to such rules.

```r
> message("Filter out helices above a certain p-value")
> transat <- transat[which(transat$value <= 0.001),]
```

1.5 Colouring Structures

We can also assign colour to the structure according to base-pairs values.

```r
> message("Assign colour to basepairs according to p-value")
> transat$col <- col <- colourByValue(transat, log = TRUE)
> message("Coloured encoded in 'col' column of transat structure")
> plotDoubleHelix(transat, known, line = TRUE, arrow = TRUE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col,
+ "fill"), inset = 0.05, bty = "n", border = NA, cex = 0.75,
+ title = "TRANSAT P-values")
```
1.6 Overlapping Multiple Structures

A neat way of visualizing the concordance between two structure is an overlapping structure diagram, which we can use to overlap the predicted TRANSAT structure and the known RFAM structure. Predicted basepairs that exist in the known structure are drawn above the line, and those predicted that are not known to exist are drawn below. Those known but unpredicted are shown in black above the line.

> plotOverlapHelix(transat, known, line = TRUE, arrow = TRUE, scale = FALSE)
1.7 Visualizing Multiple Sequence Alignments

In addition to visualizing the structure alone, we can also visualize a secondary structure along with aligned nucleotide sequences. In the following, we will read in a multiple sequence alignment obtained from RFAM, and visualize the known structure on top of it.

We can also annotate the alignment colours according to their agreement with the known structure. If a sequence can form as basepair as dictated by the structure, the basepair is coloured green, else red. For green basepairs, if a mutation has occurred, but basepairing potential is retained, it is coloured in blue (dark for mutations in both bases, light for single-sided mutation). Unpaired bases are in black and gaps are in grey.

```r
> message("Multiple sequence alignment of interest")
> fasta_file <- system.file("extdata", "fasta.txt", package = "R4RNA")
> fasta <- readFasta(fasta_file)
> message("Plot covariance in alignment")
> plotCovariance(fasta, known, cex = 0.5)
```
1.8 Multiple Sequence Alignements with Annotated Arcs

Arcs can be coloured as usual. It should be noted that structures with conflicting basepairs (arcs sharing a base) cannot be visualized properly on a multiple sequence alignment, and are typically filtered out (e.g. drawn in grey here).

\[ \text{plotCovariance(fasta, transat, cex = 0.5, conflict.filter = "grey")} \]

1.9 Additional Colouring Methods

Various other methods of colour arcs exist, along with many options to control appearances:

1.9.1 Colour By Covariation (with alignment as blocks)

\[ \text{col <- colourByCovariation(known, fasta, get = TRUE)} \]
\[ \text{plotCovariance(fasta, col, grid = TRUE, legend = FALSE)} \]
\[ \text{legend("topright", legend = attr(col, "legend"), fill = attr(col,} \]
1.9.2 Colour By Conservation (with custom alignment colours)

```r
> custom_colours <- c("green", "blue", "cyan", "red", "black", "grey")
> plotCovariance(fasta, col <- colourByConservation(known, fasta, get = TRUE), palette = custom_colours, cex = 0.5)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"), inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "Conservation")
```

1.9.3 Colour By Percentage Canonical Basepairs (with custom arc colours)

```r
> col <- colourByCanonical(known, fasta, custom_colours, get = TRUE)
> plotCovariance(fasta, col, base.colour = TRUE, cex = 0.5)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"), inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "% Canonical")
```
1.9.4 Colour Pseudoknots (with CLUSTALX-style alignment)

```r
> col <- colourByUnknottedGroups(known, c("red", "blue"), get = TRUE)
> plotCovariance(fasta, col, base.colour = TRUE, legend = FALSE,
+    species = 23, grid = TRUE, text = TRUE, text.cex = 0.2, cex = 0.5)
```

2 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 2.13.2 (2011-09-30), x86_64-unknown-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C,
  LC_MONETARY=C, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C,
  LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: R4RNA 0.1.4
- Loaded via a namespace (and not attached): tools 2.13.2