Drawing and Editing the Secondary Structure(s) of RNA.

Yann Ponty, Fabrice Leclerc

Drawing and editing the secondary structure(s) of RNA

Yann Ponty[1*] and Fabrice Leclerc[2]

[1] CNRS & INRIA AMIB
Laboratoire d'Informatique de l'X (LIX) UMR 7161
Ecole Polytechnique, Palaiseau, France

[2] CNRS
Institut de Génétique et Microbiologie (IGM) UMR 8621
Université Paris Sud, Orsay, France

[*] Correspondence should be addressed to Yann Ponty:
e-mail: yann.ponty@lix.polytechnique.fr

Abstract Secondary structure diagrams are essential, in RNA biology, to communicate functional hypotheses and summarize structural data, and communicate them visually as drafts or finalized publication-ready figures. While many tools are currently available to automate the production of such diagrams, their capacities are usually partial, making it hard for a user to decide which to use in a given context.

In this chapter, we guide the reader through the steps involved in the production of expressive publication-quality illustrations featuring the RNA secondary structure. We present major existing representations and layouts, and give precise instructions to produce them using available free software, including **jViz.RNA**, the **PseudoViewer**, **RILogo**, **R-chie**, **RNAplot**, **R2R** and **VARNA**. We describe the file formats and structural descriptions accepted by popular RNA visualization tools. We also provide command-lines and Python scripts to ease the user's access to advanced features. Finally, we discuss and illustrate alternative approaches to visualize the secondary structure in the presence of probing data, pseudoknots, RNA-RNA interactions and comparative data.

Key words: RNA Visualization, Secondary Structure, Graph Drawing, Pseudoknots, Non-Canonical Motifs, Structure-Informed Multiple Sequence Alignments.

1 Introduction

1.1 Context and Definitions

The secondary structure of RNA represents a discrete abstraction, or projection, of a three-dimensional structure restricted to (a subset of) its interacting positions. As such, it is naturally used within schematics depicting the general architecture of a given RNA conformation. It provides a context for various sequential information, including evolutionary conservation, accessibility to chemical/enzymatic probes, or predicted single-strandedness. In structural studies, it is also used as a scaffold to perform an interactive three-dimensional modeling by homology. Such schematics are finally used to illustrate functional scenario involving RNA, possibly interacting with another molecule or ligand, raising the need for the production of publication-quality diagrams.

Formally, the **secondary structure** of an RNA is simply a set of base-pairs connecting some of the positions in an RNA sequence. Any base-pair will be denoted by a pair (i, j), indicating that the nucleotide at position i in the sequence is paired with the nucleotide at position j. In its strictest definition, this set of base-pairs obeys additional constraints:

- **Only canonical interactions:** Any base-pair must be canonical (A-U, G-C, or G-U bases, interacting on Watson-Crick/Watson Crick edges in *cis* orientation);
- **Non-crossing base-pairs:** Any pair of base-pairs (i, j) and (i', j') should be free of crossing interactions ($i < i' < j < j'$ or $i' < i < j' < j$). This property forbids the representation of complex structural features like pseudoknots;
- **Single partner:** Any position may interact with at most a single distinct partner.

These restrictions lead to **tree-like objects** which can be drawn on the 2D plane, and are easy to analyze visually. They also ensure the computational tractability of the underlying algorithmic processing, while retaining the capacity to suggest the general architecture (helices, junctions. . .) adopted within a given conformations.

However, enforcing these restrictions leads to a loss of information which could deteriorate the quality of subsequent analyses. In such cases, a more detailed view of the base-base interaction network must be adopted. Some or all of the above constraints can be lifted, leading to the **extended secondary structure**, a representation which typically supports additional annotations, such as stacked bases, or interacting edges and orientation for base-pairs. Crossing helices, usually referred to as **pseudoknots**, using a topological analogy, are also supported by some tools, but their planar layout is usually challenging, and even intrinsically unfeasible for some RNA architectures.

1.2 Objectives of RNA visualization

The secondary structure can be naturally represented as a graph whose vertices are individual nucleotides. In this representation, edges are expected to connect consecutive nucleotides in the sugar-phosphate backbone, or pairs of positions involved in a base-pair mediated by hydrogen bonds. Besides these formal requirements, additional properties have been identified as desirable by earlier work (1):

Modularity Inspection of the drawing should suggest a decomposition of the structure into functional domains. For instance, helices should be easy to identify.

Robustness Structural similarity should be revealed by similar-looking representations. In particular, the introduction of few evolutionary events (insertions/deletions of single or paired nucleotides) should not lead to significant changes.

Clarity The resulting drawing should lend itself to an easy visual analysis. In particular, overlapping nucleotides and crossing lines should be avoided, and a constant distance should separate backbone-consecutive nucleotides and base-pairs respectively.

Table 1: Main features of selected tools offering a visualization of the RNA secondary structure.

Name	Platform(s) [**BWML**[a]]	Ease of use	Layouts	Interactivity	Pseudoknots	Ext. sec. str.[b]	Output[c]	Reference
jViz.RNA	●●●●	++	Circular Linear Graph	+	+	●	**EPS** PNG	(2) (3)
PseudoViewer	●●○○	++	Graph		++	○	**EPS SVG** PNG GIF	(4)
RNAMovies	●●●●	++	Graph		+	○	**SVG** PNG JPEG GIF	(5)
RILogo	●●●●	+	Linear			○	**SVG**	(6)
R-chie	●●●●	++	Linear		+	○	**PDF** PNG	(7)
RNAplot	●[d] ●●●	+	Graph			○	**PS SVG** GML	(8)
RNAView	○●●●	+	Graph		+	●	**PS**	(9)
RNAViz	○●●●	+	Graph	++	+	○	**PDF**[e]	(10)
R2R	○●●●	-	Graph		+	○	**PDF SVG**	(11)
S2S/Assemble	○●●●	+	Linear Graph	++	+	●	**SVG**	(12) (13)
VARNA	●●●●	++	Circular Linear Graph	++	+	●	**EPS SVG** PNG JPEG GIF	(14)
xRNA	○●●●	+	Graph	++	+	○	**EPS**	–

[a] Indicates support (●) or lack of support (○) for **B**rowser-based, **W**indows, **M**ac and **L**inux executions. [b] Indicates support (●) or lack of support (○) for an extended secondary structure. [c] Bold output formats indicate vector graphics, allowing for convenient post-processing. [d] Only available from the Vienna RNA webserver, available at http://rna.tbi.univie.ac.at/. [e] Export accessible through a print option, using a custom printer driver such as Adobe Distiller.

Realism Drawings should ideally constitute a projection of the three-dimensional structure. In this respect, relative distances between consecutive nucleotides and base-pairing partners should be proportional to that observed in three-dimensional structures.

Aesthetics Arguably the most subjective aspect of RNA visualization, encompasses a large variety of criteria ranging from standard color schemes to a strict adherence to historical representations for classic families of RNAs (e.g. cloverleaf shape for transfer RNAs).

Unfortunately, satisfying combinations of these properties may be computationally intractable, or simply even impossible for intrinsic reasons. Consequently, many alternative representations and layout strategies have been proposed over the years, each arising as a tradeoff.

1.3 Existing tools

Many tools are now available for visualizing the secondary structure of RNA. These tools differ on many aspects, depending on their intended application and functionalities. Among the distinguishing features of available tools, one denotes the presence of a graphical user interface, allowing for a convenient post-processing

before resorting to a time-consuming general-purpose editor. Moreover, while certain formats may be converted into others in a lossless manner, most conversion may lose some format-specific data, and a support for a rich variety of input format is therefore desirable. The output format of a software is also of great importance, as generated figures typically require further post-processing before meeting the quality criteria expected by publishers. Therefore, vector formats such as Portable Document Format (PDF), or Scalable Vector Graphics (SVG) should always be preferred to bitmap formats. Table 1 summarizes the foremost tools available for visualizing the secondary structure. Which, of the available alternatives, should be preferred will typically depend on the intended application. We hope that this overview of existing tools and representations will assist the reader in his choice of the right visualization, and ultimately ease the production of more informative pictures.

1.4 Outline

In this chapter, we focus on the necessary steps involved in the production of publication-quality illustrations involving the RNA secondary structure. After introducing the main data sources and file formats, we describe a preferred workflow, and stress on the advantages of vector graphics manipulation. Then, we describe major proposed representations and layouts for the secondary structure. We also provide simple command lines to invoke main tools. We also describe how to address the specific visualization needs arising from exemplary use-cases: chemical probing experiments, RNA pseudoknots and interactions, and multiple-sequence alignments.

2 Materials

Reflecting the diversity of functions and cellular processes involving non-coding RNAs, secondary structure data is organized within a maze of domain-specific databases, and encoded in a variety of file formats. We briefly mention these sources, and describe in further details the syntax of major file formats.

2.1 Data sources

At the primary structure level, including sequences and alignments, the RFAM database (15) is the authoritative source of RNA data. Unfortunately this centralized authority is without a counterpart on the secondary structure level, leading to a wealth of specialized repositories focusing on RNAs of specific functions. The reason of this situation is mostly due to the young history of RNA computational biology, coupled with the fact that, up until recently, RNA structural data was relatively scarce compared to proteins. Consequently, databases have typically arisen from a variety of domain-specific efforts, which have not currently undergone standardization.

Of notable interest at the secondary structure level is the STRAND database (16), a collection of secondary structures found in, or inferred from, a variety of data bases. However, the purpose of this database, which was initially assembled to train new energy models from existing sequence/structure data and benchmark computational prediction tools, conflicts with the goal of completeness, and secondary structure data must currently be sought within a variety of specialized databases. One should however note that recent initiatives, such as the RNA ontology consortium (17) or the RNA Central project (18), have led to propositions for more rational organisations of RNA structural data, and a solution to these issue may be found in the near future.

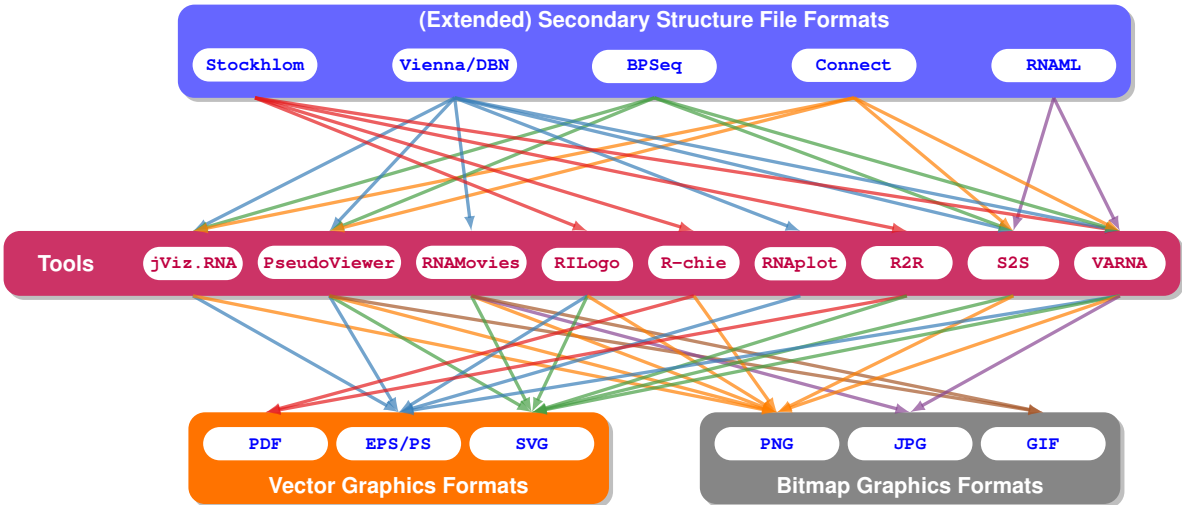


Fig. 1: Main input and output formats supported by the major visualization tools for the RNA Secondary Structure.

2.2 RNA secondary structure file formats

Reflecting the diversity of biological contexts and computational use-cases involving RNA secondary structure, many formats were proposed over time to support its description. Aside from the ubiquitous FASTA format, extended by the popular Vienna RNA software package (19) to include a dot-bracket encoding of the secondary structure, the STOCKHOLM format is used within RFAM to present a WUSS-formatted consensus secondary structure, possibly including pseudoknots. The BPSEQ and CONNECT formats represent an adjacency list, indicating the partner (if any) of each position.

Notably, the RNAML format (20) represents a unifying effort of the community to represent virtually any sort of RNA-related data. While this format is not widely used to represent the secondary structure because of its intrinsic verbosity, it is clearly the format of choice to represent and manipulate the extended secondary structure, including non-canonical base-pairs.

2.2.1 STOCKHOLM format

STOCKHOLM format files are arguably the preferred representation for RNA multiple sequence alignments. As illustrated by Figure 2, a STOCKHOLM file breaks the global alignment of a set of sequences into portions of bounded width. Unique identifying prefixes (Organism name/Accession ID) are associated with each portion. Additionally a set of mark-up lines, recognizable by their prefix `#=`, specify various additional information, including accession identifiers, experimental parameters, bibliographical references... Two mark-up line types are especially relevant to RNA bioinformatics, the `#=GC RF` lines, which indicate a sequence consensus, and the `#=GC SS_cons`, which indicates the secondary structure consensus. The latter uses a parenthesis scheme to represent base-pairing positions using corresponding parentheses. Pseudoknots can also be represented using an additional alphabetical system (`A` matching with `a`, `B` with `b`...).

2.2.2 Vienna RNA dot-parentheses (aka dot-bracket/parenthesis) and Pseudobase notations

In this format, the RNA sequence is coupled with a dot-parenthesis string where matching pairs of opening and closing parentheses identify base-pairs. This expression is well-parenthesized, meaning that any opening

```
# STOCKHOLM 1.0
#=GF ID    mir-22
#=GF AC    RF00653
...
O.latipes.1             CGUUG.CCUCACAGUCGUUCUUCA.CUGGCU.AGCUUUAUGUCCCACG..
Gasterosteus_aculeat.1  GGCUG.ACCUACAGCAGUUCUUCA.CUGGCA.AGCUUUAUGUCCUCAUCU
R.esox.1                AGCUGAGCACA...CAGUUCUUCA.CUGGCA.GCCUUAAGGUUUCUGUAG
...
#=GC SS_cons            .<<<<.<<..<<<<<<<<<<<<<..<<<<..<<<<<<<.&<<........
#=GC RF                 gGccg.acucaCagcaGuuCuuCa.cuGGCA.aGCuuuAuguccuuauaa

O.latipes.1             CCCCACCGUAAAGCU.GC.CAGUUGAAGAGCUGUUGUG..UGUAACC
Gasterosteus_aculeat.1  ACCAGC..UAAAGCU.GC.CAGCUGAAGAACUGUUGUG..GUCGGCA
R.esox.1                ACAGGC..UAAACCU.GC.CAGCUGAAGAACUGCUCUG..GCCAGCU
...
#=GC SS_cons            ....>>..>>>>>>>.>>.>>..>>>>>>>>>>>>>>...>>>>>>.
#=GC RF                 acaaac..UaaaGCu.GC.CaGuuGaaGaaCugcuGug..gucggCu
//
```

Fig. 2: Stockholm formatted file fragment, excerpted from RFAM seed alignment for miRNA mir-22. RFAM ID: RF00653.

```
> Rat Alanine tRNA
GAGGAUUUAGCUUAAUUAAAGCAGUUGAUUUGCAUUUAACAGAUGUAAGAUAUAGUCUUACAGUCCUUA
((((((...((((.....)))).(((((.......)))))...(((((((((...)))))))))))))).
```

Fig. 3: Dot-parentheses (aka dot-bracket) notation for the minimal free-energy structure of the Rat Alanine tRNA, predicted using **RNAFold**.

```
       1590      1600      1610      1620      1630
#      |123456789|123456789|123456789|123456789|123456
$ 1590 AAAAAACUAAUAGAGGGGGGACUUAGCGCCCCCCAAACCGUAACCCC=1636
% 1590 ::::::::::::::[[[[[[:::::(((]]]]]]::::)))::::::
```

Fig. 4: Pseudobase notation for the Gag/pro ribosomal frameshift site of Bovine Leukemia Virus Source: Pseudobase, entry# PKB1.

parenthesis can be unambiguously associated with a closing parenthesis, inducing a set of non-crossing base-pairs. For instance, in Figure 3, the bases at first and ante-penultimate positions are base-paired.

Since matching pairs of parenthesis cannot unambiguously identify crossing base pairs, pseudoknots cannot be represented strictly within this format. Therefore this format was extended by the PseudoBase to include support for multiple parentheses systems, allowing crossing interactions to be represented. In Figure 4, two crossing helices, forming a H-type pseudoknot, are initiated by base-pairs at positions $(1604, 1623)$ and $(1615, 1630)$ respectively.

2.2.3 BPSEQ format

The BPSeq format is an alternative to the CT format introduced by the Comparative RNA Web site (CRW, hosted at the University of Texas Austin by the Robin Gutell Lab). It essentially consists in a simplified version

```
Filename: AM286415_b.bpseq
Organism: Yersinia enterocolitica subsp. enterocolitica 8081
Accession Numbers: AM286415
Citation and related information available at http://www.rna.ccbb.utexas.edu
1 U 0
...
117 U 0
118 U 236
119 G 235
120 C 234
121 C 233
122 U 232
123 G 231
124 G 230
...
230 C 124
231 C 123
232 A 122
233 G 121
234 G 120
235 C 119
236 A 118
...
```

Fig. 5: Fragment of a BPSEQ formatted 5s rRNA inferred using comparative modelling. Source: Comparative RNA web site.

of the CT format. As illustrated by Figure 5 any BPSeq file starts with four self-explanatory lines identifying the data source and content, and is followed by the structure, specified as a sequence of space-separated triplets (x, y, z), where:

- x Position;
- y IUPAC code for base;
- z Base-pairing partner position (0 if unpaired).

2.2.4 CONNECT (CT) format

This format was introduced by **Mfold** (21), the historical tool for the ab-initio prediction of RNA secondary structure, and is still used to date by several prediction tools. After an initial header consisting of the sequence length followed by a comment (see Figure 6), each position is represented by six fields (a, b, c, d, e, f), each encoded in a fixed-width (8 characters) column:

- a Position;
- b IUPAC code for base;
- c Position of previous base in the backbone (5'-3' order, 0 is used if first position);
- d Position of next base in the backbone (5'-3' order, 0 is used if last position);
- e Base-pairing partner position (0 if unpaired);
- f Position (duplicated). This format can be gently abused to allow for the description of pseudoknots, although such motif cannot be predicted by Mfold.

```
80      dG = -33.48 [Initially -35.60]
1       U       0       2       80      1
2       G       1       3       79      2
3       G       2       4       78      3
4       G       3       5       77      4
5       A       4       6       76      5
6       U       5       7       0       6
7       G       6       8       75      7
...
75      U       74      76      7       75
76      U       75      77      5       76
77      C       76      78      4       77
78      C       77      79      3       78
79      U       78      80      2       79
80      A       79      0       1       80
```

Fig. 6: CONNECT format for the **Mfold** 3.7 (21) predicted structure of the human let-7 pre-miRNA.

```
HEADER    RNA                                     27-JUL-09   3IGI
TITLE     TERTIARY ARCHITECTURE OF THE OCEANOBACILLUS IHEYENSIS GROUP
TITLE    2 II INTRON
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: GROUP IIC INTRON;
COMPND   3 CHAIN: A;
...
ATOM   8009  P     U A 375      19.076  79.179 370.688  1.00 66.25           P
ATOM   8010  OP1   U A 375      18.815  77.862 371.313  1.00 83.22           O
ATOM   8011  OP2   U A 375      19.869  80.203 371.409  1.00 56.32           O
...
CONECT 8654 8520
CONECT 8655 8521
CONECT 8658 8531
MASTER      717    0   66    0    0    0   69    6 8656    2  123   33
```

Fig. 7: Fragment of a three-dimensional model for the Oceanobacillus iheyensis Group II intron (PDBID: 3IGI).

2.2.5 PDB file format

The PDB format is a comprehensive text-formatted representation of macromolecules used with the authoritative eponym repository of experimentally-derived 3D models (22). Originally introduced to represent protein structure, it has been enriched over the years to include fine details of the experimental protocol used for any structure derivation. However, it does not support detailed information regarding base-pairing position, and is mostly used by RNA software as a raw geometrical descriptor of a 3D model, as illustrated by Figure 7.

2.2.6 RNAML format

RNAML (20) is an XML format, introduced to address the dual need to unify data representations related to RNA, and to represent novel important features of its structure (e. g. non-canonical base-pairs and motifs). Although still challenged in its former goal by less structured, domain-specific, formats (arguably because of the intrinsic verbosity arising from its ambitious goals), RNAML has established itself as the format of choice for an enriched

```
<?xml version="1.0"?>
<!DOCTYPE rnaml SYSTEM "rnaml.dtd">
<rnaml version="1.0">
   <molecule id="xxx">
      <sequence>
         ...
         <seq-data>
            UGUGCCCGGC AUGGGUGCAG UCUAUAGGGU...
         </seq-data>
         ...
      </sequence>
      <structure>
         <model id="yyy">
            <base> ... </base> ...
            <str-annotation>
               ...
               <base-pair>
                  <base-id-5p><base-id><position>2</position></base-id></base-id-5p>
                  <base-id-3p><base-id><position>260</position></base-id></base-id-3p>
                  <edge-5p>+</edge-5p>
                  <edge-3p>+</edge-3p>
                  <bond-orientation>c</bond-orientation>
               </base-pair>
               <base-pair comment="?">
                  <base-id-5p><base-id><position>4</position></base-id></base-id-5p>
                  <base-id-3p><base-id><position>259</position></base-id></base-id-3p>
                  <edge-5p>S</edge-5p>
                  <edge-3p>W</edge-3p>
                  <bond-orientation>c</bond-orientation>
               </base-pair>
               ...
            </str-annotation>
         </model>
      </structure>
   </molecule>
  <interactions> ... </interactions>
</rnaml>
```

Fig. 8: RNAML formatted fragment of an all-atom 3D model for the Oceanobacillus iheyensis Group II intron (PDBID: 3IGI, also shown in Figure 7), produced by the **RNAView** software. The `base-pair` XML sections are automatically annotated geometrically from the PDB file, and represent base pairing positions. Such base-pairs may be non-canonical, i.e. they may involve non-standard pairs of nucleotides, interacting edges, or orientation. Here, positions 2 and 260 form a canonical base-pair (U-A, both Watson-Crick edges, *cis* orientation), while positions 4 and 259 form a non-canonical base-pair (U-U, Sugar/Watson-Crick edges, *cis* orientation).

symbolic description of the tertiary structure, as illustrated by Figure 8. Accordingly, it is currently supported by most automated methods capturing non-canonical interactions and motifs.

3 Methods

3.1 Producing publication-quality pictures of RNA

The production of publication-quality illustrations can be greatly eased by the choice of adequate tools. Indeed, while one could in principle produce illustrative pictures of RNA by relying only on vector graphics software such as **Inkscape** or **Adobe® Illustrator®**, such a task would quickly be extremely time-consuming as soon as RNAs of moderate lengths are considered. In particular, this scenario would require spending a huge amount of time on trivial tasks, such as the layout of individual bases, or the reorientation of helices, and would not necessarily yield homogeneous output pictures.

For these reasons, current workflows attempt to keep using domain-specific tools as far as possible. As illustrated by Figure 9, a variety of file formats (see Section 2.2) can be used to represent the secondary structure, possibly resulting from an automated annotation of a 3D all-atom model. Such files are then used, possibly in

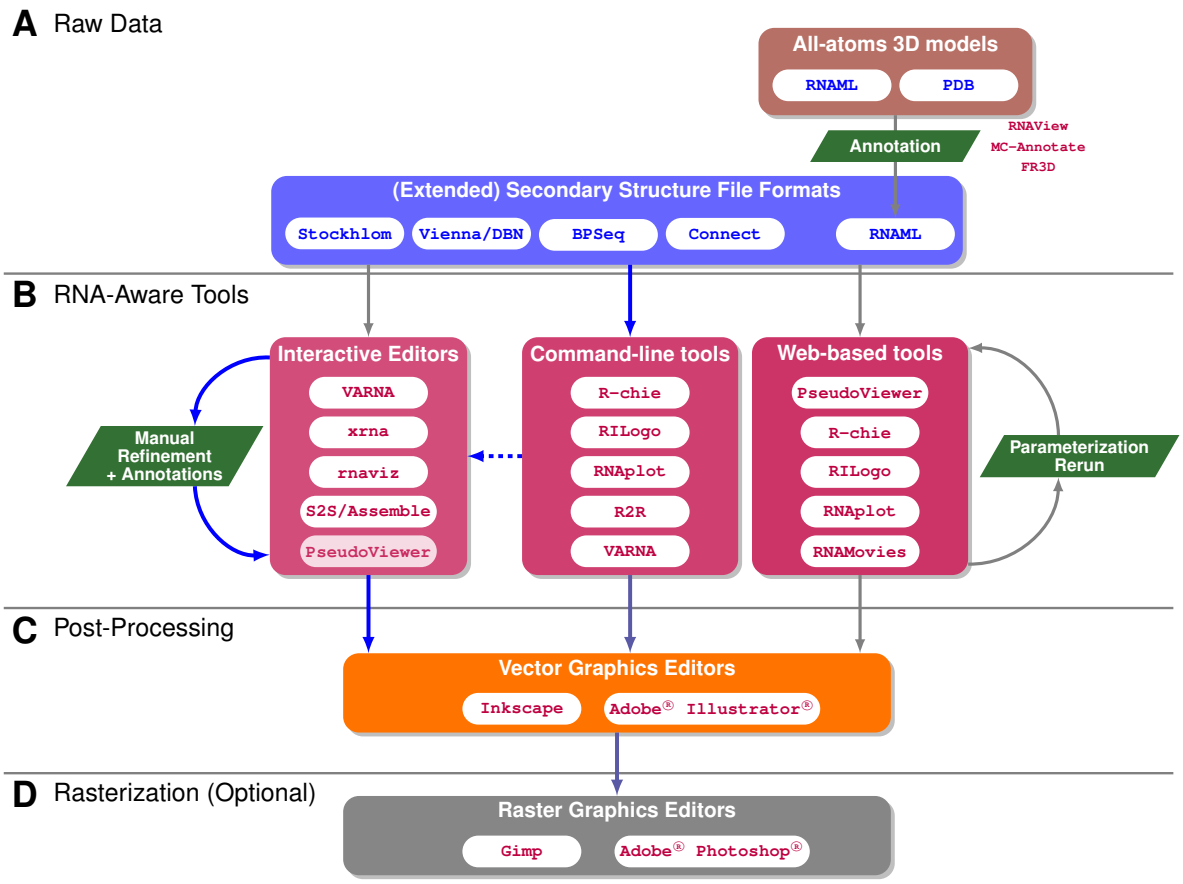


Fig. 9: Typical workflow for the production of publication-quality diagrams. The preferred execution (blue arrows, dashed arrow indicates a functionality which is not widely available) will start with a scripted production of a data-rich initial draft, followed by a interactive refinement within a specialized GUI, and conclude with a limited post-processing session using a general-purpose vector graphics editor.

combination with additional data and annotations, as input for a large number of software which use some algorithm to produce an initial layout. This layout may then be further edited within a dedicated GUI, e.g. to refine the automated layout, or to add further annotations to the drawing. Finally, the output may be exported as a picture, encoded using a general purpose picture format. Vector format pictures may then be manually edited using tools such as **Inkscape** or **Adobe® Illustrator®**, typically to add legends or merge several pictures into panels.

One should always prefer vector formats (SVG, PDF, EPS/PS) over bitmap graphics (PNG, JPG). Indeed, the latter, having lost track of the underlying geometry, only allows for very basic translations and color adjustment while the former, by retaining a symbolic description of the picture, enables a convenient post-processing including rotations and grouping/dissociation of objects, and the possibility of correcting factual errors such as erroneous nucleotides... Finally, bitmap pictures of virtually any resolution can be generated from a vector graphics, while the production of a bitmap picture amounts to being blocked in a given resolution. Such a commitment is arguably dangerous before the final zoom level of the picture, and the requirements of the publisher, are known. Working with bitmap picture may leads users to create files of extreme resolutions, which are subsequently difficult to handle, and sometimes even store. Therefore, the rasterization of RNA secondary structure pictures should be postponed as much as possible in the figure-creation process.

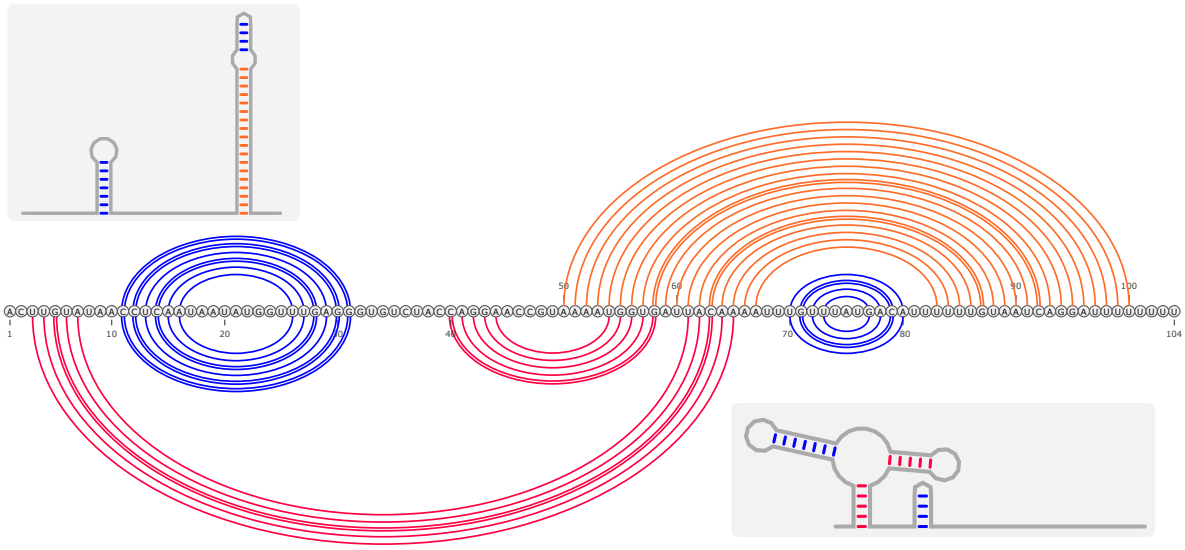


Fig. 10: ON (above) and OFF (below) states of the pbuE adenine riboswitch (23), jointly represented using a linear layout created using **VARNA** (14), followed by minimal post-processing. Shared structural elements (blue arcs) are easily identified in this representation, a property which is not satisfied by their typical graph layout (gray boxes).

3.2 Representations, layouts and basic usage

Many representations have been proposed to display the secondary structure of RNA, each with its strengths and limitations. In this section, we give an overview of the existing propositions and, for each representation and tool, we provide minimal command lines and/or **python** scripts to obtain them.

3.2.1 Linear layout

The linear layout represents base-pairs as arcs drawn over and/or under a linearly-drawn RNA sequence. It arguably constitutes one of the easiest way to represent the secondary structure, an can be thought as a genomic perspective over the secondary structure. Among the strengths of this representation, one counts an easy identification of helices as sets of nested arcs, and a convenient (unbiased) representation of pseudoknots. This representation may also be easily adapted into a side-by-side visual comparison of two or several candidate structures for a given RNA, provided an alignment is available at the sequence level. In such a joint representation, shared base-pairs are easy to identify, as illustrated by Figure 10.

However, the linear representation suffers from a poor density of information. Indeed, the height of arcs associated with base-pairs typically grows proportionally to the distance between its associated positions, leading to diagrams whose total area scales quadratically with the sequence length. Consequently, one must either accept to lose the fine details of the sequence/structure, or resort to interactive visualization strategies based on zoom/-pan navigation operations. Such strategies, however, will impede the visualization of long-range interactions, by forbidding the simultaneous visualization of – sequentially distant – interacting positions.

Such diagrams can be generated, for instance starting from a DBN file `XXX.txt`, using **VARNA** (14):

```
# Running VARNA (Linear layout/SVG output)
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd
   -i XXXX.yyy -o out.svg -algorithm line
```

or the **RChie** (7) software:

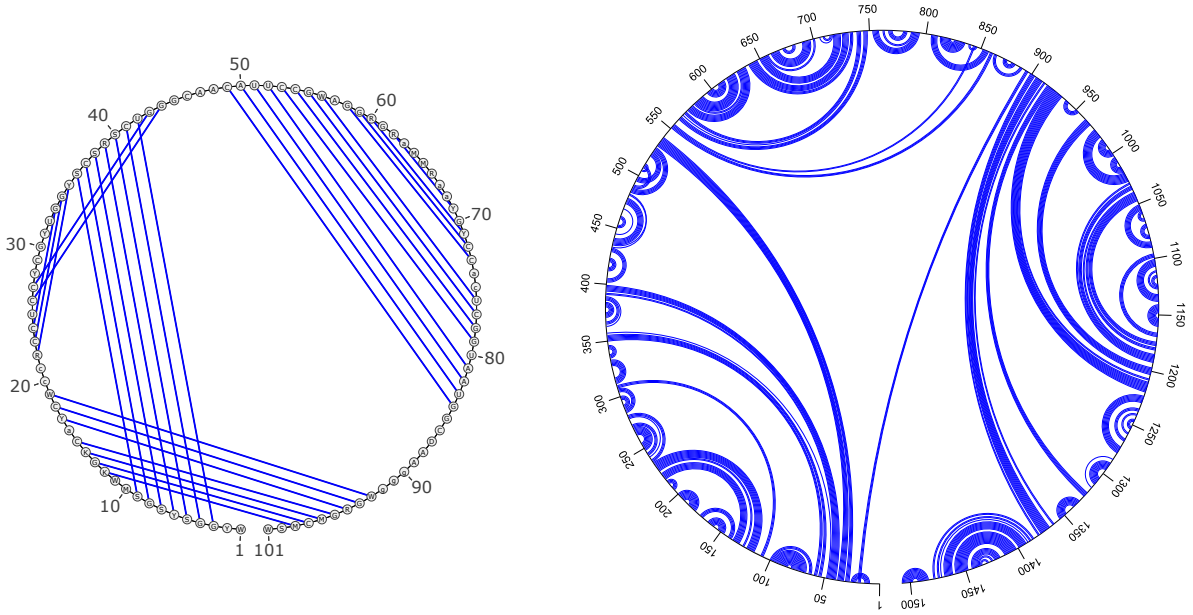


Fig. 11: Left: RFAM consensus sequence/structure for the Hepatitis delta virus ribozyme family (RFAM id: RF00094), drawn using **VARNA** (14); Right: comparative model for the 16s ribosomal RNA in Thermus thermophilus (Source: Comparative RNA Web site (24)), drawn using **jviz.RNA** (2)

```
# Running R-chie (Linear layout/PDF output)
Rscript rchie.R --format1 "vienna" --pdf --output=out.pdf XXX.yyy
```

jviz.RNA (2) may also be used to generate such a representation in an interactive way.

3.2.2 Circular Layout

The circular layout constitutes a modified version of the linear layout, in which the sequence is drawn along a circle. In this representation, base-pairs are either drawn as arcs, or as chords of the circle, as illustrated by Figure 11.

This representation is usually more compact than its linear counterpart (albeit only by a constant factor), and puts more emphasis on long distance interactions. However, one must remain cautious in a visual analysis of such a diagram, as the height of an arc is no longer strictly proportional to its distance. This phenomenon has a particularly strong impact on the representation of helices involving both ends of an RNA. In this case, helices are represented by chords/arcs which are in close proximity to the circular support (see helix supported by $(19, 93)$ in Figure 11, Left), and may erroneously be interpreted as a local structural feature.

This representation may also be of limited help to build one's intuition regarding the similarity of secondary structures. For instance, drawing two structures simultaneously using both inward and outward base-pair will not even assign similar-looking patterns to perfectly conserved helices. Furthermore, it does not easily allow for a visual realignment of homologous structures, as minor changes in the structure will lead to helices having different orientations. For these reasons, the circular layout is typically reserved for an automated broad overview of the structural organization in larger RNAs, mixing long and short range interactions.

Preferred software for producing this representation include **jViz.RNA** (2) (using the GUI + cautious post-processing, see Note 1), or **VARNA** (14), invoked as:

```
# Running VARNA (Linear layout/PostScript output)
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd
   -i XXXX.yyy -o out.eps -algorithm circular
```

3.2.3 Squiggle plots: Planar graph representations

This representation attempts to draw the secondary structure as a schematics of its three-dimensional structure. Helices are almost universally represented as straight, ladder-like, segments. Besides this norm, layout strategies largely differ, especially with respect to the layout of multiples loops (3-way junctions and more), as summarized by Figure 12. This representation is usually preferred to illustrate functional scenario.

RNAView (Figure 12.A) draws the secondary structure of an RNA (PDB model) as a direct 2D projection of the three-dimensional model, chosen to maximize the spread of the diagram. This strategy results in drawings which, despite being usually self-overlapping, are often indicative of the relative orientation of helices for small RNAs, as illustrated by Figure 12.A. The software is also mentioned in Section 2.2 (Figure 8) for its capacity to annotate the base-pairs of a 3D model. After downloading/installing the software from **http://ndbserver.rutgers.edu**, a PostScript projection can be produced from an input RNA (PDB format file `XXXX.pdb`), by simply running the command:

```
# Running RNAView on RNA 3D model (PDB format)
rnaview -p XXXX.pdb
```

Note that, in addition to the output PostScript file, this command also creates an RNAML file `XXXX.pdb.xml`, which contains the extended secondary structure for this 3D model.

VARNA (Figure 12.B) offers two types of layout strategies to render squiggle plots. The first one, named the radial strategy, aims at a consistent spacing of backbone-consecutive and paired nucleotides. To that purpose, unpaired nucleotides are positioned along an imaginary circle, whose radius is computed so that consecutive elements are always distant by a predefined distance. After downloading the latest version of **VARNA** as a JAR archive `VARNAvx-y.jar` from **http://varna.lri.fr**, the default radial strategy can be used to draw an extended secondary structure (denoted by a RNAML file, including non-canonical base-pairs, pseudoknots and multiple interactions per position, and output as a SVG file, through the **single-line** command:

```
# Running VARNA on the ext. sec. str. (RNAML -> Vector graphics)
java -cp VARNAvX-Y.jar fr.orsay.lri.varna.applications.VARNAcmd
   -i XXXX.pdb.xml -o YYYY.svg
```

The resulting layout, illustrated by Figure 12.B, does not guarantee non-overlaping diagrams. The result may therefore require further manual post-processing for larger RNAs, a task which can be performed within **VARNA**'s dedicated GUI, provided that the specific file format `varna` is chosen for the output:

```
# Running VARNA (RNAML -> VARNA session file)
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd
   -i XXXX.pdb.xml -o YYYY.varna
```

Alternatively, the NAView algorithm (25) is implemented in **VARNA**, and usually ensures a non-overlapping layout, resulting in layouts which are similar to Figure 12.C. It can be invoked by setting the `algorithm` command-line option:

```
# Running VARNA (NAView algorithm -> PostScript output)
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd
   -i XXXX.pdb.xml -o YYYY.eps -algorithm naview
```

A *skeleton* view can also be produced, as shown in both subpanels of Figure 10, by using the combination of options below (replacing `XXXX` with your input file, `YYYY` with the output file, including an extension which will decide its type, and `ZZZ` with the length of the RNA):

```
# Running VARNA (NAView algorithm -> PostScript output)
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd -i XXXX -o YYYY -flat true
     -highlightRegion "1-ZZZ:fill=#A0A0A0,outline=#A0A0A0,radius=10" -drawBases false -backbone "#A0A0A0"
      -fillBases "#A0A0A0" -bpStyle simple -numPeriod 50
```

RNAPlot (Figure 12.C), a software which is part of the Vienna RNA package, also uses the NAView algorithm for its default layout. As can be seen from Figure 12.C, a peculiarity of this layout is its unique orientation, going counter-clockwise in the 5'→3' direction. After successful installation of the package, downloaded from **http://www.tbi.univie.ac.at/RNA/**, it is invoked on a simple DBN file (see Figure 3 for an example) `xxxx.yyy`, by running the command:

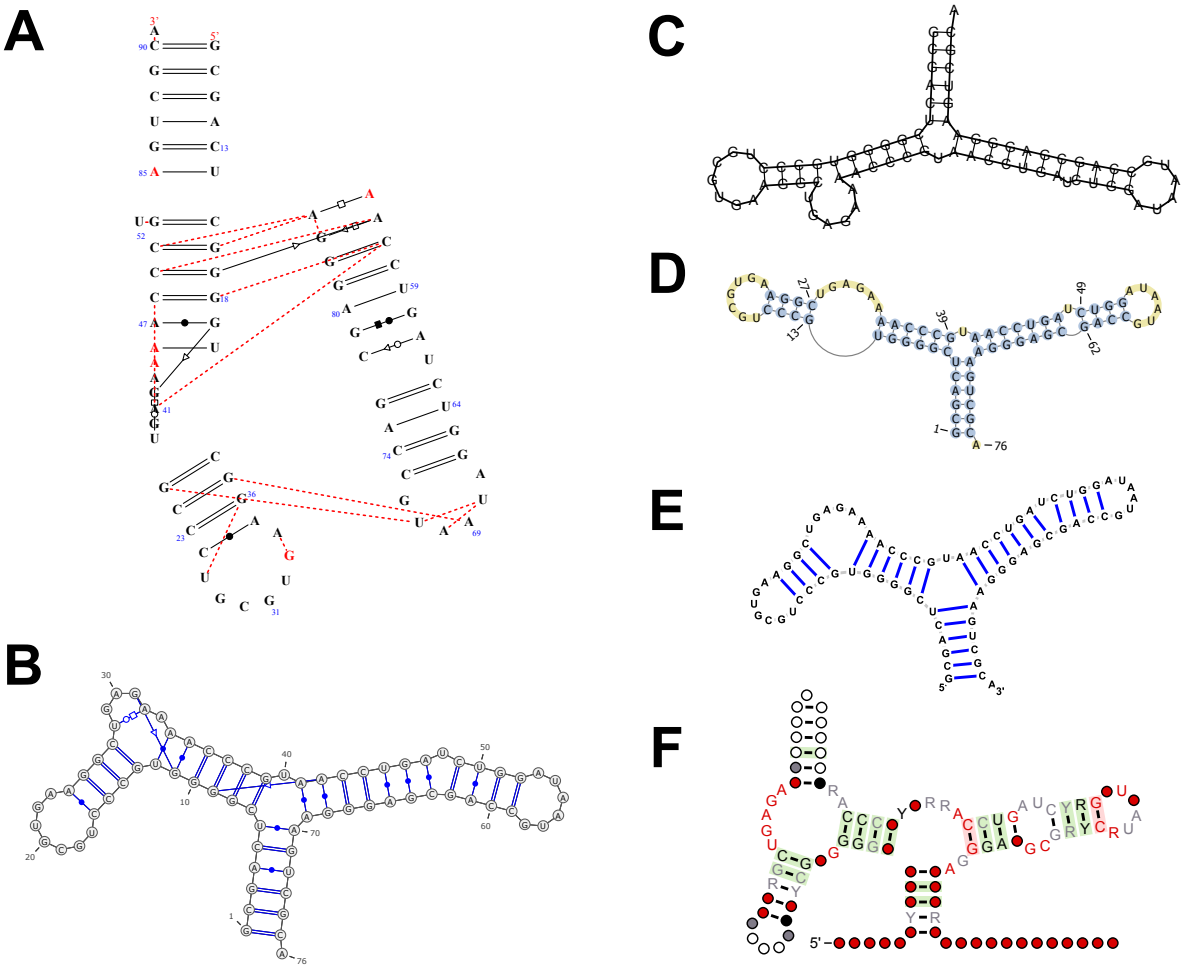


Fig. 12: Various layout strategies for the display of (extended/consensus) RNA secondary structures as (outer) planar graphs, also known as squiggle plots. Source: three-dimensional model of the TPP riboswitch (PDB id:2HOM), and associated RFAM family (RFAM: RF00059).

```
# Running RNAplot (NAView algorithm)
RNAplot < xxxx.yyy
```

The resulting `PostScript` drawing is output to a file `zzz_ss.ps`, where `zzz` is the first token found on the comment line of the input file (defaults to `rna.ps` if absent). A radial layout can also be specified through a dedicated option:

```
# Running RNAplot (Radial algorithm)
RNAplot  --layout-type=0 < xxxx.yyy
```

The **`PseudoViewer`** (Figure 12.D) uses an original strategy for choosing the relative orientations of helices. As illustrated by Figure 12.D, pairs of consecutive helices, connected by a bulge or an interior loops, are drawn along the same axis. The resulting structures appear simpler, possibly at the cost of uneven distances along the backbone. Furthermore, multiple junctions/loops are stretched and oriented to completely forbid overlaps. Finally, this advanced layout strategy satisfactorily supports a very large class of pseudoknots. The software can be used online, as a web server or a web service, at **`http://pseudoviewer.inha.ac.kr/`**. Graphical User Interfaces (GUI) are provided for the Windows platform but, to the best of our knowledge, no command line version is currently available.

jViz.RNA (Figure 12.E) implements an interactive layout strategy for squiggle plots, which relies on a spring model. Both backbone bonds and base-pairs are modeled as springs, each with a preferred length and tension. The layout is iteratively refined, simulating the springs reactions until an equilibrium is found. As illustrated by Figure 12.E, helices are not necessarily drawn in rectangular boxes, and are not even necessarily straight, leading longer helices to exhibit a *wavy* shape. However, a user may intervene during the process to *disentangle* the structure, or modify the spring tensions to obtain non-overlapping diagrams. Like the **PseudoViewer**, this software does not offer *offline* command-line capabilities, and a dedicated GUI, freely available at **http://jviz.cs.sfu.ca**, must be used to produce drawings.

The **R2R** (Figure 12.F) software also uses a radial layout as its default algorithm. However, more sophisticated layout strategies can be specified, for instance in order to force angles to fall within a restricted list (typically $k \cdot \pi/8, \forall k \in [0,15]$). As illustrated Figure 12.F, another interesting feature is the *flat* drawing of unpaired nucleotides along the exterior loop. Another interesting aspect is the display of comparative information regarding a structural family, represented as a STOCKHOLM-formatted multiple sequence alignment including a consensus structure. The installation and usage of **R2R**, described in details in Section 3.5, is perhaps a bit more involved than its competitors, but the quality of the resulting picture is clearly worth the effort.

3.2.4 Tree layout

In the absence of *crossing* interactions such as pseudoknots, RNA secondary structures can be unambiguously decomposed in a variety of ways, leading to tree-like objects. Such a decomposition of the conformation space is at the core of any dynamic programming scheme, a strategy for solving combinatorial optimization problems which is especially popular in RNA bioinformatics. For instance, atomic contributions in the Turner energy model (28) are entirely determined by the joint knowledge of internal nodes (a base-pair) in combination with their immediate children. Consequently, this representation may be useful to illustrate the principles underlying RNA algorithms.

Unfortunately, there are currently few available options to produce such a representation, and the most realistic option consists in using the general-purpose graph visualization software **GraphViz** (26), which unfortunately requires a DOT-formatted file as input. Consequently, a secondary structure, typically denoted by a dot-parenthesis expression (see Section 2.2.2), will require some conversion. We provide in Note 2 a minimal **python** (2.x) script to convert, and dump to the standard output as a DOT-formatted file, a secondary structure.

Invoking this code for a given sequence/structure, while capturing the output through standard I/O redirection, one obtains a file which can then be fed to the **dot** utility of **GraphViz**. This utility can then be configured to produce quality pictures in virtually any format, including many vector formats such as PDF and SVG.

3.3 Mapping probing data onto secondary structure models

Footprinting techniques are widely applied to map the 2D and 3D structures of RNA using both chemical and/or enzymatic probes. They provide useful information about the relative accessibility of the RNA to a given probe at the nucleotide resolution. Depending on the physico-chemical conditions or the presence of different molecular partners (RNA, proteins, antibiotics, etc.), the chemical or enzymatic probing data give a snapshot of the RNA structure under specific conditions. They are used to study the conformational changes during RNA folding or induced upon protein or ligand binding (29); they also nicely complement the structural data obtained on the conformational changes associated with RNA folding (30) or RNA/protein interactions (31).

With the development of high-throughput techniques using Selective 2'-Hydroxyl acylation Analyzed by Primer Extension (SHAPE) chemistry (SHAPE-seq (32)), UV-crosslinking or immunoprecipitation (HITS-CLIP (33)), emerging computational methods attempt to integrate such data into RNA structure predictions (34, 35, 36). However, these predictions are never entirely accurate, and must be confronted with the experimental data in an iterative refinement. Visualization is therefore an important aspect in the process of

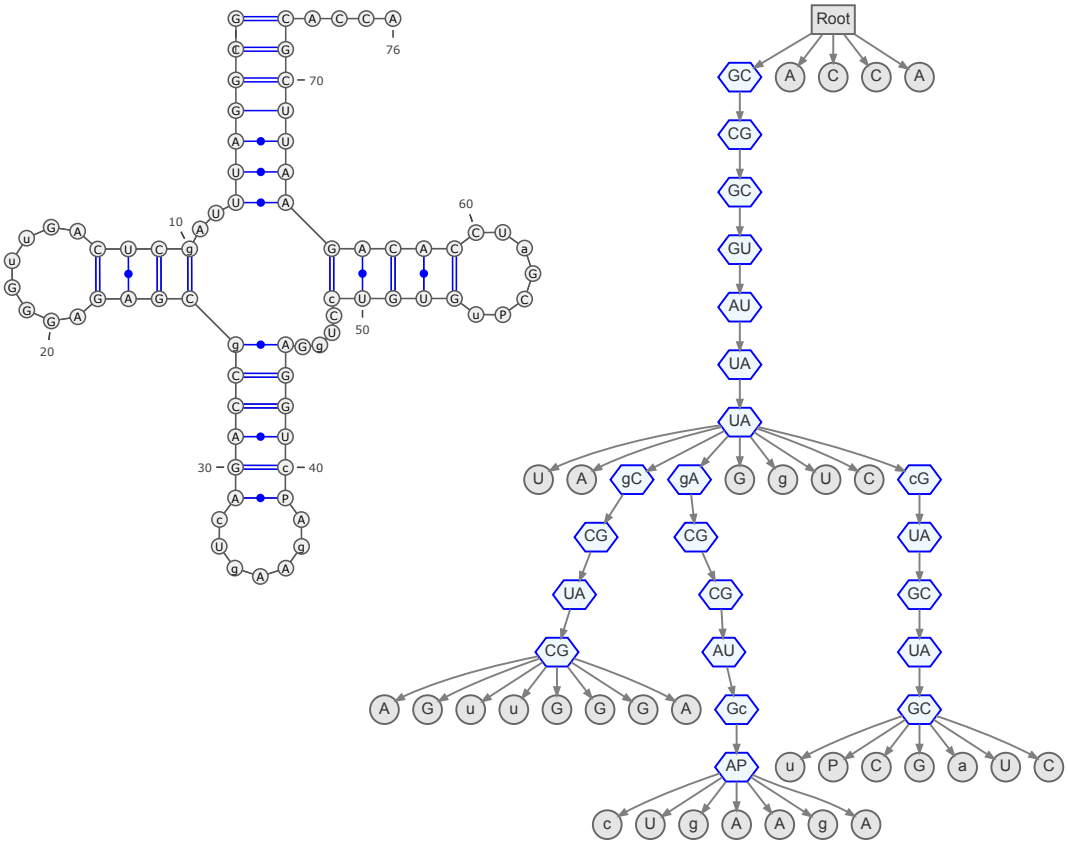


Fig. 13: Planar graph layout using **VARNA** (14) and tree representation using the default options of the **dot** (26) algorithm of **GraphViz** (27) of the secondary structure of a transfer RNA (PDB id: 1TN2:A, annotated using **RNAView** (9)).

generating primary 2D representations and models. Probing data can be represented using graphical annotations (symbols, color maps and marks, labels, etc.) mapped onto a 2D structure.

In this use-case, we focus on probing data obtained on the H/ACA guide RNAs from the snoR9 family (RFAM: RF00065) in *Pyrococcus abyssi* (37). Here, one may map chemical probing data onto a secondary structure by generating different representations for the same RNA molecule, corresponding to chemical probing data obtained under different conditions. Since the structure is used to provide a context for the interpretation of probing data, one will typically use a template for the secondary structure (sequence/data and layout) onto which the molecular properties are displayed. Chemical probing data can be represented in different ways:

- Symbols may be used to indicate the degree of accessibility to some chemical or enzymatic probe (Figure 14.B);
- A color map can also be used to superimpose the accessibility directly onto the secondary structure (Figure 14.C);
- The chemical probing data associated with conformational changes of the RNA (folding or interactions with ligands) can be displayed in different panels to provide a graphical indication of the condition-dependent changes, affecting the tertiary and secondary structures.

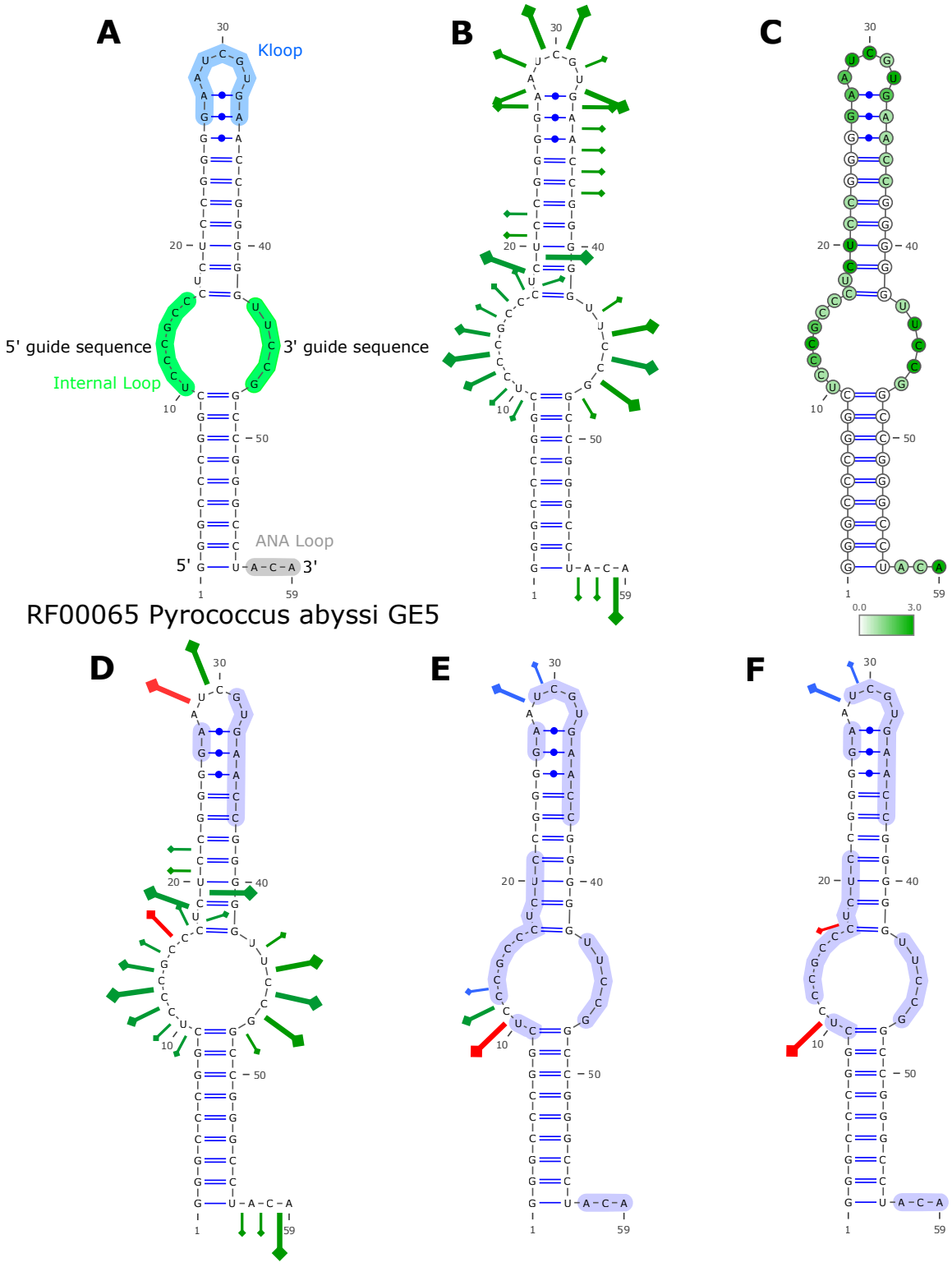


Fig. 14: 2D structure representations integrating chemical probing of the Pab21 HACA sRNP from *P. abyssi* (RFAM: RF00065), as generated by **VARNA**. A: 2D structure representation of the HACA RNA motif highlighting the key structural features. B: Free RNA Chemical probing indicating the accessible positions for Pb cleavage on a scale from 1 to 3 in the intensity of the reaction. C: Alternate representation showing the accessibility using a color map. D-E-F: Modifications of the RNA probing data due to the successive binding of L7Ae, CBF5 and NOP10, and the RNA substrate. Green pins: initial probing on RNA; red pins: increase in reactivity; blue pins: decrease in reactivity; violet regions: newly protected positions.

VARNA offers multiple features to ease the production of such diagrams. Specifically, an extensive array of command-line options can be used to produce a collection of pictures that share a general orientation. Three types of annotations are of particular interest:

- **Color-maps:** A color map associates numerical values to each position in an RNA (`colorMap` option), and uses a (multiple) color gradient (`colorMapStyle` option) to associate colors to each position, based on their associated value. Minimal and maximal values can be manually input (`colorMapMax` and `colorMapMin` options), otherwise they are simply guessed from the values. For color maps, the tedious task of manually inputing the values (e.g. chemical reactivities), on a nucleotide-per-nucleotide basis, can be avoided by loading an external file containing all the data at once from within the GUI. Alternatively, one may use an *ad hoc* script to format the command such that values are already provided to the software, as shown in Note 8 (panel C), whose execution produces Figure 14.C.
- **Glyphs:** Arrows, pins or additional symbols are sometimes used to indicate significantly accessible (backbone) regions, or a difference in accessibility. Position-specific markers can be specified using the `chemProb` options in **VARNA**, as shown in Listings 8 (Panels B,D,E, and F), resulting in the corresponding panels of Figure 14.
- **Highlighted Regions (a.k.a. sausages) :** Regions highlights may be used to emphasize a change of reactivity observed on a portion of the backbone, possibly hinting towards a local conformational change. To produce such annotations, the `highlightRegion` option in **VARNA** may be used, and allows to specify the width and color of the region, as shown in panels D, E and F of Figure 14 (produced as shown in Note 8).

RNAplot can also be used to display accessibility values as a color map, using the `--pre` and `--post` options to *decorate* the PostScript output. Some basic examples can be found in the listing below:

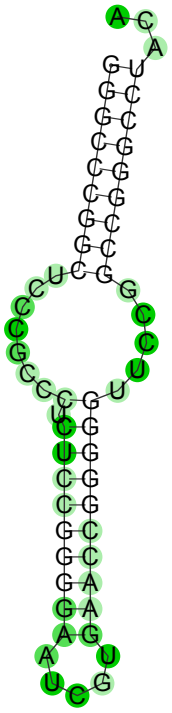
```
# Running RNAplot from a DBN file 'XXX.txt'

# Marking nucleotide 10
RNAplot --post "10 cmark" < XXX.txt
# Drawing backbone for region (10,15) using red color (r=1,g=b=0) and line thickness 2
RNAplot --pre "10 15 2 1. 0. 0. omark" < XXX.txt
# Defining a custom PS macro and using it to fill base number 10 in blue (r=g=0, b=1)
RNAplot --pre "/cfmark {setrgbcolor newpath 1 sub coor exch get aload pop fsize 2 div 0 360 arc fill} bind def 10 0. 0. 1.
     cfmark" < XXX.txt
```

As can be seen in this last example, achieving even simple operations may require an extensive knowledge of the PostScript language, whose suffix-order for operations induces a steep learning curve. Furthermore, formatting the command line manually would quickly become tedious. Therefore, we propose in Listing 1 a minimal **Python** wrapper which, given a DBN formatted RNA and a list of values, invokes **RNAplot** to produce a *color map*-decorated output similar to that of Figure 14.C:

Listing 1: Using `RNAplot` to display probing values as a color map.

```
import os,sys
def getColor(val, minval, maxval,col1=(1.,1.,1.),col2=(0.0,0.8,0.0)):
    (r1,g1,b1),(r2,g2,b2) = col1,col2
    span = float(maxval)-float(minval)
    k = (float(val)-float(minval))/(span if span!=0 else 1.0)
    l = 1.-k
    return (r1*l+r2*k,g1*l+g2*k,b1*l+b2*k)
def formatValuesAsPS(values):
    minval,maxval = min(values),max(values)
    valtab = [" %s %s cfmark"%(i+1,"%.2f %.2f %.2f"%getColor(v,minval,maxval)) for i,v in enumerate(
        values)]
    return  "".join(valtab)
# Invokes RNAplot to display a color map
#   Arg1: path to DBN file
#   Arg2: list of comma-separated values (eg "1,2,3")
if __name__ == "__main__":
    inputFile = sys.argv[1]
    values = sys.argv[2].split(",")
    extraMacro = "/cfmark {setrgbcolor newpath 1 sub coor exch get aload pop fsize 2 div 0 360 arc
        fill} bind def"
    os.system("RNAplot --pre \" %s %s\" < %s"%(extraMacro,formatValuesAsPS(values),inputFile))
```



3.4 Displaying pseudoknots and interactions

Pseudoknots and RNA-RNA interactions depart from the tree-like paradigm, and therefore constitute a challenge to computational methods. Most prediction algorithms address the problem by restricting the search space. However, visualization tools cannot arbitrarily enforce such restrictions. This narrows down the choice of suitable layouts and tools to few options.

3.4.1 Pseudoknots

Basic representations, such as the linear and circular representations, do not require a complicated layout, and therefore remain largely unaffected by the presence of pseudoknot. From a software designer perspective, supporting pseudoknot for these representations is then mostly a matter of being able to parse suitable input formats. **`jViz.RNA`** (Figure15.B), **`R-chie`** and **`VARNA`** (Figure15.D) may be used without much additional effort. In particular, the latter will also support more complicated structures, such as those featuring multiple partners for a given nucleotide, as illustrated by Figure15.D.

However, a large subset of existing pseudoknots can be represented planarly as squiggle plots, i.e. in such a way that base pairs and backbone are non-crossing. To create such representations, the **`PseudoViewer`** is the only fully satisfactory option. Pseudoknots are laid out within dedicated boxes, that are embedded in a tree-like general layout. As shown in Figure 15.A, the result is aesthetically pleasing even in the absence of user intervention, its only drawback being a consequent backbone distortion. Alternatively, the spring layout of **`jViz.RNA`** can be used to obtain decent squiggle plots, as illustrated by Figure 15.C usually after some manual disentangling from the user.

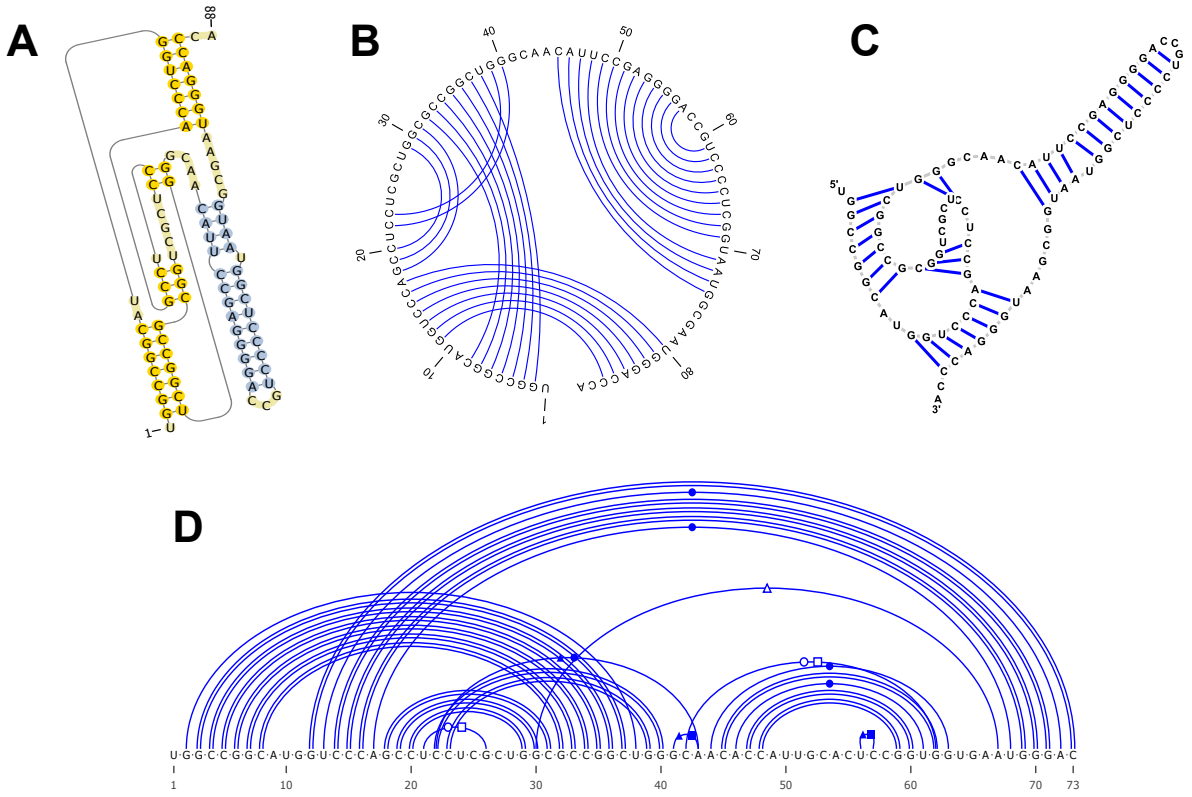


Fig. 15: Pseudoknotted secondary structure of the Hepatitis Delta virus ribozyme (PseudoBase entry PKB75), rendered as a planar squiggle plot by the **PseudoViewer** (**A**), as a circular diagram and as a *force-driven* spring layout by **jViz.RNA** (**B** and **C**). Extended secondary structure of a variant (PDB id: 1SJ3), drawn linearly by **VARNA** (**D**).

3.4.2 Interactions

The need for automated representations of RNA-RNA interactions is relatively recent, and have arisen from a recent increase of interest in the computational prediction of RNA-RNA interactions.

RILogo is currently the only automated tool that produces representations of RNA-RNA interactions. The web version requires as input either one or two multiple sequence alignments (`Stockholm` or a modified version of `FASTA`). In the case of a single file, the consensus secondary structure will use a special character `&` to indicate the end of the first RNA and the beginning of the second. The base-pairs which are split by the separator then be involved in the hybridization of the two RNAs. Alternatively, two `Stockholm` files may be specified, using one or several parentheses systems (`A` / `a`, `B` / `b`...) to indicate the subset of base-pairs involved in the interaction as follows:

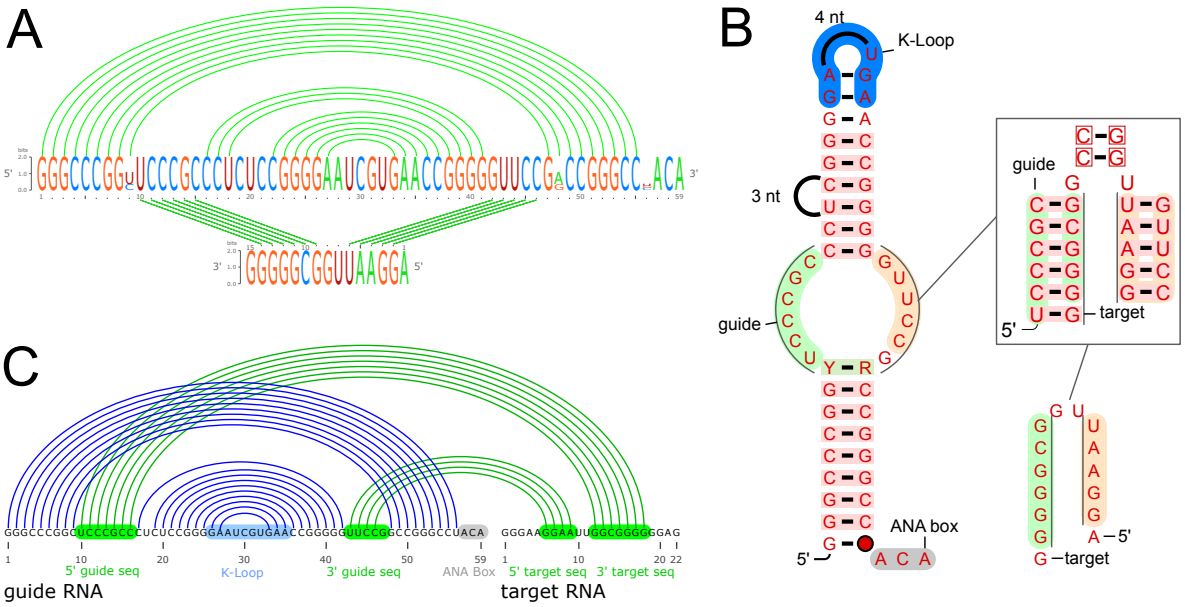


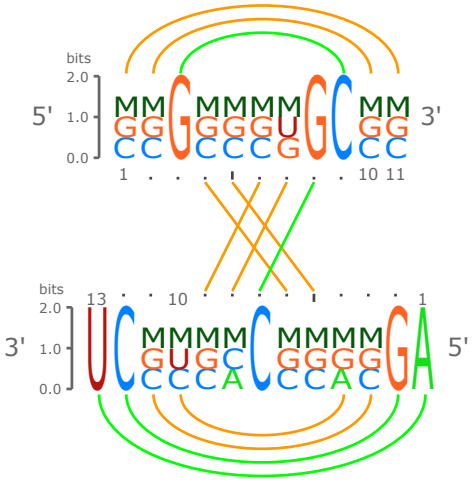
Fig. 16: Available representations for RNA-RNA interactions: consensus secondary structures of the H/ACA guide and its target sequence from the 16S rRNA (RFAM: RF00065) automatically generated by **RILogo** (Panel A), or as a manually-edited pseudoknoted structure using **R2R** (Panel B) and **VARNA** (Panel C) .

Listing 2: Fasta-like input file `XXXX.txt` expected by RILogo

```
>organism1
GCGGGGUGCGC&AGGACCCACUCCU
>organism2
CGGCCCGGCCG&AGCGGGCCGCGCU
>structure
(((AABBB)))&((((aabbb))))
```

Listing 3: Invoking RILogo

```
rilogo XXXX.txt > output.svg
```



Feeding such a file to the **RILogo** webserver or software (both available at **http://rth.dk/resources/rilogo**) as illustrated in Note 3 one obtains arc-annotated (linear) representations of the interacting families, similar to the ones above and shown in Figure 16.A.

Neither **R2R** nor **VARNA** offers a native support for RNA-RNA interactions. However, it is possible to use some tricks to represent consensus sequence and secondary structures for RNA-RNA interactions. Within **R2R**, using templates and options designed for pseudoknots, one may generate some representations where both interacting RNAs are merged into the same alignment using some random sequence spacer. Then, the sequence spacer and its associated annotations may be manually removed using appropriate software (**Inkscape** or commercial alternatives) to display a clean representation of consensus sequences and secondary structures of

two interacting RNAs (see Figure 16.B). **VARNA** may also be used in its linear representation to represent the interaction as a pseudoknot, possibly in combination with annotations, and manually post-process the output to obtain result similar to Figure 16.C.

3.5 Visualizing structure-informed RNA alignments

The identification of novel families of structured ncRNAs relies heavily on the implementation of a comparative approach. This approach starts with a multiple sequence alignments, from which a draft consensus secondary structure is identified. This draft is then used to refine the alignment, and retrieve novel homologs, upon which the secondary structure can be further refined. Iterating these steps leads to a final structure-informed multiple sequence alignment, whose quality must be assessed before concluding on the existence of a new functional family. To that purpose, a global visualization of the conservation/covariation levels in the context of the structure is essential.

The **R2R** software is particularly suited to generate representations of structure-informed RNA alignments. Beside allowing to decorate a squiggle plot using conservation and covariation levels, **R2R** also allows to define subfamilies of sequences, possibly associated with diverging secondary structures. Another unique feature of **R2R** is its definition of a complete set of directives to express preferences. These preferences may then be used to produce sets of similar-looking representations for individual sequences, helping in the visual identification of differences. Figure 17, a panel of secondary structure representations produced for the RF00065 RFAM family, illustrates such capabilities. Here, the consensus structure of the RF00065 family is displayed using variable-length regions and color-shaded backbone regions for sequence elements that vary in size associated with the internal loop, terminal loop or 3'-end regions of the RNAs. Each member of the family is also displayed with the full sequence to view the details of each member in the family. Two subfamilies corresponding to different sizes in the 5' guide sequence (7 or 8 nucleotides) of the internal loop are shown to zoom on the variations in the loop size.

A typical usage of **R2R** includes the following steps:

Step 1 **(Initial Alignment)** Start from a `Stockholm` file `XXXX.sto` featuring a multiple sequence alignment, along with a consensus secondary structure denoted in WUSS notation. The software will refuse a *blocked* alignment (in which sequences are broken into blocks of fixed lengths), and we provide in Listing 4 a minimal **Python** script to transform a blocked `Stockholm` file into a linear one.

Listing 4: Transforming an `Stockholm`-formatted RFAM multiple sequence alignment into an unblocked alignment accepted by `R2R` as input.

```
import os,sys,string
def unblockStockholm(inpath,outpath):
    outfile = open(outpath,"w")
    (seqIDs,seqContents) = ([],{})
    for l in open(inpath):
                if len(l)>12 and ((not l.startswith("#")) or (l[:12] in ["#=GC SS_cons","#=GC RF
                        "])):
                        (id,content) = (l[0:37].strip(),l[37:-1].strip())
                        if id not in seqContents:
                                seqIDs.append(id)
                                seqContents[id] = ""
                        seqContents[id] += content
                elif len(l)>1:
                        for id in seqIDs:
                                outfile.write(string.ljust(id, 37)+seqContents[id]+"\n")
                        seqIDs,seqContents = [],{}
                        outfile.write(l)
    outfile.close()
# Transforms a 'blocked' STOCKHOLM file (eg RFAM alignment) into a linear one
#   Arg1: path to 'blocked' STOCKHOLM file
#   Arg2: path to 'linearized' STOCKHOLM output file
if __name__=="__main__":
    (inStoc,outFile) = (sys.argv[1],sys.argv[2])
    unblockStockholm(inStoc,outFile)
```

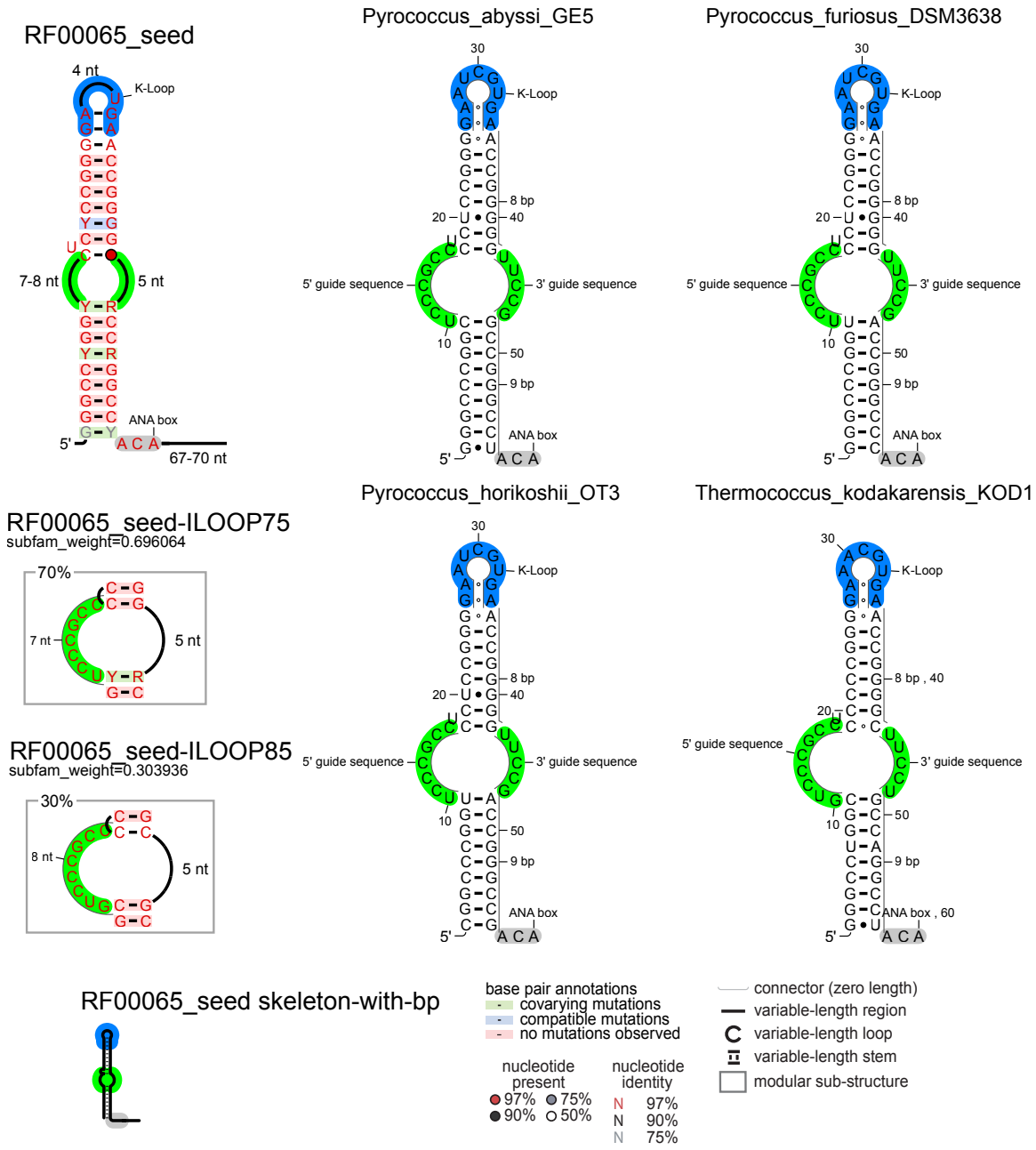


Fig. 17: Consensus sequence and secondary structures of the H/ACA guide RNA from the snoR9 family (RFAM: RF00065) generated by **R2R** from the RFAM seed alignment. Four individual secondary structures from the seed alignment are annotated to display the common structural features (internal loop, K-loop, ANA box) and the subtle differences in sequence and loop size. The modular sub-structures correspond to two subfamilies: ILOOP75 and ILOOP85 where the 5' guide sequence of the internal loop contains either 7 or 8 nucleotides, respectively.

Step 2 **(Computing statistics)** Conservation and covariation levels must be computed, and appended to the alignment into a file `XXXX.cons.sto` by running the command

```
# Compute conservation levels
r2r --GSC-weighted-consensus XXXX.sto XXXX.cons.sto 3 0.97 0.9 0.75 4 0.97 0.9 0.75 0.5 0.1
```

The numerical values occurring in the above command correspond to the recommended thresholds and parameter values, and we direct to the manual any user with specific needs and expectations in this respect.

Step 3 **(User preferences)** The produced file may then be manually edited to specify preferences regarding the layout and annotations. To that purpose, *generic feature* lines, starting with `#GF=` can be added appended to the file prior to the terminal `\\` characters. A huge collection of such features is available, whose complete listing would perhaps require a dedicated chapter. We illustrate some of the most salient features in Notes 4 and 5.

Step 4 **(Batch metafile)** Once these preferences have been expressed and stored within a `Stockholm` file `XXXX.user.sto`, a `meta` file must be defined. In a minimal setting, it may simply contain the name of the processed `Stockholm` input file. More complex examples, as shown in Note 3, may include several lines to produce multiple figures. These figures may either represent the content of multiple `Stockholm` files, focus on specific sequences/organisms (see `oneseq` keyword), or produce a coarse-grained *skeleton* view (see `skeleton-with-pairbonds` keyword). For instance, the following `meta` file produces:

1. a consensus drawing;
2. a drawing for *Pyrococcus furiosus*, adhering to the same layout rules;
3. a *skeleton* view of the consensus.

```
XXXX.user.sto
XXXX.user.sto oneseq Pyrococcus_furiosus
XXXX.user.sto skeleton-with-pairbonds
```

Step 5 **(Running R2R)** Once the `meta` file `XXXX.meta` is ready, the software can simply be run through either of the following commands:

```
r2r XXXX.meta XXXX.pdf
r2r XXXX.meta XXXX.svg
```

to produce either a `PDF` or `SVG` output, amenable to further vector post-processing.

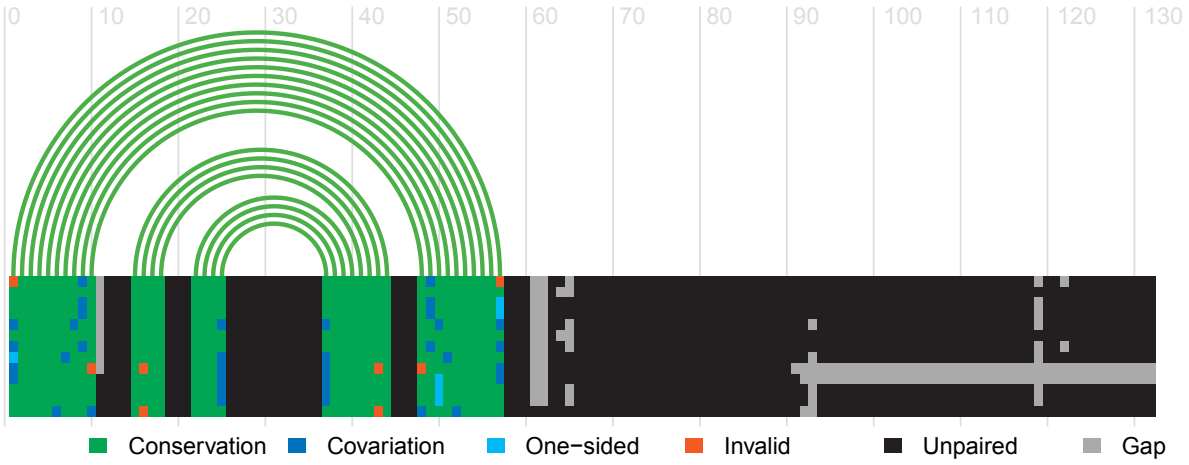


Fig. 18: Structure-informed representation of the RFAM multiple-sequence alignment of the snoR9 family (RFAM: RF00065), produced using **R-Chie**.

The representations offered by **R2R** allow for a critical inspection of a proposed consensus structure. However, they do not necessarily inform the viewer regarding the quality of the underlying alignment. Therefore, we advocate complementing this representation with sequence-focused alternatives produced by **R-chie**, an **R** package which uses a linear layout to draw one or several structures, stacked on top of a multiple sequence alignment. This sequence alignment is typically color-encoded to indicate conservation, insertion, compatibility or contradiction with the consensus structure. **R-chie** can be invoked online, or downloaded at **http://www.e-rna.org/r-chie**. Once the necessary **R** dependencies are installed, it can be executed on a *gapped* FASTA multiple sequence alignment `XXXX.txt` and a Vienna/DBN consensus structure `YYYY.txt` via the following command:

```
Rscript rchie.R --msafile XXXX.txt --colour1 "#4DAF4A" --msacol "#00A651,#0072BC,#00B9F2,#F15A22,#231F20
    ,#AAAAAA,#DA6FAB" --pdf  --output="out.pdf"  --format1 "vienna" YYYY.txt
```

Running the command produces a PDF file named `out.pdf` similar to that shown in Figure 18.

4 Notes

1. **jViz.RNA** tends to create extremely large EPS files, which cannot be easily manipulated within existing vector graphics editors. This problem apparently originates from the way characters are output. Indeed, texts are apparently not exported as PostScript strings, but as splines defined using a large number of points. Besides increasing the file size, this means that the nucleotides and annotations cannot be easily manipulated within dedicated editors. Therefore, unless a bitmap picture is eventually sought and very limited post-processing is involved, we generally do not recommend using **jViz.RNA** to draw RNAs of length greater than 300 nts.
2. Minimal **python** script to convert a secondary structure, denoted by a dot-parenthesis notation, into a **Graphviz** input file (**DOT** format) amenable to a tree-like layout.

```
import sys
# Get custom styles by changing these lines (cf GraphViz manual)
STYLE_DEFAULT = "shape=\"rectangle\",style=filled,margin=\"0,0\",fontsize=20,color=grey40,fontcolor=grey20,fillcolor=grey90,
      fontname=\"Helvetica\""
STYLE_UNPAIRED = "shape=\"circle\",color=blue,fillcolor=aliceblue"
STYLE_PAIRED = "shape=\"hexagon\""
STYLE_EDGES = "color=grey50"
# Converts an RNA sequence/secondary structure (dot-parenthesis notation) into a DOT-formatted GraphViz input, written into
      a previously opened file f
def drawAsDOT (seq,secstr,f):
    print >> f, "digraph rna{\n  node [%s];\n  edge [%s];\n  -1 [label=\"Root\"];"%(STYLE_DEFAULT,STYLE_EDGES)
    stack = [-1]
    for i in range(len(secstr)):
        k = stack[-1]
        if secstr[i]=="(":
            stack.append(i)
            print >> f, "  %s -> %s;"%(k,i)
        elif secstr[i]==")":
            stack.pop()
            print >> f, "  %s [label=\"%s\",%s];"%(k,seq[k]+seq[i],STYLE_PAIRED)
        else:
            print >> f, "  %s -> %s;"%(k,i)
            print >> f, "  %s [label=\"%s\",%s];"%(i,seq[i],STYLE_UNPAIRED)
    print >> f, "}"

#Typical usage of this script (RNAToDOT.py):
#   python RNAToDOT.py (((...))) GGGAUACCC > rnaTree.dot
#   dot -Tpdf -o rnaTree.pdf rnaTree.dot
if __name__=="__main__":
  struct,seq = sys.argv[1],sys.argv[2]
  drawAsDOT (seq,struct,sys.stdout)
```

3. **R2R** meta files including the instructions to process the input files associated with the different representations generated in Figures 16 and 17.

```
RF00065_guide_target.cons.sto
RF00065_guide_target.sto    oneseq  Pa21-S892
RF00065_guide_target-hybrid.cons.sto
```

```
RF00065_seed.cons.sto
RF00065_seed.sto    oneseq  Pyrococcus_furiosus
RF00065_seed.sto    oneseq  Pyrococcus_abyssi_GE
RF00065_seed.sto    oneseq  Pyrococcus_horikoshi
RF00065_seed.sto    oneseq  Thermococcus_kodakar
RF00065_seed.cons.sto   skeleton-with-pairbonds
RF00065_seed-ILOOP75.cons.sto
RF00065_seed-ILOOP85.cons.sto
```

4. Stockholm seed alignment of the snoR9 family (RFAM ID: RF00065) decorated with **R2R** labels to generate a series of consensus and secondary structures for RNA-RNA interactions (see Figure 16 for the graphical rendering).

```
# STOCKHOLM 1.0

Pa21-S892                   GGGCCCGGC.....UCCCGCCCUCUCCGGG...GA.............AUC.................GUGA.ACCG.GG.GGUUCCG..G.CCGGGCCU..ACA.
      UUUUUUUUUUUUUUAGGAAUUGGCGGGGGGNNNNN
Ph21-S880                    GGGCCCGGU.....UCCCGCCCUCUCCGGG...GA.............AUC.................GUGA.ACCG.GG.GGUUCCG..A.CCGGGCCG..ACA.
      UUUUUUUUUUUUUUAGGAAUUGGCGGGGGNNNNN
Pf1-S880                    GGGCCCGGU.....UCCCGCCCUCUCCGGG...GA.............AUC.................GUGA.ACCG.GG.GGUUCCG..A.CCGGGCCC..ACA.
      UUUUUUUUUUUUUUAGGAAUUGGCGGGGGNNNNN
#=GC SS_cons                <<<<<<<<<...........<<<...<<<<...<<.............________.........-..>>.>>>>->>->.......->->>>>>>>>.............
      ..........................
#=GC SS_cons_l               .............<<<<<<.....................................................................<<<<<.....................
      ........>>>>>..>>>>>>.......
#=GC R2R_LABEL               q...........1LL1LLL...e.f...........4.p.................p.........5KHH.HHiH.HHBHLLLLLj78H9HHHhHHHr....a.
      CCCCC20CCCCCCC.GGGGGmonGGGGGg.....
#=GC SUBFAM_hybrid_R2R_LABEL --------------AAaAAAbf-------------------------------------------------------g.chHHHH--------------------------
      --------DDDDDmonEEEEi------
#=GC R2R_XLABEL_KLOOP        ..............................KKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKK.....................................
      ..........................
#=GC R2R_XLABEL_KLOOPL       ..................................................k.....................................................................
      ..........................
#=GC R2R_XLABEL_ILOOP        ..............PQXXXX..W.....................................................................YYYYYRSU.V.....................
      ........YYYYY..PPPPPP......
#=GC R2R_XLABEL_ILOOPL       ..................x....w.....................................................................y..su.v.....................
      ..........................
#=GC R2R_XLABEL_ANA          ..........................................................................................................CCC........
      ..........................

#=GF R2R ignore_ss_except_for_pairs _l outline_only_bp
#=GF R2R place_explicit 2 2-- -45 1 0 0 0 -90
#=GF R2R place_explicit n n-- -45 1 0 0 0 -90
#=GF R2R place_explicit m m-- -45 1 0 0 0 -90
#=GF R2R delcols 0
#=GF R2R tick_label l guide
#=GF R2R tick_label K K-Loop
#=GF R2R tick_label a ANA box
#=GF R2R tick_label g target
#=GF R2R keep q r
#=GF R2R var_backbone_range 4 5
#=GF R2R outline_nuc L
#=GF R2R outline_nuc l
#=GF R2R outline_nuc G
#=GF R2R outline_nuc n
#=GF R2R outline_nuc j
#=GF R2R circle_nuc 2 rgb:0,0,0
#=GF R2R circle_nuc C rgb:0,0,0
#=GF R2R nuc_color 2 rgb:0,0,0
#=GF R2R nuc_color C rgb:0,0,0
#=GF R2R var_backbone_range_size_fake_nucs 3 e f
#=GF R2R shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R shade_along_backbone ANA:C rgb:200,200,200
#=GF R2R outline_along_backbone ILOOP:Y rgb:255,228,196
#=GF R2R outline_along_backbone ILOOP:P rgb:193,255,193
#=GF R2R outline_along_backbone ILOOP:Q rgb:193,255,193
#=GF R2R outline_along_backbone ILOOP:X rgb:193,255,193
#=GF R2R shade_along_backbone ILOOP:Y rgb:255,228,196
#=GF R2R shade_along_backbone ILOOP:P rgb:193,255,193
#=GF R2R shade_along_backbone ILOOP:Q rgb:193,255,193
#=GF R2R shade_along_backbone ILOOP:X rgb:193,255,193

#=GF SUBFAM_PERL_PRED hybrid return 1;
#=GF SUBFAM_hybrid_R2R subst_ss _l primary outline_only_bp
#=GF SUBFAM_hybrid_R2R box_nuc b rgb:200,0,0
#=GF SUBFAM_hybrid_R2R box_nuc f rgb:200,0,0
#=GF SUBFAM_hybrid_R2R box_nuc g rgb:200,0,0
#=GF SUBFAM_hybrid_R2R box_nuc c rgb:200,0,0
#=GF SUBFAM_hybrid_R2R outline_nuc E
#=GF SUBFAM_hybrid_R2R outline_nuc D
#=GF SUBFAM_hybrid_R2R outline_nuc m
#=GF SUBFAM_hybrid_R2R outline_nuc n
#=GF SUBFAM_hybrid_R2R outline_nuc i
#=GF SUBFAM_hybrid_R2R set_dir pos0 -90
#=GF SUBFAM_hybrid_R2R place_explicit b b-- +45 3 0 0 0 0
#=GF SUBFAM_hybrid_R2R place_explicit g g-- +90 1.5 0 0 0 90
#=GF SUBFAM_hybrid_R2R place_explicit c c-- 90 1 0 0 0 0
#=GF SUBFAM_hybrid_R2R place_explicit h h-- +45 3 0 0 0 90
#=GF SUBFAM_hybrid_R2R place_explicit m m-- 0 1 0 0 0 -90
#=GF SUBFAM_hybrid_R2R place_explicit o o-- 0 2.5 0 0 0 90
#=GF SUBFAM_hybrid_R2R tick_label i target
#=GF SUBFAM_hybrid_R2R tick_label a guide
#=GF SUBFAM_hybrid_R2R shade_along_backbone A rgb:193,255,193
#=GF SUBFAM_hybrid_R2R shade_along_backbone a rgb:193,255,193
#=GF SUBFAM_hybrid_R2R shade_along_backbone E rgb:193,255,193
#=GF SUBFAM_hybrid_R2R shade_along_backbone n rgb:193,255,193
#=GF SUBFAM_hybrid_R2R shade_along_backbone D rgb:255,228,196
```

```
#=GF SUBFAM_hybrid_R2R shade_along_backbone d rgb:255,228,196
#=GF SUBFAM_hybrid_R2R shade_along_backbone h rgb:255,228,196
#=GF SUBFAM_hybrid_R2R shade_along_backbone H rgb:255,228,196

#=GF R2R_oneseq Pa21-S892 shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq Pa21-S892 shade_along_backbone ILOOP:P rgb:193,255,193
#=GF R2R_oneseq Pa21-S892 shade_along_backbone ILOOP:Q rgb:193,255,193
#=GF R2R_oneseq Pa21-S892 shade_along_backbone ILOOP:X rgb:193,255,193
#=GF R2R_oneseq Pa21-S892 shade_along_backbone ILOOP:Y rgb:255,228,196
#=GF R2R_oneseq Pa21-S892 shade_along_backbone ANA:C rgb:200,200,200

//
```

5. Stockholm seed alignment of the snoR9 family (RFAM ID: RF00065) decorated with **R2R** labels to generate a series of consensus and secondary structures (see Figure 17 for the graphical rendering).

```
# STOCKHOLM 1.0
#=GF ID    snoR9
#=GF AC    RF00065
#=GF DE    Small nucleolar RNA snoR9
#=GF AU    Bateman A, Daub J
#=GF GA    50.0
#=GF NC    49.8
#=GF TC    68.7
#=GF SE    Bateman A
#=GF SS    Published; PMID:12032319
#=GF TP    Gene; snRNA; snoRNA; CD-box;
#=GF BM    cmbuild  -F CM SEED; cmcalibrate --mpi -s 1 CM
#=GF BM    cmsearch  -Z 274931 -E 1000000  --toponly  CM SEQDB
#=GF DR    SO:0000593 SO:C_D_box_snoRNA
#=GF DR    GO:0006396 GO:RNA processing
#=GF DR    GO:0005730 GO:nucleolus
#=GF RN    [1]
#=GF RM    12032319
#=GF RT    Noncoding RNA genes identified in AT-rich hyperthermophiles.
#=GF RA    Klein RJ, Misulovin Z, Eddy SR;
#=GF RL    Proc Natl Acad Sci U S A 2002;99:7542-7547.
#=GF CC    snoRNA R9 is a member of the C/D class of snoRNA which contain
#=GF CC    the C (UGAUGA) and D (CUGA) box motifs.  R9 was identified in a
#=GF CC    computational screen in AT-rich hyperthermophiles [1]. R9 was
#=GF CC    found to overlap with the smaller snoRNA R19 which is currently a
#=GF CC    member of Pyrococcus C/D box snoRNA family Rfam:RF00095.
#=GF WK    http://en.wikipedia.org/wiki/Small_nucleolar_RNA_snoR9
#=GF SQ    5

#=GS Pyrococcus_furiosus    AC    AE009950.1/163991-163864
#=GS Pyrococcus_abyssi_GE   AC    AJ248283.1/230575-230449
#=GS Pyrococcus_horikoshi   AC    BA000001.2/215834-215709
#=GS P.furiosus             AC    AF468960.1/1-128
#=GS Thermococcus_kodakar   AC    AP006878.1/47908-47779

Pyrococcus_furiosus                GGGCCCGGUU.CCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGACCGGGCCCACA..
       AUGGGAUGAUGACCUUUUGCUUUACUGAACACAUGAUGACCACGCCCUUCGCUGAC.CUAAAUAUUUGAC
Pyrococcus_abyssi_GE               GGGCCCGGCU.CCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA..G..
       UUAUGAUGAACUUUUGCUUUGCUGAUGUGGUGAUGAGCACGCCCUUCGCUGAUACUCUCUCGUCCAU
Pyrococcus_horikoshi               CGGCCCGGUU.CCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGACCGGGCCGACA..GG.
       GGAUGAAGAGCUUUUGCUUUGCUGAGCAGAUGAUGACCACGCCCUUCGCUGAC.CU.GCUAUUUGAC
P.furiosus                         GGGCCCGGUU.CCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGACCGGGCCCACA..
       AUGGGAUGAUGACCUUUUGCUUUACUGAACACAUGAUGACCACGCCCUUCGCUGAC.CUAAAUAUUUGAC
Thermococcus_kodakar               GGGCCUGGCGUCCCGCCCUCCCCGGGGAAACGUGAACCGGGGCUUCCUGCCAGGCCUACACCGGGGGAUGAAGAGCUUUUGCUUUGCUGAC..
       UGUGAUGAGCACGCCCUUCACUGACCCCGUAUCAGCUCU
#=GC SS_cons                       <<<<<<<<<.........<.<<<<<<<<<.....>>>>>>>>>>.....>>>>>>>>>.....................................................
       .........................
#=GC RF                            gGGCCCGGcu.CCCgCCCUCUCCGGGGAAUCGUGAACCGGGGGuUCCggCCGGGCCcACA..
       auguuAUGAUGAaCUUUUGCUUUaCUGAagagaUGAUGAgCACGCCCUUCGCUGAu.CUaaaUauUugAu
#=GR Pyrococcus_furiosus DEL_COLS  .....................................................------------------------------------------
       ------------------------
#=GR Pyrococcus_abyssi_GE DEL_COLS .....................................................------------------------------------------
       ------------------------
#=GR Pyrococcus_horikoshi DEL_COLS .....................................................------------------------------------------
       ------------------------
#=GR P.furiosus DEL_COLS           .....................................................------------------------------------------
       ------------------------
#=GR Thermococcus_kodakar DEL_COLS .....................................................------------------------------------------
       ------------------------
#=GC R2R_LABEL                     p........1......2..........3..4...QQQQqQQQ5...6RRRRrRRRs...7.....................................................
       .......................8
#=GC R2R_XLABEL_KLOOP              ..........................KKKKKKKKK.....................................................................
       .........................
#=GC R2R_XLABEL_KLOOPL             ................................k.....................................................................
       .........................
#=GC R2R_XLABEL_ILOOP              .........IIIIIIII.........................JJJJJ.....................................................................
       .........................
#=GC R2R_XLABEL_ILOOPL             .............i.................................j.....................................................................
       .........................
#=GC R2R_XLABEL_ANA                ...............................................................CCC...................................................
       .........................
#=GC R2R_XLABEL_ANAL               ...............................................................c.....................................................
       .........................
#=GC SUBFAM_LABEL_TERM             ..........x.....................................................................................................
       .........................
#=GC SUBFAM_ILOOP75_R2R_LABEL      -------<..........B>----------------------<.KLMNO.>------------------------------------------------------
       ------------------------
#=GC SUBFAM_ILOOP85_R2R_LABEL      -------<..........B>----------------------<.KLMNO.>------------------------------------------------------
       ------------------------

#=GF R2R keep p s
#=GF R2R var_backbone_range 1 2
#=GF R2R var_backbone_range 3 4
#=GF R2R var_backbone_range 5 6
#=GF R2R var_backbone_range 7 8
```

```
#=GF R2R shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R shade_along_backbone ILOOP:J rgb:0,255,0
#=GF R2R shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R shade_along_backbone ANA:C rgb:200,200,200
#=GF R2R tick_label ANAL:c ANA box
#=GF R2R tick_label KLOOPL:k K-Loop

#=GF SUBFAM_REGEX_PRED ILOOP75 TERM [.]
#=GF SUBFAM_REGEX_PRED ILOOP85 TERM [A-Z]

#=GF SUBFAM_ILOOP75_R2R no5
#=GF SUBFAM_ILOOP75_R2R shade_along_backbone ILOOP:I rgb:0,255,0
#=GF SUBFAM_ILOOP75_R2R tick_label ILOOPL:i 7 nt
#=GF SUBFAM_ILOOP75_R2R outline_nuc ILOOP:I
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l M M 5 nt
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l B B
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l K K
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l L L
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l N N
#=GF SUBFAM_ILOOP75_R2R var_backbone_range_size_fake_nucs l O O

#=GF SUBFAM_ILOOP85_R2R no5
#=GF SUBFAM_ILOOP85_R2R shade_along_backbone ILOOP:I rgb:0,255,0
#=GF SUBFAM_ILOOP85_R2R tick_label ILOOPL:i 8 nt
#=GF SUBFAM_ILOOP85_R2R outline_nuc ILOOP:I
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l M M 5 nt
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l B B
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l K K
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l L L
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l N N
#=GF SUBFAM_ILOOP85_R2R var_backbone_range_size_fake_nucs l O O

#=GF R2R_oneseq Pyrococcus_furiosus shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_furiosus tick_label ILOOPL:i 5' guide sequence
#=GF R2R_oneseq Pyrococcus_furiosus shade_along_backbone ILOOP:J rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_furiosus tick_label ILOOPL:j 3' guide sequence
#=GF R2R_oneseq Pyrococcus_furiosus shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq Pyrococcus_furiosus var_backbone_range_size_fake_nucs 10 7 8
#=GF R2R_oneseq Pyrococcus_furiosus inline_nuc ILOOP:I
#=GF R2R_oneseq Pyrococcus_furiosus inline_nuc ILOOP:J
#=GF R2R_oneseq Pyrococcus_furiosus inline_nuc KLOOP:K
#=GF R2R_oneseq Pyrococcus_furiosus outline_nuc Q
#=GF R2R_oneseq Pyrococcus_furiosus outline_nuc q
#=GF R2R_oneseq Pyrococcus_furiosus outline_nuc R
#=GF R2R_oneseq Pyrococcus_furiosus outline_nuc r
#=GF R2R_oneseq Pyrococcus_furiosus outline_nuc s
#=GF R2R_oneseq Pyrococcus_furiosus tick_label q 8 bp
#=GF R2R_oneseq Pyrococcus_furiosus tick_label r 9 bp

#=GF R2R_oneseq Pyrococcus_abyssi_GE shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_abyssi_GE tick_label ILOOPL:i 5' guide sequence
#=GF R2R_oneseq Pyrococcus_abyssi_GE shade_along_backbone ILOOP:J rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_abyssi_GE tick_label ILOOPL:j 3' guide sequence
#=GF R2R_oneseq Pyrococcus_abyssi_GE shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq Pyrococcus_abyssi_GE var_backbone_range_size_fake_nucs 10 7 8
#=GF R2R_oneseq Pyrococcus_abyssi_GE inline_nuc ILOOP:I
#=GF R2R_oneseq Pyrococcus_abyssi_GE inline_nuc ILOOP:J
#=GF R2R_oneseq Pyrococcus_abyssi_GE inline_nuc KLOOP:K
#=GF R2R_oneseq Pyrococcus_abyssi_GE outline_nuc Q
#=GF R2R_oneseq Pyrococcus_abyssi_GE outline_nuc q
#=GF R2R_oneseq Pyrococcus_abyssi_GE outline_nuc R
#=GF R2R_oneseq Pyrococcus_abyssi_GE outline_nuc r
#=GF R2R_oneseq Pyrococcus_abyssi_GE outline_nuc s
#=GF R2R_oneseq Pyrococcus_abyssi_GE tick_label q 8 bp
#=GF R2R_oneseq Pyrococcus_abyssi_GE tick_label r 9 bp

#=GF R2R_oneseq Pyrococcus_horikoshi shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_horikoshi tick_label ILOOPL:i 5' guide sequence
#=GF R2R_oneseq Pyrococcus_horikoshi shade_along_backbone ILOOP:J rgb:0,255,0
#=GF R2R_oneseq Pyrococcus_horikoshi tick_label ILOOPL:j 3' guide sequence
#=GF R2R_oneseq Pyrococcus_horikoshi shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq Pyrococcus_horikoshi var_backbone_range_size_fake_nucs 10 7 8
#=GF R2R_oneseq Pyrococcus_horikoshi inline_nuc ILOOP:I
#=GF R2R_oneseq Pyrococcus_horikoshi inline_nuc ILOOP:J
#=GF R2R_oneseq Pyrococcus_horikoshi inline_nuc KLOOP:K
#=GF R2R_oneseq Pyrococcus_horikoshi outline_nuc Q
#=GF R2R_oneseq Pyrococcus_horikoshi outline_nuc q
#=GF R2R_oneseq Pyrococcus_horikoshi outline_nuc R
#=GF R2R_oneseq Pyrococcus_horikoshi outline_nuc r
#=GF R2R_oneseq Pyrococcus_horikoshi outline_nuc s
#=GF R2R_oneseq Pyrococcus_horikoshi tick_label q 8 bp
#=GF R2R_oneseq Pyrococcus_horikoshi tick_label r 9 bp

#=GF R2R_oneseq P.furiosus shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R_oneseq P.furiosus tick_label ILOOPL:i 5' guide sequence
#=GF R2R_oneseq P.furiosus shade_along_backbone ILOOP:J rgb:0,255,0
#=GF R2R_oneseq P.furiosus tick_label ILOOPL:j 3' guide sequence
#=GF R2R_oneseq P.furiosus shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq P.furiosus var_backbone_range_size_fake_nucs 10 7 8
#=GF R2R_oneseq P.furiosus inline_nuc ILOOP:I
#=GF R2R_oneseq P.furiosus inline_nuc ILOOP:J
#=GF R2R_oneseq P.furiosus inline_nuc KLOOP:K
#=GF R2R_oneseq P.furiosus outline_nuc Q
#=GF R2R_oneseq P.furiosus outline_nuc q
#=GF R2R_oneseq P.furiosus outline_nuc R
#=GF R2R_oneseq P.furiosus outline_nuc r
#=GF R2R_oneseq P.furiosus outline_nuc s
#=GF R2R_oneseq P.furiosus tick_label q 8 bp
#=GF R2R_oneseq P.furiosus tick_label r 9 bp

#=GF R2R_oneseq Thermococcus_kodakar shade_along_backbone ILOOP:I rgb:0,255,0
#=GF R2R_oneseq Thermococcus_kodakar tick_label ILOOPL:i 5' guide sequence
#=GF R2R_oneseq Thermococcus_kodakar shade_along_backbone ILOOP:J rgb:0,255,0
```

```
#=GF R2R_oneseq Thermococcus_kodakar tick_label ILOOPL:j 3' guide sequence
#=GF R2R_oneseq Thermococcus_kodakar shade_along_backbone KLOOP:K rgb:0,129,255
#=GF R2R_oneseq Thermococcus_kodakar var_backbone_range_size_fake_nucs 10 7 8
#=GF R2R_oneseq Thermococcus_kodakar inline_nuc ILOOP:I
#=GF R2R_oneseq Thermococcus_kodakar inline_nuc ILOOP:J
#=GF R2R_oneseq Thermococcus_kodakar inline_nuc KLOOP:K
#=GF R2R_oneseq Thermococcus_kodakar outline_nuc Q
#=GF R2R_oneseq Thermococcus_kodakar outline_nuc q
#=GF R2R_oneseq Thermococcus_kodakar outline_nuc R
#=GF R2R_oneseq Thermococcus_kodakar outline_nuc r
#=GF R2R_oneseq Thermococcus_kodakar outline_nuc s
#=GF R2R_oneseq Thermococcus_kodakar tick_label q 8 bp
#=GF R2R_oneseq Thermococcus_kodakar tick_label r 9 bp

#=GF Makefile skeleton-with-pairbonds
//
```

6. Shell script to execute the **R2R** program:

```
#!/bin/tcsh -f
#

set R2RPATH = ""
if ( $R2RPATH == "" ) then
echo "Usage: please set the R2RPATH variable"
exit
endif

set r2r = "$R2RPATH/src/r2r"
set subfam = "perl $R2RPATH/src/SelectSubFamilyFromStockholm.pl"
set GSCparams = '3 0.97 0.9 0.75 4 0.97 0.9 0.75 0.5 0.1'

$r2r --GSC-weighted-consensus RF00065_seed.sto RF00065_seed.cons.sto $GSCparams

$subfam RF00065_seed.cons.sto ILOOP75 > RF00065_seed-ILOOP75.sto
$r2r --GSC-weighted-consensus RF00065_seed-ILOOP75.sto RF00065_seed-ILOOP75.cons.sto $GSCparams
$subfam RF00065_seed.cons.sto ILOOP85 > RF00065_seed-ILOOP85.sto
$r2r --GSC-weighted-consensus RF00065_seed-ILOOP85.sto RF00065_seed-ILOOP85.cons.sto $GSCparams

setenv R2R_DUMP_FILE RF00065_seed.txt
$r2r RF00065_seed.meta RF00065_seed.pdf
$r2r RF00065_seed.meta RF00065_seed.svg


$r2r --GSC-weighted-consensus RF00065_guide_target.sto RF00065_guide_target.cons.sto $GSCparams

$subfam RF00065_guide_target.cons.sto hybrid > RF00065_guide_target-hybrid.sto
$r2r --GSC-weighted-consensus RF00065_guide_target-hybrid.sto RF00065_guide_target-hybrid.cons.sto $GSCparams

setenv R2R_DUMP_FILE RF00065_guide_target.txt
$r2r RF00065_guide_target.meta RF00065_guide_target.pdf
$r2r RF00065_guide_target.meta RF00065_guide_target.svg
```

7. Execution of the command-line standalone version of **VARNA** to generate Arc representations (see Figure 10 for graphical rendering).

```
# Top
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-highlightRegion "10-16:radius=10,fill=#acf269,outline=#acf269;43-47:radius=10,fill=#acf269,outline=#acf269" \
-o RF00065_testarc1.svg -algorithm line -drawBases False -spaceBetweenBases 0.6 -bpStyle line

# Bottom
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-chemProb "10-11:glyph=pin,dir=out,intensity=3,color=#f90000;17-18:glyph=pin,dir=out,intensity=1,color=#f90000;"\
-highlightRegion "26-27:radius=10,fill=#d2ccf4,outline=#d2ccf4;31-37:radius=10,fill=#d2ccf4,outline=#d2ccf4" \
-o RF00065_testarc2.svg -algorithm line -drawBases False -spaceBetweenBases 0.6 -bpStyle line
```

8. Execution of the command-line standalone version of **VARNA** for a series of representations (see Figure 14 for the graphical rendering).

```
# panel A
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-highlightRegion "10-16:radius=10,fill=#acf269,outline=#acf269;43-47:radius=10,fill=#acf269,outline=#acf269" \
-o RF00065_panelA.svg -algorithm radiate -flat true -drawBases False -spaceBetweenBases 0.6 \
-title "RF00065: Pyrococcus abyssi GE" -titleSize 12

# panel B
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-chemProb "9-10:glyph=pin,dir=out,intensity=1,color=#3e844b;10-11:glyph=pin,dir=out,intensity=1,color=#3e844b;"\
```

```
-o RF00065_panelB.svg -algorithm radiate -flat true -drawBases False -spaceBetweenBases 0.6

# panel C
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-colorMap "0;0;0;0;0;0;0;0;0;1;1;2;3;2;1;1;1;1;3;3;1;1;0;0;0;2;2;2;3;3;1;3;2;\
1;1;1;1;0;0;0;0;0;1;3;3;3;1;0;0;0;0;0;0;0;0;0;1;1;3" \
-colorMapMax 3 -colorMapMin 0 -colorMapStyle green \
-o RF00065_panelC.svg -algorithm radiate -flat true -drawBases True -spaceBetweenBases 0.6

# panels D, E, F
java -cp VARNAvx-y.jar fr.orsay.lri.varna.applications.VARNAcmd \
-sequenceDBN "GGGCCCGGCUCCCGCCCUCUCCGGGGAAUCGUGAACCGGGGGUUCCGGCCGGGCCUACA" \
-structureDBN "(((((((((.......(.((((((((((.....)))))))))).....)))))))))..." \
-chemProb "9-10:glyph=pin,dir=out,intensity=1,color=#3e844b;10-11:glyph=pin,dir=out,intensity=1,color=#3e844b;"\
-highlightRegion "26-27:radius=10,fill=#d2ccf4,outline=#d2ccf4;31-37:radius=10,fill=#d2ccf4,outline=#d2ccf4" \
-o RF00065_panelD.svg -algorithm radiate -flat true -drawBases False -spaceBetweenBases 0.6
```

Acknowledgements Both the authors wish to thank the French *Centre National de la Recherche Scientifique* (CNRS) for its continued support throughout the years. YP acknowledges the NSF-funded RNA Ontology Consortium, and Jim Procter for stimulating discussions.

References

[1] G. Muller, C. Gaspin, A. Etienne, and E. Westhof. Automatic display of RNA secondary structures. *Comput Appl Biosci*, 9(5):551–561, Oct 1993. doi:10.1093/bioinformatics/9.5.551.

[2] Kay C. Wiese, Edward Glen, and Anna Vasudevan. JViz.Rna–a Java tool for RNA secondary structure visualization. *IEEE Trans Nanobioscience*, 4(3):212–218, Sep 2005. doi:10.1109/TNB.2005.853646.

[3] Boris Shabash, Kay C. Wiese, and Edward Glen. Improving the Portability and Performance of jViz.RNA - A Dynamic RNA Visualization Software. In *PRIB*, pages 82–93, 2012. doi:10.1007/978-3-642-34123-6_8.

[4] Yanga Byun and Kyungsook Han. PseudoViewer3: generating planar drawings of large-scale RNA structures with pseudoknots. *Bioinformatics*, 25(11):1435–1437, Jun 2009. doi:10.1093/bioinformatics/btp252.

[5] Alexander Kaiser, Jan Krüger, and Dirk J Evers. RNA Movies 2: sequential animation of RNA secondary structures. *Nucleic Acids Res*, 35(Web Server issue):W330–W334, Jul 2007. doi:10.1093/nar/gkm309.

[6] Peter Menzel, Stefan E. Seemann, and Jan Gorodkin. RILogo: visualizing RNA-RNA interactions. *Bioinformatics*, 28(19):2523–2526, Oct 2012. doi:10.1093/bioinformatics/bts461.

[7] Daniel Lai, Jeff R Proctor, Jing Yun A Zhu, and Irmtraud M Meyer. R-CHIE: a web server and R package for visualizing RNA secondary structures. *Nucleic Acids Res*, 40(12):e95, Jul 2012. doi:10.1093/nar/gks241.

[8] Andreas R Gruber, Ronny Lorenz, Stephan H Bernhart, Richard Neuböck, and Ivo L Hofacker. The Vienna RNA websuite. *Nucleic Acids Res*, 36(Web Server issue):W70–W74, Jul 2008. doi:10.1093/nar/gkn188.

[9] H. Yang, F. Jossinet, N. Leontis, L. Chen, J. Westbrook, H.M. Berman, and E. Westhof. Tools for the automatic identification and classification of RNA base pairs. *Nucleic Acids Res*, 31(13):3450–3460, 2003. doi:10.1093/nar/gkg529.

[10] Peter De Rijk, Jan Wuyts, and Rupert De Wachter. RnaViz 2: an improved representation of RNA secondary structure. *Bioinformatics*, 19(2):299–300, Jan 2003. doi:10.1093/bioinformatics/19.2.299.

[11] Zasha Weinberg and Ronald R Breaker. R2R–software to speed the depiction of aesthetic consensus RNA secondary structures. *BMC Bioinformatics*, 12:3, 2011. doi:10.1186/1471-2105-12-3.

[12] Fabrice Jossinet and Eric Westhof. Sequence to Structure (S2S): display, manipulate and interconnect RNA data from sequence to structure. *Bioinformatics*, 21(15):3320–3321, Aug 2005. doi:10.1093/bioinformatics/bti504.

[13] Fabrice Jossinet, Thomas E Ludwig, and Eric Westhof. Assemble: an interactive graphical tool to analyze and build RNA architectures at the 2D and 3D levels. *Bioinformatics*, 26(16):2057–2059, Aug 2010. doi:10.1093/bioinformatics/btq321.

[14] Kévin Darty, Alain Denise, and Yann Ponty. VARNA: Interactive drawing and editing of the RNA secondary structure. *Bioinformatics*, 25(15):1974–5, August 2009. doi:10.1093/bioinformatics/btp250.

[15] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S. R. Eddy. Rfam: an RNA family database. *Nucleic Acids Res*, 31(1):439–441, 2003. doi:10.1093/nar/gkg006.

[16] Mirela Andronescu, Vera Bereg, Holger Hoos, and Anne Condon. RNA STRAND: The RNA secondary structure and statistical analysis database. *BMC Bioinformatics*, 9(1):340, 2008. doi:10.1186/1471-2105-9-340.

[17] Neocles B Leontis, Russ B Altman, Helen M Berman, Steven E Brenner, James W Brown, David R Engelke, Stephen C Harvey, Stephen R Holbrook, Fabrice Jossinet, Suzanna E Lewis, François Major, David H Mathews, Jane S Richardson, James R Williamson, and Eric Westhof. The RNA Ontology Consortium: an open invitation to the RNA community. *RNA*, 12(4):533–541, Apr 2006. doi:10.1261/rna.2343206.

[18] Alex Bateman, Shipra Agrawal, Ewan Birney, Elspeth A Bruford, Janusz M Bujnicki, Guy Cochrane, James R Cole, Marcel E Dinger, Anton J Enright, Paul P Gardner, Daniel Gautheret, Sam Griffiths-Jones, Jen Harrow, Javier Herrero, Ian H Holmes, Hsien-Da Huang, Krystyna A Kelly, Paul Kersey, Ana Kozomara, Todd M Lowe, Manja Marz, Simon Moxon, Kim D Pruitt, Tore Samuelsson, Peter F Stadler, Albert J Vilella, Jan-Hinnerk Vogel, Kelly P Williams, Mathew W Wright, and Christian Zwieb. RNAcentral: A vision for an international database of RNA sequences. *RNA*, 17(11):1941–1946, Nov 2011. doi:10.1261/rna.2750811.

[19] I. L. Hofacker, W. Fontana, P. F. Stadler, L. S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie / Chemical Monthly*, 125(2):167–188, 1994. doi:10.1007/BF00818163.

[20] A. Waugh, P. Gendron, R. Altman, J. W. Brown, D. Case, D. Gautheret, S. C. Harvey, N. Leontis, J. Westbrook, E. Westhof, M. Zuker, and F. Major. RNAML: a standard syntax for exchanging RNA information. *RNA*, 8(6):707–717, 2002. doi:10.1017/S1355838202028017.

[21] M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequencesusing thermodynamics and auxiliary information. *Nucleic Acids Res.*, 9:133–148, 1981. doi:10.1093/nar/9.1.133.

[22] H. M. Berman, W. K. Olson, D. L. Beveridge, J. Westbrook, A. Gelbin, T. Demeny, S. H. Hsieh, A. R. Srinivasan, and B. Schneider. The nucleic acid database. A comprehensive relational database of three-dimensional structures of nucleic acids. *Biophys J*, 63(3):751–759, 1992. doi:10.1016/S0006-3495(92)81649-1.

[23] Jean-François Lemay and Daniel A Lafontaine. Core requirements of the adenine riboswitch aptamer for ligand binding. *RNA*, 13(3):339–350, Mar 2007. doi:10.1261/rna.142007.

[24] J.J. Cannone, S. Subramanian, M.N. Schnare, J.R. Collett, L.M. D'Souza, Y. Du, B. Feng, N. Lin, L.V. Madabusi, K.M. Muller, N. Pande, Z. Shang, N. Yu, and R.R. Gutell. The Comparative RNA Web (CRW) Site: An Online Database of Comparative Sequence and Structure Information for Ribosomal, Intron, and other RNAs. *BMC Bioinformatics*, 3(2), 2002. doi:10.1186/1471-2105-3-15.

[25] R. E. Bruccoleri and G. Heinrich. An improved algorithm for nucleic acid secondary structure display. *Comp Appl Biosci*, 4(1):167–173, 1988. doi:10.1093/bioinformatics/4.1.167.

[26] Emden R. Gansner, Eleftherios Koutsofios, Stephen C. North, and Kiem phong Vo. A Technique for Drawing Directed Graphs. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 19(3):214–230, 1993. doi:10.1109/32.221135.

[27] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE*, 30(11):1203–1233, 2000. doi:10.1002/1097-024X(200009)30:11<1203::AID-SPE338>3.3.CO;2-E.

[28] D.H. Mathews, J. Sabina, M. Zuker, and D.H. Turner. Expanded sequence dependence of thermodynamic parameters improvesprediction of RNA secondary structure. *J. Mol. Biol.*, 288:911–940, 1999. doi:10.1006/jmbi.1999.2700.

[29] Zhili Xu and Gloria M Culver. Chemical probing of RNA and RNA/protein complexes. *Meth Enzymol*, 468:147–65, Jan 2009. doi:10.1016/S0076-6879(09)68008-3.

[30] Jörg C Schlatterer and Michael Brenowitz. Complementing global measures of RNA folding with local reports of backbone solvent accessibility by time resolved hydroxyl radical footprinting. *Methods*, 49(2):142–7, Oct 2009. doi:10.1016/j.ymeth.2009.04.019.

[31] Dominique Fourmy and Satoko Yoshizawa. Protein-RNA footprinting: an evolving tool. *Wiley interdisciplinary reviews RNA*, 3(4):557–66, Jan 2012. doi:10.1002/wrna.1119.

[32] Stefanie A Mortimer, Cole Trapnell, Sharon Aviran, Lior Pachter, and Julius B Lucks. SHAPE-Seq: High-Throughput RNA Structure Analysis. *Curr Protoc Chem Biol*, 4(4):275–97, Dec 2012. doi:10.1002/9780470559277.ch120019.

[33] Chaolin Zhang and Robert B Darnell. Mapping in vivo protein-RNA interactions at single-nucleotide resolution from HITS-CLIP data. *Nat Biotechnol*, 29(7):607–14, Jul 2011. doi:10.1038/nbt.1873.

[34] Zhengqing Ouyang, Michael P Snyder, and Howard Y Chang. SeqFold: Genome-scale reconstruction of RNA secondary structure integrating high-throughput sequencing data. *Genome Res*, Oct 2012. doi:10.1101/gr.138545.112.

[35] Kourosh Zarringhalam, Michelle M Meyer, Ivan Dotu, Jeffrey H Chuang, and Peter Clote. Integrating chemical footprinting data into RNA secondary structure prediction. *PLoS ONE*, 7(10):e45160, Jan 2012. doi:10.1371/journal.pone.0045160.

[36] Christopher W Leonard, Christine E Hajdin, Fethullah Karabiber, David H Mathews, Oleg V Favorov, Nikolay V Dokholyan, and Kevin M Weeks. Principles for understanding the accuracy of SHAPE-directed RNA structure modeling. *Biochemistry*, 52(4):588–95, Jan 2013. doi:10.1021/bi300755u.

[37] J.-B. Fourmann, A.-S. Tillault, M. Blaud, F. Leclerc, C. Branlant, and B. Charpentier. Comparative study of RNA conformational dynamics during assembly of two box H/ACA ribonucleoprotein pseudouridine-synthases revealing uncorrelated efficiencies in assembly and activity. *PLoS ONE*, 8(7):e70313, 2013. doi:10.1371/journal.pone.0070313.