LocARNA 2.0: versatile simultaneous alignment and folding of RNAs

Sebastian Will

LocARNA 2.0: versatile simultaneous alignment and folding of RNAs

Sebastian Will

Abstract Generating accurate alignments of non-coding RNA sequences is indispensable in the quest for understanding RNA function. Nevertheless, aligning RNAs remains a challenging computational task. In the twilight-zone of RNA sequences with low sequence similarity, sequence homologies and compatible, favorable (a priori unknown) structures can be inferred only in dependency of each other. Thus, simultaneous alignment and folding (SA&F) remains the gold-standard of comparative RNA analysis, even if this method is computationally highly demanding. This text introduces to the recent release 2.0 of the software package LocARNA, focusing on its practical application. The package enables versatile, fast and accurate analysis of multiple RNAs. For this purpose, it implements SA&F algorithms in a specific, lightweight flavor that makes them routinely applicable in large scale. Its high performance is achieved by combining ensemble-based sparsification of the structure space and banding strategies. Probabilistic banding strongly improves the performance of LocARNA 2.0 even over previous releases, while simplifying its effective use. Enabling flexible application to various use cases, LocARNA provides tools to globally and locally compare, cluster and multiply align RNAs based on optimization and probabilistic variants of SA&F, which optionally integrate prior knowledge, expressible by anchor and structure constraints.

Key words: Simultaneous alignment and folding, RNA alignment, RNA secondary structure prediction, multiple sequence alignment, comparative analysis

1 Introduction

RNA molecules are known as multifaceted players in biological systems, which perform a variety of specific functions through their complex and individual spatial structures. A key technology for unraveling the biological function of RNA molecules is their computational comparison. Such methods are e.g. used to identify of functionally similar RNAs based on evolutionary conserved sequence and structure, find conserved sequence and structure patterns, characterize RNA families, and predict reliable structure models.

Sebastian Will (ORCID: 0000-0002-2376-9205)
LIX, École Polytechnique, Institut Polytechnique de Paris, Palaiseau, France, e-mail: sebastian.will@polytechnique.edu

Since structure can be conserved for RNAs of even quite remote sequences, comparing RNAs cannot solely rely on standard sequence alignment methods but requires to take structure information into account. This poses specific challenges, in the typical case where the RNAs' structures are a priori unknown.

Focussing on this case of unaligned RNAs with unknown structure, three main strategies have been identified to infer alignments and likely structure models [5]. Crucially, such methods often rely on RNA secondary structure prediction (in the sense of free energy minimization [26] and tools like RNAfold of the Vienna RNA package [11]) to complement the structural information. The first strategy, aligns the RNA sequences solely relying on 'one-dimensional' sequence information; RNA-specific structure prediction based methods are then applied to infer a common two-dimensional secondary structure from the alignment, typically called, the *consensus structure*. Such inference is efficiently performed by tools like RNAalifold [1] in an extension of standard RNA energy minimization. The second, reverse strategy starts by predicting structure models of the single RNAs (e.g. using efficient energy minimization as implemented in RNAfold); then, it aligns the structure-annotated RNAs by specialized tree alignment or tree editing-like algorithms. This scheme is directly implemented by tools like RNAforester [10] and MARNA [18].

Both strategies are limited in theory and practice by their separation of alignment and structure prediction. In the first strategy, the alignment has to be constructed without structural information; this works well only for highly similar RNAs, but was demonstrated to plainly fail for RNAs below 60% sequence similarity [6]. The second strategy requires reliably predicted structures, whereas the accuracy of prediction from single sequences is limited. In principle, this cyclic dependence of the alignment and structure prediction phase is best resolved by optimizing the structure model and the alignment at the same time, i.e. by simultaneous alignment and folding (SA&F). Already in 1985, Sankoff presented an efficient solution [17] to this problem, however the time and space complexity of this original SA&F algorithm prohibit its direct practical application.

Several later implementations of the Sankoff algorithm, made moderate applications of SA&F feasible due to strong heuristic restrictions of the search space [7, 8]. Only much later developments paved the way for large-scale, routine application of the SA&F-paradigm for RNA structure analysis. Notably, PMcomp [9] introduced the idea of light-weight SA&F that strongly reduces the computational overhead of SA&F (at first, by a constant factor), which was then picked up by a whole family of SA&F methods, e.g. [23, 20, 3, 22].

1.1 LocARNA 2.0

Here, we provide an introduction to the practical application of LocARNA; a software package that implements a series of RNA alignment algorithms following the general line of light-weight SA&F. By applying additional strategies that overcome the efficiency problems of SA&F, LocARNA combines the potential for very fast RNA comparisons with the advantages of SA&F. Moreover, it implements variants and extensions that support flexible use in routine as well as many non-standard scenarios.

Specifically, the text introduces for the first time into specific characteristics and features of the latest release LocARNA 2.0. Among new features, in-detail improvements and under-the-hood changes, the single most notable benefit of this release comes from a newly implemented sequence similarity-based probabilistic banding strategy, which transparently speeds up all alignment variants. This technique neither requires alignment-mode specific parameter settings nor is likely to com-

promise the alignment accuracy, since it automatically adapts to the alignment mode and relevant alignment space; thereby it significantly improves the performance and usability of LocARNA.

1.2 Scope and overview of the text

The text discusses the following topics concerning the practical use of the software in more depth:

- installation and setup of the software
- general usage for pairwise and multiple alignment
- the general mechanism of pairwise SA&F by LocARNA
- the construction of multiple alignments
- implemented heuristics that balance speed and accuracy
- structure constraints, anchor constraints, and realignment

This detailed discussion is necessarily limited to a selection of the possibilities available due to the software package. Nevertheless it covers its elementary as well as advanced usage in specialized scenarios. We are going to touch theoretical background of the software in terms of computational models, heuristics, and combinatorial optimization algorithms where this is directly beneficial for the successful, advanced use of the software. For further technical details, the reader is referred to the corresponding original research publications [14, 15, 16, 21, 22, 23, 24].

1.3 Noteworthy omissions from this text

Many advanced or specialized features of the LocARNA package had to be omitted from this introductory book chapter. Nevertheless, to provide at least a more comprehensive overview, the most important ones shall be mentioned in a cursory summary for further reference:

Local and semi-global alignment.

While the chapter focuses on global alignment of entire input sequences, LocARNA supported various modes of local alignment from its beginnings (this circumstance causing the acronym). Such, the package provides local and semi-local alignment, where it either finds the best alignment of subsequences or allows free end gaps to accommodate scenarios like finding the occurrence of a small sequence in a larger one, loosely defined 3'-ends, etc. Local alignment even comes with a special flavor that can leave out dissimilar substructure (*structure local alignment*; [16]).

Clustering of RNAs.

One of the earliest applications of LocARNA was the clustering of non-coding RNAs by their (local) sequence and structure similarity. Clustering moderately-sized sets of RNAs (several hundreds to thousands) using SA&F is sufficiently fast and directly possible using a special mode of the multiple alignment tool `mlocarna`. For this purpose, it is recommended to use multi-threading (`mlocarna` option `--threads`) and limit the maximum alignment size (option `--max-alignment-size`). For

this use case, `mlocarna` computes a distance-annotated tree of the input RNAs and intermediary 'cluster' alignments up to the given maximum size.

Probabilistic alignment, alignment reliabilities, and consistency-based multiple alignment.

LocARNA implements the computation of partition functions over SA&F, which is used to derive match probabilities and column-wise reliability score [21]. Match probabilities are moreover the basis for improvements to the construction of multiple alignments. This is achieved by computing probabilistic consistency-based multiple alignment (similar to ProbCons [4], which was introduced for pure sequence alignment). Moreover, probabilistic alignment is used in LocARNA to score improvements due to (experimental) iterative refinement of multiple alignment.

Exact matches of local sequence-structure patterns.

The package implements the method ExpaRNA-P [15], which finds exact matches in RNA structure ensembles. These can be useful to understand similarities of large RNA molecules (without *a priori* known structure). We suggested a method to use such matches as anchor speed up alignments of large structural RNAs.

Improved pairwise alignment methods.

The package supports to exchange the pairwise alignment method in its construction of multiple alignments. It implements the pairwise SA&F algorithm SPARSE [22], which extends the alignment model of PMcomp (by loop insertions and deletions comparable to the original Sankoff algorithm) and speeds up pairwise RNA alignment using strong structure-ensemble based sparsification. Other supported pairwise alignment methods are CARNA [2] and Pankov [14].

2 Material: Installing LocARNA

It is recommended to install the software on Linux or MacOS using the package manager Conda as described below. Windows users can install Ubuntu with Windows Subsystem for Linux (WSL) and then follow the installation instructions for Linux as below from the Ubuntu terminal.

As alternative to the Conda installation, the software can be compiled and installed from a source tar-ball or directly from its repository on GitHub https://github/s-will/locarna. While compiling from source should be limited to special cases, instructions for the installation from source are provided with the LocARNA online documentation at https://s-will.github.io/LocARNA/. Recall that this chapter describes the release 2.0 of LocARNA.

2.1 Installation via Conda

It's recommended to install Conda from https://conda.io/en/latest/miniconda.html following the given instructions for your system. LocARNA is best installed in a dedicated Conda environment. First, this environment has to be created (once) and activated (once per session) by running

```
conda create -n locarna
conda activate locarna
```

in a terminal. Then, LocARNA is installed in this environment by

```
conda install -c conda-forge -c bioconda locarna
```

As a main benefit of using the package manager Conda, this single command installs the most recent LocARNA package without requiring compilation and together with its dependencies; i.e first of all, the Vienna RNA package [11] (Conda package `viennarna`). The option `-c bioconda` tells Conda to find the package on the Bioconda channel.

After successful installation, the following call should report the LocARNA version (at least 2.0).

```
locarna --version
```

3 Methods

3.1 Elementary usage

The central functionality of the package is the computation of (multiple) alignments and consensus structures via the tool `mlocarna`, starting from a set of RNA sequences. As a running example, we consider 5 tRNAs. We assume that their sequences are specified in a fasta file:

File tRNA_5.fa:

```
>X03715.1_288-361
GCGCCCAUAGAUCAAUUGGAUAGAUCGUUUGACUACGGAUCAAAAGGUUGAGGGUUCGAUUCCUUCUGGGCGCG
>J01435.1_264-194
UAGAUUGAAGCCAGUAAGUAGGGUAUUUAGUUGUUAACUAAAUUUUCGUAGGUUUGAAUCCUUCCAAUCUA
>X16886.1_781-711
AGUAAAGUAAGCUAAUAAAAAGCUUUUGGGUUCAUACCUCAAAAAUGGAAGGAUAAAUACCUCCCUUUAUU
>V00654.1_12038-12108
ACUUUUAAAGGAUAGUAGUUUAUCCGUUGGUCUUAGGAACCAAAAAAUUGGUGCAACUCCAAAUAAAAGUA
>AE004237.1_2976-2903
GCGUCCGUAGCUCAGUUGGUUAGAGCACCACCUUGACAUGGUGGGGGUCGGUGGUUCGAGUCCACUCGGACGCA
```

To run the examples, first create (or obtain) this file. Running `mlocarna` (again, from the command line) without any further parameters like

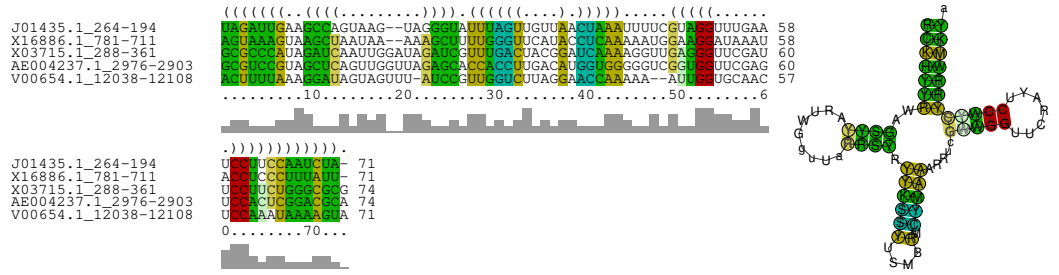

Fig. 1 The alignment and consensus structure for the running example of 5 tRNAs. These figures are generated using RNAalifold from the alignment computed by `mlocarna` with default parameters. The color annotation encodes evolutionary support of base pairs in the specific way of RNAalifold; the bars below of the alignment show column-wise sequence conservation; finally, RNAalifold reports "most informative" IUPAC code for each column in the 2D structure plot (RNAalifold option `--mis`).

```
mlocarna tRNA_5.fa
```

computes a multiple alignment that aims to explain both the sequence as well as the structure homology of the sequences. The command prints the alignment and the inferred common structure as text output to the screen.[1] (At the same time, `mlocarna` writes several result and intermediary files to a subdirectory `tRNA_5.out`.)

```
J01435.1_264-194     UAGAUUGAAGCCAGUAAG--UAGGGUAUUUAGUUGUUAACUAAAUUUUCGUAGGUUUGAA
X16886.1_781-711     AGUAAAGUAAGCUAAUAA--AAAGCUUUUGGGUUCAUACCUCAAAAAUGGAAGGAUAAAU
X03715.1_288-361     GCGCCCAUAGAUCAAUUGGAUAGAUCGUUUGACUACGGAUCAAAAGGUUGAGGGUUCGAU
AE004237.1_2976-2903 GCGUCCGUAGCUCAGUUGGUUAGAGCACCACCUUGACAUGGUGGGGGUCGGUGGUUCGAG
V00654.1_12038-12108 ACUUUUAAAGGAUAGUAGUUU-AUCCGUUGGUCUUAGGAACCAAAAA--AUUGGUGCAAC

J01435.1_264-194     UCCUUCCAAUCUA-
X16886.1_781-711     ACCUCCCUUUAUU-
X03715.1_288-361     UCCUUCUGGGCGCG
AE004237.1_2976-2903 UCCACUCGGACGCA
V00654.1_12038-12108 UCCAAAUAAAAGUA

alifold              (((((((..((((.........)))).(((((((....).)))))).....(((((.....
                     .)))))))))))). (-39.72 = -16.98 + -22.74)
```

The last line of the output reports the consensus structure of the alignment; this structure is computed—based on LocARNA's alignment—by the tool RNAalifold (with command line arguments `--mis --cfactor=0.6 --nfactor=0.5`). The output line includes the total energy calculated by this tool, as well as its composition from the average Turner energy and conservation score. RNAalifold as well produces graphical output of the alignment and consensus structure as shown in Figure 1. The RNAalifold-specific color annotation of base-paired columns and base pairs in these figures reflects the diversity of consensus base pairs - related to the idea of compensatory mutations. (Please find more details of the graphical representation in RNAalifold's documentation.)

[1] For this presentation, we added the formatting option `--width 60`, which breaks the sequence lines after 60 characters, instead of 120 per default. In consequence, this alignment is printed in two blocks.

Input and output files of `mlocarna`.

The tool writes it results to a target directory, whose name is (by default; otherwise use `--tgtdir`) derived from the input file; e.g. for input file `Examples/tRNA.fa`, the target directory would be `Examples/tRNA.out`. The fasta input file is preserved in the target directory as `input/input.fa`. Results are written to the subdirectory results; e.g. `mlocarna` writes the result alignment to `results/result.aln` in the format of ClustalW. Given the option `--stockholm`, `mlocarna` additionally writes the result alignment and it's consensus structure in Stockholm format to `results/result.stk`. Along with these files, `mlocarna` writes several other files, containing final and intermediary results, to the target directory. For example, it writes figures as the ones of Figure 1 of the result alignment and consensus structure as produced by RNAalifold in PostScript format to subdirectory `results`. As an alternative to fasta input, `mlocarna` can read its input sequences from a file in ClustalW format. This requires the option `--realign`, since it serves to realign already aligned sequences. By default, `mlocarna` ignores the gaps in the input alignment.

Getting specific help for `mlocarna`.

The command line tool `mlocarna` can be finely controlled by a rich set of command line parameters. Note that only some of them will be immediately obvious and are therefore easily applicable in standard applications of the tool. Others will become clear only in the course of this text and/or after more experience with the tool. A reference of the command line parameters is provided with the manual page of `mlocarna`. It is found online at https://s-will.github.io/LocARNA/md_src_Utils_mlocarna.html and can be displayed from the command line using `mlocarna`'s option `--man` or by the command

```
man mlocarna
```

3.2 Remarks on the nature of default results and advanced usage of LocARNA

For users of the tool (even occasional ones), it is helpful to understand the nature of LocARNA's main output in more detail. In its default mode, given unaligned input sequences without *a priori* known structure, the tool predicts a *good* alignment of the *entire* input sequences (*global* alignment). In the case of RNAs, a good alignment must not only align similar nucleotides with little insertions and deletions, but as well possess a consensus structure that is thermodynamically favorable for each of the single RNAs. LocARNA therefore optimizes a combined evaluation of the alignment and the predicted consensus structure.

On the one hand, thanks to this ability of considering sequence similarity and quality of predicted structure at the same time, which is the core idea of simultaneous alignment and folding, LocARNA can produce good alignments of RNAs in the twilight or even midnight zone, where strong sequence similarity signals are lacking and thus sequence-only alignment methods fail. This is as well the case of our running example from the deep midnight zone (with only 40% average pairwise sequence identity.)

On the other hand, alignment of RNAs with low sequence identity poses specific problems that are not all known from simpler sequence alignment tools. Consequently, LocARNA could struggle to

find a good alignment for various reasons. While some issues are fundamental and subject of ongoing research, many of these problems can be dealt with by specific parametrization of the tool (e.g. relative focus on sequence or structure similarity; gap cost), specifying a priori knowledge in the form of constraints (anchor constraints, specifying aligned positions and/or structure constraints, providing knowledge on the structure space), and specific alignment modes (e.g. local and semi-local alignment; but as well, probabilistic alignment to learn about local alignment reliability). An important aspect is the high computational complexity of the method, which requires to make trade-offs (which in LocARNA are interpretable and controlled) between accuracy and execution speed.

In summary, while LocARNA responds to these challenges of RNA alignment by offering a high degree of flexibility (different alignment modes, different SA&F variants, various heuristics, parametrization, constraints, ...), using these possibilities effectively, requires some insight into its computational machinery (as discussed in the next subsection).

3.3 The construction of multiple and pairwise alignments by LocARNA

The tool `mlocarna` constructs multiple alignments by the progressive method as it is well-known from classic alignment tools like ClustalW [19].

Note that the LocARNA package implements several extensions and refinement of the progressive strategy, like probabilistic consistency-transformations and iterative refinement, which are available with specific alignment modes. Moreover, alignment construction based on T-Coffee is available via the tool `locarnate` of the package. To limit the scope of this text, we discuss only the purely progressive method, the default strategy of `mlocarna`, in more detail.

The main idea of the progressive method is to construct multiple alignments based on the repeated computation of pairwise alignments. This has the main advantage that in the case of sequence alignment and as well of simultaneous alignment and folding, the pairwise problem is precisely defined and can be solved efficiently. In contrast, the multiple alignment problem is computationally hard (NP hard); therefore progressive construction does not guarantee to find the very best multiple alignment.

3.3.1 Pairwise alignment by the tool `locarna`

In its default setting, `mlocarna` employs the low level tool `locarna` for computing the pairwise alignments. Applying a dynamic programming algorithm [23], this tool optimizes over pairwise alignments and their simultaneously foldings into a common RNA secondary structure. First, we consider this tool largely as a black box, but explain some more details below. It follows the general idea of PMcomp [9] to derive the optimal alignment and common structure from the two RNA sequences and the base pair probabilities of the both single sequences. The latter are predicted based on the thermodynamic model by the McCaskill algorithm [13] (as implemented in the Vienna RNA package). Together with the sequences we consider the base pair probabilities input of the pairwise alignment tool.

Figure 2 visualizes the base pair probabilities of each of two example sequences as heatmaps (equivalent to the dot plot representation of the Vienna RNA package). The figure also shows that the probable base pairs are typically very sparse and highlights the more probable base pairs. The LocARNA algorithm was designed to exploit this sparsity, which in LocARNA fundamentally speeds up computations and reduces the memory footprint over traditional SA&F. The effect is controlled

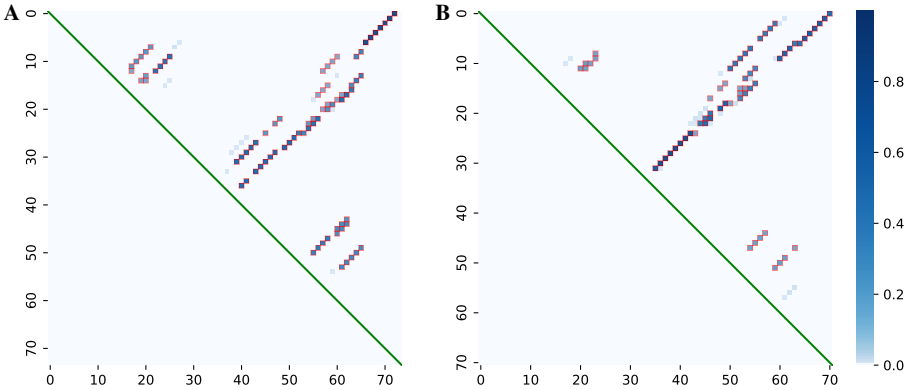

Fig. 2 Base pair probabilities of **A)** tRNAs AE004237.1_2976-2903 **B)** J01435.1_264-194. The heatmap colors show the probabilities of base pairs (i, j). These probabilities from single sequence folding allow effectively restricting the folding space in simultaneous alignment and folding. The red boxes mark base pairs with probabilities greater equal 10^{-2}. For this threshold, LocARNA would consider only the highlighted base pairs. This strategy significantly speeds up LocARNA while preserving accuracy.

by a probability threshold (argument `-p` of the tools `mlocarna` and `locarna`; the threshold for the minimum base pair probability defaults to 10^{-3}).

Banding strategies.

In addition to controlling the structure space using a minimum base pair probability, LocARNA controls the alignment space. A common simple strategy to speed up pairwise alignment algorithm is to restrict the maximum difference between alignable positions i (of the first sequence) and j (of the second sequence). In typical dynamic programming alignment algorithms, one would technically limit this difference for all matrix entries (i, j). In consequence, such *banding methods* compute only bands (around the diagonal) of the DP matrices. Thus, already before version 2.0, LocARNA implemented two "ad-hoc" banding strategies that are illustrated in Figure 3. The first one, which is more generally applicable, restricts the computed matrix entries around the matrix diagonal (option `--max-diff`). For the case of realignment, a technique was developed [24] to restrict the alignment space relative to existing alignments (option `--max-diff-aln`). This allows the fast realignment of initial alignments, whether these come from a database or are computed by less costly alignment methods. In this setting, the technique overcomes problems with large insertions and deletions.

Probabilistic banding in LocARNA 2.0.

Recent LocARNA replaces the default banding strategy to allow more flexible and stronger banding that automatically adapts to the lengths and sequence similarity of the input sequences. This allows LocARNA 2.0 to run (for specific applications, even much) faster with default parameters than its predecessors. It builds on the computation of trace probabilities in sequence alignments of the input sequences. After computing the probability for each matrix cell (i, j) that it occurs on the trace of the alignment, this technique applies a threshold probability to decide about the matrix cells computed

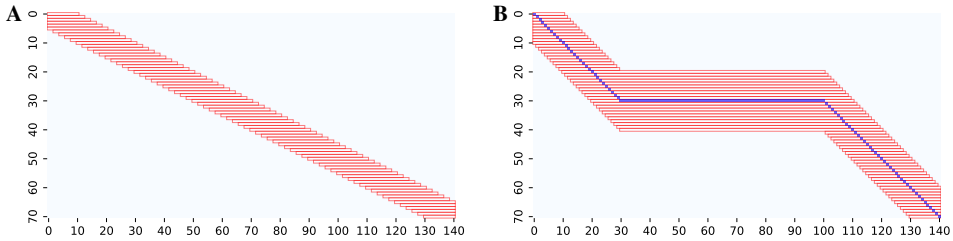

Fig. 3 Banding by a maximal position difference $\Delta = 10$. The (due to the banding strategy) computed matrix cells are highlighted in red. **A)** in the case of unequal sequence lengths, here n=70 and m=140, the length difference can be distributed by $|i - j \cdot n/m| \leq \Delta$. Before version 2.0, LocARNA mainly used this banding method (for global alignment), which is still available with argument `--max-diff` Δ. **B)** This approach still struggles with unequally distributed insertions. In an attempt to overcome this in the case of realignment, given an initial alignment (in the example with one large insertion; trace in blue), LocARNA can restrict the difference to this trace (with argument `---max-diff-aln` Δ). In practice, this allows LocARNA to quickly realign the given initial alignment (which could e.g. be computed by less accurate methods). Going beyond these ad-hoc or very specialized strategies, LocARNA 2.0 implements a probabilistic banding technique that can be universally applied and thus allows easier and more flexible use of strong banding (see Figure 4).

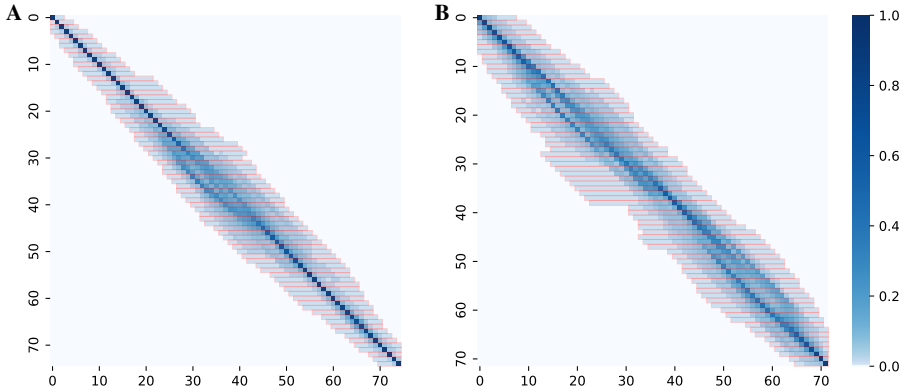

Fig. 4 Trace probabilities from pairwise sequence alignments of **A)** tRNAs AE004237.1_2976-2903 with X03715.1_288-361i and **B)** J01435.1_264-194 with V00654.1_12038-12108. The color indicates the probability that an alignment trace passes each matrix cell (i, j). Such probabilities are computed from forward/backward partition functions. These sequence-alignment inferred trace probabilities allow effectively restricting the alignment space in simultaneous alignment and folding. This banding method significantly speeds up LocARNA 2.0 while preserving accuracy. For trace probability threshold 10^{-5}, LocARNA's simultaneous and folding algorithm would consider only the entries in red boxes.

by the computationally more costly SA&F algorithm. Figure 4 illustrates the effect for two pairs of example sequences.

Probabilistic banding was similarly used before by SA&F tools like RAF [3] and Dynalign [7]. The strategy is applicable because trace probabilities from sequence alignments can be computed quickly by a forward / backward partition function algorithm (modeling alignment probabilities in a statistical mechanics approach; i.e. assuming Boltzmann distribution of the alignments based on their sequence alignment scores). This technique works typically well because high-scoring alignments from SA&F are unlikely to be strongly improbable in the sequence alignment model.

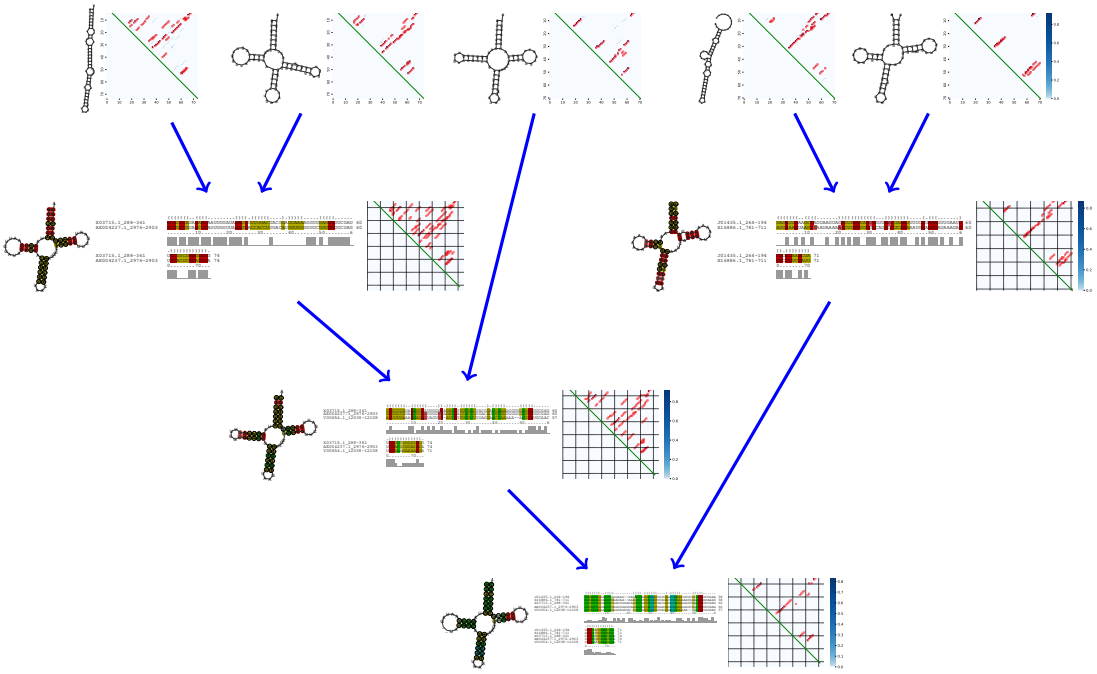

Fig. 5 Progressive alignment of the five tRNAs from the running example. The figure shows base pair probability heatmaps ("dot plots") and minimum free energy structures of the input structures in the first row. The blue arrows represent the guide tree and thus show the order of progressive alignment steps that combine two sequences, a sequence and an alignment or two alignments into a larger alignment. For each intermediate or final alignment, we show the RNAalifold consensus structure, the alignment figure drawn by RNAalifold and the 'averaged' base pair probabilities as heat map. In these heat maps, the base pairs with probability $\geq 10^{-3}$, which are (or, for the final alignment, would be) considered by `locarna` in the next progressive step, are highlighted.

3.3.2 Progressive alignment by `mlocarna`

Figure 5 depicts the progressive alignment for the running example. As pre-computation, `mlocarna` calculates the base pair probability matrices of all input sequences. For each input RNA, it stores the sequence and pair probabilities (above the threshold), in a file in the `input` subdirectory. These files are written in LocARNA's pp format and can serve as direct input to the tool `locarna`. The first row of Figure 5 represents the state after the pre-computation. For illustration, we also show the predicted secondary structure (minimum energy structure) of each sequence.

The progressive scheme starts by constructing a phylogenetic tree ("guide tree") from all pairwise distances between the input sequences. These distances are obtained by computing (optimal) pairwise alignments of all quadratically many sequence pairs.

The guide tree (blue arrows in the figure) dictates the order of constructing the multiple alignment from the input sequences. The progressive alignment therefore first aligns the closest two sequences; then (in the example) the next two closest sequences. This procedure then comes to the point where a sequence has to be combined with an alignment - or more generally, where given two alignments, we want to derive a larger alignment comprising all their sequences. Following the general idea of progressive alignment, LocARNA derives this larger alignment by aligning the two alignments. Here,

one can essentially apply the same pairwise SA&F algorithm as in the case of two input sequences. However, in extension, the sequence similarities are computed from the input alignment columns and the structure prediction is based on (in principle) average base pair probabilities—technical details are found in [23].

These progressive alignment steps are repeated in the order of the tree until an alignment of all sequences is derived. In the course, of this scheme, intermediary alignments (and their averaged base pair probabilities) are derived. Especially when aligning larger sets of RNAs (or even more in clustering applications), this information can become valuable. Therefore, `mlocarna` stores this data in subdirectory `intermediates`—as ClustalW aln files and as files in pp-format, which contain the alignment and its (consensus) base pair probabilities.

3.4 Structure and anchor constraints

An important feature of LocARNA's alignment tools is the support of constraints that allow users to guide the alignment due to prior knowledge. There are two major types of prior knowledge that can be encoded as constraints.

- *First,* information about the structure of input RNAs. Here, one can provide structure constraints strings (for each input RNA) in the syntax of the Vienna RNA package (compare option `-C, --constraint` of RNAfold). These constraints are then directly applied in the prediction of the input sequences' base pair probabilities. In this way, they limit the considered structure space in the SA&F algorithm.
- *Second,* prior knowledge about homologous positions. LocARNA can be forced to align specific positions of the input sequences by specifying anchors (thus, imposing *anchor constraints*). It is noteworthy that anchor constraints could be inconsistent to each other or conflict with other heuristics. In such cases LocARNA will not produce an alignment.

3.4.1 Specification of constraints

For their use in `mlocarna`, both types of constraints can be specified in extension of the standard fasta format via 'constraint lines'. Here is an example input for `mlocarna` of two sequences together with structure and anchor constraint annotation in the extended fasta-like format.

File example-w-constraints.fa:

```
>A
GACCCUGGGAACAUUAACUACUCUCGUUGGUGAUAAGGAACA
..((.(....xxxxxx..................))).xxx #S
..........000000......................111 #1
..........123456......................123 #2
>B
ACGGAGGGAAAGCAAGCCUUCUGCGACA
.(((....xxxxxx.......))).xxx #S
........000000...........111 #1
........123456...........123 #2
```

In this example, the two structure constraints (lines annotated #S), specify that certain sequence positions must be unpaired (`x`) and others must not be crossed by base pairs (corresponding brackets; the exact syntax and semantics of these constraints strings is defined by RNAalifold.) For each sequence, we specify named anchors, where the names should be read top-down over the anchor lines #i. That is, the file defines names `01,02,...,06, 11,12,13` (of length two) for according positions of each sequence. Positions of equal names must be aligned to each other. Moreover, names must be specified in lexicographic order.

A simple mlocarna run on this file will by default take the constraints into account and produce an alignment that is guided by the restricted structure space and aligns all anchors:

```
A         GACCCUGGGAACAUUAACUACUCUCGUUGGUGAUAAGGAACA
B         A--CGGAGGGAAAGCA---AGCCUUCUGCG---------ACA
#A1       ..........000000.......................111
#A2       ..........123456.......................123

alifold  ..(((((((((...........)))))..)).).......... (-12.85 =  -3.70 +  -9.15)
```

While the alignment clearly satisfies the anchor constraints, note that the alifold structure does not necessarily satisfy the structure constraint annotation (as seen in this case). This happens since the RNAalifold predicts the unconstrained best structure from the alignment; recall that the structure constraints affect only the input structure space.

3.4.2 Anchor specifications as bed format annotation

Anchors can alternatively be specified in bed format. This is useful to annotate larger alignments and or include information from sequence annotation. Using this method, the same anchor constraints as in file `example-w-anchors.fa` (by lines tagged #1, #2) could be specified by

File example-anchors.bed:

```
A   10      16      first_box
B   8       14      first_box
A   39      42      ACA-box
B   25      28      ACA-box
```

As suggested by the example, anchor regions receive arbitrary names and contig (or sequence) names. Again, the idea is to enforce the alignment of equally named regions. This bed file can be used in combination with a fasta input file that defines the sequences and potential structure constraints. For example, to reproduce the above example, we define

File example-wo-anchors.fa:

```
>A
GACCCUGGGAACAUUAACUACUCUCGUUGGUGAUAAGGAACA
..((.(....xxxxxx..................))).xxx #S
>B
ACGGAGGGAAAGCAAGCCUUCUGCGACA
.(((....xxxxxx.......))).xxx #S
```

and then call mlocarna as follows:

```
mlocarna --anchor-constraints example-anchors.bed  example-wo-anchors.fa
```

3.4.3 Realignment

Another way to specify prior knowledge on the sequence alignment space is provided by LocARNA's realignment mode. In this mode (option `--realign`), `mlocarna` is called with input in the form of an alignment in ClustalW format, e.g.

```
mlocarna --realign example-realign.aln
```

Then, `mlocarna` uses the unaligned sequences of the alignment as input (i.e. it removes all gap symbols). The real benefit of this input mode is that it allows defining anchor constraints as 'consensus constraints' in the input, e.g.

File example-realign.aln:

```
CLUSTAL W

vhuU                AGCUCACAACCGAACCCAUU-------------UGGGAGGUUGUGAGCU
fdhA                CGCCACCCUGCGAACCCAAUAUAAAAUAAUACAAGGGAGCAG-GUGGCG
fruA                --CCUCGAGGGGAACCCGAA-------------AGGGACCCGAGAGG--
#A1                 ..*...........CCC.............................5..
#S                  ((((((.((((...(((.................))).)))).))))))
```

Note that the 'names' in the anchor annotation string (line #A1) are arbitrary. All symbols apart from blank and dot are interpreted as defining an anchor for the respective position of each sequence. On this input `mlocarna` then produces an alignment that preserves the matches in the marked alignment columns. For example, running

```
mlocarna --realign example-realign.aln
```

yields the alignment

```
vhuU                AGCUCACAACCGAACCCAUU-------------UGGGAGGUUGUGAGCU
fdhA                CGCCACCCUGCGAACCCAAUAUAAAAUAAUACAAGGGAGCAG-GUGGCG
fruA                --CCUCGAGGGGAACCCGA-------------AAGGGACCCGAGAGG--
#A1                 ..0...........123.............................4..
```

Banding in relation to an alignment.

Note that realignment moreover gives access to a special type of banding heuristic that restricts the alignment space in relation to a given alignment. The central idea is illustrated (for pairwise alignment) in Figure 3B but is available, generalized to multiple alignment, in `mlocarna`.

For the purpose of limited (and therefore fast) realignment of an existing alignment, we specify a maximal distance to the alignment. To realign our example alignment in distance 1 (in addition to respecting the constraint annotation), one uses the following syntax:

```
mlocarna --realign example-realign.aln --max-diff 1 --max-diff-aln .
```

For this simple example, we obtain exactly the same alignment as above, which changed the original alignment only slightly (only in distance one). Nevertheless, limiting the realignment distance guarantees limited modification and speeds up the alignment, which becomes important to handle a large amount of larger alignments (for example, whole genome realignment in REAPR [24]).

4 Notes

4.1 Benchmarking LocARNA's performance—Comparison to the previous release

A benchmark on Bralibase 2.1 provides a general impression of the performance of LocARNA 2.0, in absolute terms as well as compared to the latest previous release 1.9.2.3. For this purpose, without aiming at a comprehensive benchmark, we consider only the sets k2 and k10 for pairwise and 10-way alignments, respectively. The benchmark scripts are provided with the source of the LocARNA package and provide experimental support (through a make file Makefile-benchmark) to reproduce the results, benchmark LocARNA on different benchmark sets, or measure performance with different options.

The Bralibase 2.1 benchmark [25], suggested to compute the two alignment statistics SPS and SCI. The SPS is a sum-of-pairs similarity measure to the reference structure in the benchmark set. The SCI (structure conservation index) is defined as ratio between the average minimum free energy of the single RNAs and the RNAalifold minimum free energy of the alignment. Thus, in this benchmark, it measures the alignment's suitability to guide the prediction of *some* common RNA structure. In addition the structure prediction accuracy is measured by MCC [12] (Matthews correlation coefficient), which measures the similarity of the predicted structure to the Rfam-derived reference structure.

Table 1 Benchmark on k2 and k10 of Bralibase 2.1, comparing LocARNA 2.0 to the previously released version 1.9.2.3. Set k2 consists of 8976 pairwise alignment; k10, of 845 10-way alignments. Time is reported as total user time of running all instances in parallel on Intel(R) Core(TM) i7-10810U CPU; since this machine distributes to 12 cores, the real time is about 1/12 of the reported user time; space, as maximum resident set size of the parallel execution.

	LocARNA 1.9 (`--max-diff 60`)		LocARNA 2.0 (default)	
	k2	k10	k2	k10
SPS	0.86	0.91	**0.89**	0.92
SCI	1.01	0.92	1.01	0.92
MCC	0.81	0.87	**0.82**	0.87
Time (min)	294	1143	**110**	**76**
Space (MB)	49	70	**26**	**30**

Table 1 summarized the results for LocARNA 2.0 and the previous release. Since LocARNA 2.0 only marginally improves these measures on average, users of older versions should not expect generally more accurate alignments from LocARNA 2.0 in standard global alignment. Here, apart from the generally improved usability and higher flexibility, the main advantage is found in computational efficiency. As such, the benchmarks show dramatically improved run-times and space consumption. LocARNA 2.0 computes k2 in about 40% of the time and in 55% of the space; for k10, it takes 0.06% of the time (15-fold speed up) and 44% of the space. To make this comparison realistic for experienced users of previous versions of LocARNA, we used the old version with argument `--max-diff 60`, which activates a moderate ad-hoc banding heuristic. Note that the performance increase over old LocARNA with default parameters is even higher (for k10, roughly, a total factor of 30 in time and 3.5 in space). The time and space improvements in LocARNA 2.0 can be mostly attributed to its probabilistic banding strategy.

References

1. Stephan H. Bernhart, Ivo L. Hofacker, Sebastian Will, Andreas R. Gruber, and Peter F. Stadler. RNAalifold: improved consensus structure prediction for RNA alignments. *BMC Bioinformatics*, 9:474, 2008.
2. Alessandro Dal Palù, Mathias Möhl, and Sebastian Will. A Propagator for Maximum Weight String Alignment with Arbitrary Pairwise Dependencies. In *Principles and Practice of Constraint Programming – CP 2010*, pages 167–175. Springer, Berlin, Germany, 2010.
3. Chuong B. Do, Chuan-Sheng Foo, and Serafim Batzoglou. A max-margin model for efficient simultaneous alignment and folding of RNA sequences. *Bioinformatics*, 24(13):i68–76, 2008.
4. Chuong B. Do, Mahathi S. P. Mahabhashyam, Michael Brudno, and Serafim Batzoglou. ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Research*, 15(2):330–340, February 2005.
5. Paul P. Gardner and Robert Giegerich. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics*, 5:140., September 2004.
6. Paul P. Gardner, Andreas Wilm, and Stefan Washietl. A benchmark of multiple sequence alignment programs upon structural RNAs. *Nucleic Acids Research*, 33(8):2433–9, 2005.
7. Arif Ozgun Harmanci, Gaurav Sharma, and David H. Mathews. Efficient pairwise RNA structure prediction using probabilistic alignment constraints in Dynalign. *BMC Bioinformatics*, 8:130, 2007.
8. Jakob H. Havgaard, Elfar Torarinsson, and Jan Gorodkin. Fast Pairwise Structural RNA Alignments by Pruning of the Dynamical Programming Matrix. *PLOS Computational Biology*, 3(10):e193, October 2007.
9. I. L. Hofacker, S. H. Bernhart, and P. F. Stadler. Alignment of RNA base pairing probability matrices. *Bioinformatics*, 20(14):2222–7, 2004.
10. Matthias Höchsmann. The tree alignment model : algorithms, implementations and applications for the analysis of RNA secondary structures, 2005. [Online; accessed 19. Sep. 2022].
11. Ronny Lorenz, Stephan H. Bernhart, Christian Höner Zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F. Stadler, and Ivo L. Hofacker. ViennaRNA Package 2.0. *Algorithms Mol Biol*, 6:26, 2011.
12. B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochem. Biophys. Acta*, 405:442–451, 1975.
13. J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–19, 1990.
14. Milad Miladi, Martin Raden, Sebastian Will, and Rolf Backofen. Fast and accurate structure probability estimation for simultaneous alignment and folding of RNAs with Markov chains. *Algorithms for Molecular Biology*, 15(1), November 2020.
15. Christina Otto, Mathias Möhl, Steffen Heyne, Mika Amit, Gad M. Landau, Rolf Backofen, and Sebastian Will. ExpaRNA-P: simultaneous exact pattern matching and folding of RNAs. *BMC Bioinformatics*, 15(1):404., December 2014.
16. Wolfgang Otto, Sebastian Will, and Rolf Backofen. *Structure Local Multiple Alignment of RNA*. Gesellschaft für Informatik e. V., 2008.
17. David Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.*, 45(5):810–825, 1985.

18. Sven Siebert and Rolf Backofen. MARNA: multiple alignment and consensus structure prediction of RNAs based on sequence structure comparisons. *Bioinformatics*, 21(16):3352–9, 2005.
19. J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673, November 1994.
20. Elfar Torarinsson, Jakob H. Havgaard, and Jan Gorodkin. Multiple structural alignment and clustering of RNA sequences. *Bioinformatics*, 23(8):926–32, 2007.
21. Sebastian Will, Tejal Joshi, Ivo L. Hofacker, Peter F. Stadler, and Rolf Backofen. LocARNA-P: Accurate boundary prediction and improved detection of structural RNAs. *RNA*, 18(5):900–14, 2012.
22. Sebastian Will, Christina Otto, Milad Miladi, Mathias Möhl, and Rolf Backofen. SPARSE: quadratic time simultaneous alignment and folding of RNAs without sequence-based heuristics. *Bioinformatics*, 31(15):2489–2496, August 2015.
23. Sebastian Will, Kristin Reiche, Ivo L. Hofacker, Peter F. Stadler, and Rolf Backofen. Inferring non-coding RNA families and classes by means of genome-scale structure-based clustering. *PLoS Comput Biol*, 3(4):e65, 2007.
24. Sebastian Will, Michael Yu, and Bonnie Berger. Structure-based whole-genome realignment reveals many novel noncoding RNAs. *Genome Research*, 23(6):1018–1027, June 2013.
25. Andreas Wilm, Indra Mainz, and Gerhard Steger. An enhanced RNA alignment benchmark for sequence alignment programs. *Algorithms for Molecular Biology*, 1(1):1–11, December 2006.
26. Michael Zuker and David Sankoff. RNA secondary structures and their prediction. *Bulletin of mathematical biology*, 46(4):591–621, 1984.