1 Pseudoknot removal by an optimization approach

1.1 Modification of Nussinov-Jacobson algorithm

The original Nussinov-Jacobson algorithm (Nussinov and Jacobson 1980) calculates an RNA secondary structure containing the maximum number of base pairs, where GC and AU base pairs are allowed (optionally GU base pairs and loop-size restrictions can be added). By restricting this algorithm to the base pairs in the knotted structure it calculates a nested structure containing the maximum number of base pairs (Ponty 2006). The algorithm consists of two stages: a fill stage in which the number of base pairs in the optimal solution is calculated, and a traceback stage in which an actual optimal structure is calculated. In case multiple optimal solutions exist, the implementation of the traceback procedure determines which structure will be returned. In pseudocode the algorithm looks like this:

```
FUNCTION fill(seq_len, basepairs):
  # return a square two-dimensional matrix of size seq_len containing
  # the maximum number of base pairs between positions i and j
  # seq_len is integer, specifying the length of the RNA sequence
  # basepairs is a list of base pairs, e.g. [(1,10),(2,9),(4,12)]

  * create a 2D matrix M of size seq_len filled with zeros
  * traverse each position (i,j) in M  (0<=j<seq_len, j-1>=i>=0)
  # i is the upstream and j the downstream position in the sequence
  # M(0,0) is the upper left corner of the matrix
  * for (i,j) calculate the number of base pairs as the maximum of
      M(i+1,j)  # i unpaired
      M(i,j-1)  # j unpaired
      M(i+1,j-1)+1 if (i,j) is a base pair in the given structure
      M(i,k) + M(k+1,j) for each int k (i+1<=k<j-1) # bifurcation
   * return matrix M

FUNCTION traceback(m, i, j, basepairs):
  # Return an optimal nested structure for matrix position (i,j)
  # M is the filled matrix as returned by the fill procedure
  # i, j are the coordinates of the cell where the traceback should start
  # basepairs is a list of base pairs, same as in fill procedure

  * if M(i,j) equals 0 return the empty set
  * otherwise, if (i,j) is in basepairs and M(i,j) equals M(i+1,j-1)+1
    return the union of set([(i,j)]) and traceback(M,i+1,j-1,basepairs)
  * otherwise, for each int k (where i<=k<j)
    return union of traceback(M,i,k,basepairs) and traceback(M,k+1,j,basepairs)
```

Of course one should make sure that for all base pairs (i,j) in the list $i < j$. The algorithm has been implemented in our Python module available from the PyCogent project or the website (www.ibi.vu.nl/programs/k2nwww). The modification of the Nussinov-Jacobson algorithm has several disadvantages:

- Only one optimal solution is calculated, even though multiple optimal solutions might exist.
- This procedure is inefficient: it works fine for small lists of base pairs, but the running time gets out of hand for larger structures. Therefore the method is not available trough the web interface.
- Maximizing the number of base pairs is just one criterion that could be optimized, but there are many other options. It might be biologically relevant to incorporate sequence or alignment information (for example, think about a score for the conservation of a helix). In addition, this procedure can only score one base pair at the time, not a whole helix.

To overcome these limitations, we have implemented an new optimization approach which will be described below.

1.2 An optimization approach that calculates all optimal solutions

We have developed and implemented an optimization approach that calculates all optimal nested structures under an arbitrary score function. Any score function can be used as long as the scores are additive. The function can either maximize or minimize some value. Our method is quite efficient, because it treats paired regions as a unit (see Main Text) and it takes several other precautions to reduce the number of calculations, which will be explained below.

In addition to the optimization routine that calculates all optimal structures (OA) we provide two helper functions in case a user needs a single nested structure. OSR returns a *single* optimal solution at *random*, OSP returns a *sinlge* solution by comparing some (biological) *properties* of the optimal solutions. Two optimization criteria are implemented: maximizing the number of base pairs (BPS), maximizing the number of hydrogen bonds (HB) (a GC pair has three hydrogen bonds, AU and GU pairs have two) (Mathews et al. 1999).

The overall procedure followed by the OA method can be described as follows:

```
FUNCTION optimize_all(basepairs):
  * break the base pairs up into paired regions
  * separate non-conflicting regions from conflicting regions
  * the initial single solution is the structure containing all
    non-conflicting regions
```

```
  * for each clique of mutually conflicting paired regions
    calculate all optimal solutions and merge each one with
    each already existing solution to create a new set of optimal
    solutions
```

For each set of mutually conflicting paired regions (or knot-component) a dynamic programming search matrix is filled.

```
FUNCTION dp_matrix(paired_regions, goal, scoring_function):
  # return a filled DP search matrix.
  # Each cell should contain all optimal solutions from position i to j.
  # paired_regions (prs for short) is a set of pseudoknotted regions
  # goal is maximize or minimize
  # scoring_function is a function that returns a score for a paired region

  * pre-calculate the score of each paired region using the scoring function
  * the size S of the matrix is 2 times the number of paired regions
  * create a matrix M with S rows and columns
  * initialize each cell with the list of empty solutions
  * map the start and end positions of the regions such that each
    index in the matrix corresponds to a start or end position

  * iterate over each cell (i,j) in M (0<=j<S, j-1>.=i>=0)
  * in each cell (i,j) store the set of all optimal solutions
    from the following options:
    # what is "optimal" depends on the goal
      -- each solution in M(i,j-1)
      -- each solution in M(i+1,j)
      -- If i and j correspond to the start and end of the same
         paired region: all solutions in M(i+1,j-1) merged with
         the region starting at i and ending at j
      -- For each solution in M(i,j-1) and each solution in M(i+1,j)
         if both solutions are disjoint: the merge of M(i,j-1) and M(i+1,j)
      -- For each solution in M(i,j-1) and each solution in M(i+1,j)
         if both solutions are NOT disjoint: merge each solution
         M(i,k) with each solution in M(k+1,j) for each relevant k
  * return matrix M
```

The exact details of the procedure can be found in the source code available under the PyCogent project in SourceForge.

In normal English the description above reads as follows:

The number of paired regions in an RNA structure can be very large, and many regions will not conflict with other regions. Therefore, the optimization approach starts by adding non-conflicting regions to the solution, and splitting the conflicting regions into groups of mutually conflicting regions (a.k.a. knot-components Rødland (2006)). For example, suppose a set of regions is organized in the following order (where the prime indicates the downstream half of a region): ABCC'DD'A'B'EFGF'E'G'. In this example, region C and D are not involved in any conflict, so they will be part of the nested structure. Conflicting regions A and B would be in one group, and mutually conflicting regions E, F, and G would be in another group. For each knot-component the regions that should become part of the nested structure are calculated in a dynamic programming matrix.

The number of cells in each DP matrix is the number of given paired regions times two, because there is one row and column for each start and end point of each region. Only the top-right half of the matrix will be filled. A row index is referred to as i, a column index is referred to as j. The matrix is initialized on the diagonal (where $i == j$) with a list containing an empty solution. A list is used because we keep track of *all* possible optimal choices. For each cell (i, j) where $j > i$ we collect all the candidate-solutions as follows.

1. Add all solutions of the cell to the left, which contains the best solutions for the area from start point i to end point $j - 1$.

2. Add all solutions of the cell to the bottom, which contains the best solutions for the area from start point $i + 1$ to end point j.

3. If there is a region ranging exactly from i to j (in other words, i and j correspond to the start and end point of the same paired region), add all possible solutions from the cell to the bottom-left plus this region. The cell to the bottom-left contains the optimal solutions for the area from start point $i + 1$ to end point $j - 1$.

4. If the lists of solutions at the cells to the left and below both differ from the empty solution, we need to perform several additional tests:

   (a) For each combination of a solution in the left cell and a solution in the right cell, calculate the highest end point of the solution in the left cell and the lowest start point for the cell below. In a collection of paired regions, every region has an end point, and the highest end point is the largest number in the list of all end points. The lowest start point is identified in a similar way.

   (b) If the highest end point is lower than the lowest start point, it means both solutions are disjoint and can be added to form a better solution. Thus, merge the two solutions and add them to the list of candidate solutions.

   (c) Otherwise, the solutions are not disjoint, but, because of the pseudoknots, sub-solutions of the two solutions might be combined to form a better solution.

Create a slider (k) that runs from the lowest start point minus one to the highest end point plus one. For each combination of cells (i, k) and $(k + 1, j)$, merge all possible solutions and add them to the list of candidate solutions.

5. Store in cell (i, j) all solutions that are optimal for the given scoring function and goal. For efficiency reasons the score for each paired region is pre-calculated at the beginning of the function. There might be multiple optimal solutions among the candidate solutions.

6. Finish the calculation when the top-right cell in the matrix is filled. This cell contains the optimal solutions for the given set of paired regions.

1.3 Efficiency comparison

We compared the running time of optimizing the number of base pairs with the modification of the Nussinov-Jacobson algorithm (NR = nussinov restricted) and our optimization algorithm (OA). Before the calculation, we have improved the efficiency of the NR algorithm by making it renumber the structures such that unpaired bases do not take up space in the matrix. We applied the algorithms to the selection of canonical base pairs for five crystal structures. The calculations were done on a Mac 2.16 GHz Intel Core Duo with 1GB RAM. The running times on these five different structures with varying sequence lengths and number of base pairs are listed in the following table:

PDB ID	Seq len	# pairs	NR (CPU sec)	OA (CPU sec)	Speed-up
1U8D	67	23	0.488	0.387	1.26
2NOQ	190	51	2.884	0.384	7.51
2A64	298	89	2.631	0.387	6.80
2AVY	1530	450	290.652	0.646	449.93
2AW4	2841	821	1749.919	365.683	4.79

On the three shorter sequences the speed-up of the OA method compared to the NR algorithm is significant, but all running times are extremely short, which makes it less urgent. On the 16S rRNA structure (2AVY) the improvement in running time very large, showing that the OA method is very efficient on long structures with relative simple knot-components, where the NR method has to work its way through every base pair, even those not involved in pseudoknots. The OA algorithm needs much more time to digest the large knot-component (involving 80 paired regions and 279 base pairs) in the rRNA large subunit (2AW4), but it runs still almost five times faster than the NR method. For the average user a running time of 6 minutes is more acceptable than 30 minutes. In addition, the users obtains all optimal solutions in this shorter time.

2 Additional information about the heuristic approaches

2.1 Circular removal in conflict-elimination methods

Circular removal, leading to unsaturated nested structures, can occur both in the EC and the EG method. An example of circular removal would be that region A (which conflicts with B and C) is removed, and all of its conflicts (B and C) are removed in the downstream process, which leads in the end to region A being eliminated even though it does not conflict with any region in the nested structure. Therefore, both the EC and EG methods add these eliminated non-conflicting regions back into the list of paired regions, starting at the 5' end. By monitoring the process in a large collection of artificial RNA structures we found that the circular-removal problem occurs more often in the EG methods.

3 Supplementary Discussion

3.1 Artificial structures

We compared the behavior of the different pseudoknot elimination methods on sets of randomly generated RNA structures, in which we varied the lengths of the sequences and the number of paired regions. We observed several clear trends over our data (Table 1). First, all the heuristics are unique, in that they produce different results on at least some structures. In general, the incremental methods produced a more diverse set of solutions, while the two conflict elimination methods more often find the same solution than different solutions. For example, in 1000 knotted structures, in which ten paired regions are embedded in a sequence of length 250, the three conflict elimination approaches found a different nested structure for 34% of the knotted structures. The incremental heuristics found different solutions for 51% of the knotted structures. Finally, for 0.8% of the knotted structures, each of the heuristics found a distinct solution. The difference in behavior is illustrated in Figure 1 (main text), where all heuristics generate a different nested structure. In this specific example, the EG method found the same solution as the OA approach, although in other situations the EG method might find a non-optimal structure and other methods might find an optimal solution.

For many structures multiple optimal solutions (as found by the OA method maximizing the number of base pairs) exist, and several heuristics typically find an optimal solution for a given structure, although these optimal solutions often differ. In almost 97% of the less complex knotted structures that contained ten paired regions, at least one of the heuristics found a solution of optimal length. However, when the complexity of the pseudoknots increases, it becomes increasingly difficult for the heuristics to find an optimal solution. The EG method is the method that most often found an optimal solution, followed closely by the EC and IL methods. The results clearly demonstrated that the IR and IO methods are not designed to keep the maximum number of base pairs.

3.2 Biological structures

We studied the behavior of the different pseudoknot elimination methods on two different collections of base pairs, extracted from eleven X-ray crystal structures (see Material and Methods). One was the set of canonical base pairs, the other the collection of canonical pairs plus all immediate helix extensions. Table 2 summarizes the results for the canonical pairs (results were similar for the helix-extension pairs).

Over our database, we found that for many knotted structures a majority of methods produced the same nested structure, which contained as many base pairs as an optimal solution found by the OA approach maximizing the number of base pairs. The method maximizing the number of hydrogen bonds (HB) always found a single optimal solution. The EC heuristic always produced a solution with the maximum number of base pairs. Typically, one of the incremental methods found a nested structure containing fewer base pairs than the optimal structure. For five out of the eleven knotted structures, multiple optimal solutions (in terms of base pairs) were found. In most structures where a majority of methods returned the same nested structure, the pairs that were "broken" to obtain a nested structure agreed with the expert view.

The initial choice of base pairs and the principles behind the different heuristics influence the nested structure that is found. In addition, several optimal nested structures might exist for a single knotted structure. The different optimal solutions can be compared in several ways to determine which of the optimal solutions will be used in the analysis. These comparisons might include the number, length, or position in the sequence of the paired regions they contain.

Supplementary tables

Sequence length	100	100	250	250	500	500	1000	1000
# paired regions	7	10	10	20	10	20	10	20
# structures	1000	1000	1000	1000	1000	1000	1000	1000
AVG # optimal solutions	1.315	1.504	1.235	1.567	1.200	1.368	1.208	1.329
Optimal not found by heur	0.005	0.007	0.020	0.102	0.026	0.126	0.033	0.144
Optimal found by 1 heur	0.021	0.060	0.078	0.225	0.089	0.241	0.083	0.258
Optimal found by >1 heur	0.974	0.933	0.902	0.673	0.885	0.633	0.884	0.598
EC found optimum	0.880	0.802	0.796	0.534	0.802	0.548	0.798	0.562
EG found optimum	0.918	0.890	0.851	0.698	0.830	0.664	0.827	0.647
IO found optimum	0.350	0.229	0.217	0.060	0.235	0.126	0.226	0.053
IL found optimum	0.848	0.783	0.708	0.536	0.677	0.450	0.653	0.410
IR found optimum	0.492	0.401	0.311	0.130	0.297	0.112	0.294	0.104
all ELIM methods diff	0.231	0.344	0.341	0.633	0.308	0.587	0.288	0.577
all ELIM methods same	0.769	0.656	0.659	0.367	0.692	0.413	0.712	0.423
all INC methods diff	0.307	0.517	0.513	0.856	0.517	0.843	0.513	0.841
all INC methods same	0.147	0.069	0.063	0.008	0.053	0.005	0.052	0.006
all HEUR diff	0.000	0.004	0.008	0.143	0.013	0.148	0.006	0.165
all HEUR same	0.136	0.054	0.045	0.002	0.036	0.005	0.039	0.002

Table 1: Performance of each method on artificial RNA structures. Each column in the table describes one set of structures of a particular size. Each row in the table reports a specific statistic of a run. The top part of the table specifies the length of the sequence and the number of paired regions in that sequence, and the number of knotted structures generated in that run. The second part specifies the average number of optimal solutions, containing the maximum number of base pairs, that was found by the OA approach (AVG # optimal solutions) and how often a solution of that length was found by a heuristic. The third part specifies for each method separately how likely it was to find a solution with the maximum number of base pairs. The bottom part of the table describes how often particular groups of methods returned the same solution (ELIM = EC and EG, INC = IO, IL, and IR, HEUR = EC, EG, IO, IL, and IR).

Description	Nested structures	Comment
Phe-tRNA PDB: 1EHZ L:76, BPS:23	1 (20 bps) EC,EG,IO,IL,IR BPS,HB	D-loop/T-loop interaction removed.
16S rRNA PDB: 2AVY L:1530, BPS:450	1 (439 bps) EC,BPS 2 (439 bps) EG,IL,IR,BPS,HB 3 (428 bps) IO	PKB64 (short-range in 1, long-range in 2), PKB65, and PKB129 broken, plus two long-range interactions. D_{12}=0.982, D_{13}=0.935, D_{23}=0.953.
23S rRNA PDB: 2AW4 L:2841, BPS:821	1 (788 bps) EC,EG,IL,BPS,HB 2 (787 bps) IR 3 (765 bps) IO	All, but one, removed regions ($>$ 1 bp) matched knots found by comparative sequence analysis, e.g. PKB 148 broken. Also, many single bp long-range interactions broken. D_{12}=0.994, D_{13}=0.942, D_{23}=0.936.
A-riboswitch PDB: 1Y26 L:71, BPS:25	1 (22 bps) EC,EG,IO,IL,BPS 2 (22 bps) IR,BPS,HB	Both nested structures: interaction between L2 and L3 broken (2 bps), plus one of two base triples. D_{12}=0.913.
G-riboswitch PDB: 1U8D L:67, BPS:23	1 (20 bps) EC,EG,IO,IL,BPS 2 (20 bps) IR,BPS,HB	Both nested structures: interaction between L2 and L3 broken (2 bps), plus one of two base triples. D_{12}=0.905.
SAM-riboswitch PDB: 2GIS L:94, BPS:29	1 (27 bps) EC,EG,IO,IL,IR BPS,HB	Interaction between L2 and J3/4 removed.
HDV ribozyme PDB: 1DRZ L:72, BPS:22	1 (14 bps) EC,EG,IO,IL,IR BPS,HB	P2 (6 bps) and interaction between J1/4 and L3 (2 bps) broken.
DA ribozyme PDB: 1YKV L:49, BPS:19	1 (15 bps) EC,EG,IL,BPS,HB 2 (14 bps) IO,IR	Struct. 1: G1G2&C26C25 and A3G4&U45C44 broken. Struct. 2: A3G4&U45C44 and helix II broken. D_{12}=0.647.
Group I intron PDB: 1ZZN L:197, BPS:50	1 (44 bps) EG,IO,IL,BPS,HB 2 (44 bps) EC,IR,BPS	P3 and P7 in knot. Struct. 1: P7 broken, struct. 2: P3 broken, because P7 spans shorter range. D_{12}=0.760.
RNaseP PDB: 2A64 L:298, BPS:89	1 (82 bps) EC,IR,BPS,HB 2 (81 bps) EG,IO,IL	Struct. 1: P2 broken (6 bps); Struct. 2: P4 broken (7 bps), both: long-range interaction between J3/4 and J19/4 broken (1 bp). D_{12}=0.852.
CrPV IRES PDB: 2NOQ L:190, BPS:51	1 (36 bps) EC,EG,IR,BPS,HB 2 (36 bps) BPS 3 (33 bps) IO,IL	Struct 1: PKII, PKIII and 3' side of PKI broken, Struct. 2: PKII, PKIII, 5' side PKI broken, Struct. 3: P2.2, PKII, 3' PKI. D_{12}=0.756, D_{13}=0.683, D_{23}=0.500.

Table 2: Pseudoknot removal in biological structures. The data shown in the table is for the set of canonical base pairs extracted from eleven crystal structures. The column "Description" contains a short description of the molecule, the PDB ID, the length (L) and the number of base pairs in the initial knotted structure (BPS). The "Nested structures" column reports for each unique solution how many base pairs it contained (between parentheses) and by which methods it was found (BPS is the OA method maximizing the number of base pairs, HB is the OA method maximizing the number of hydrogen bonds). The "Comment" column reports which paired regions were removed when compared to the literature (Shi and Moore 2000; Holbrook 2005; Schuwirth et al. 2005; Cannone et al. 2002; van Batenburg et al. 2000; Serganov et al. 2004; Batey et al. 2004; Montange and Batey 2006; Ferré-D'Amaré et al. 1998; Serganov et al. 2005; Stahley and Strobel 2005; Kazantsev et al. 2005; Schüler et al. 2006). In case of multiple solutions, this column also contains the distance between two structures. D_{12} means the distance between structure 1 and 2. The distance measure is the number of base pairs common to both structures (i.e. the intersection) divided by the number of base pairs that is in one or the other structure (i.e. the union).

References

Batey R.T., Gilbert S.D., and Montange R.K. 2004. Structure of a natural guanine-responsive riboswitch complexed with the metabolite hypoxanthine. *Nature* **432**: 411–415.

Cannone J.J., Subramanian S., Schnare M.N., Collett J.R., D'Souza L.M., Du Y., Feng B., Lin N., Madabusi L.V., Muller K.M., Pande N., Shang Z., Yu N., and Gutell R.R. 2002. 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.

Ferré-D'Amaré A.R., Zhou K., and Doudna J.A. 1998. Crystal structure of a hepatitis delta virus ribozyme. *Nature* **395**: 567–574.

Holbrook S.R. 2005. RNA structure: the long and the short of it. *Curr. Opin. Struct. Biol.* **15**: 302–8.

Kazantsev A.V., Krivenko A.A., Harrington D.J., Holbrook S.R., Adams P.D., and Pace N.R. 2005. Crystal structure of a bacterial ribonuclease P RNA. *Proc. Natl. Acad. Sci.* **102**: 13392–13397.

Mathews D.H., Sabina J., Zuker M., and Turner D.H. 1999. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.* **288**: 911–40.

Montange R.K. and Batey R.T. 2006. Structure of the S-adenosylmethionine riboswitch regulatory mRNA element. *Nature* **441**: 1172–1175.

Nussinov R. and Jacobson A.B. 1980. Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc. Natl. Acad. Sci.* **77**: 6309–6313.

Ponty Y. 2006. *Modélisation de séquences génomiques structurées, génération aléatoire et applications*. Ph.D. thesis, Laboratoire de recherche en Informatique, Université Paris-Sud XI.

Rødland E.A. 2006. Pseudoknots in RNA secondary structures: representation, enumeration, and prevalence. *J. Comput. Biol.* **13**: 1197–1213.

Schüler M., Connell S.R., Lescoute A., Giesebrecht J., Dabrowski M., Schroeer B., Mielke T., Penczek P.A., Westhof E., and Spahn C.M.T. 2006. Structure of the ribosome-bound cricket paralysis virus IRES RNA. *Nat. Struct. Mol. Biol.* **13**: 1092–1096.

Schuwirth B.S., Borovinskaya M.A., Hau C.W., Zhang W., Vila-Sanjurjo A., Holton J.M., and Cate J.H.D. 2005. Structures of the bacterial ribosome at 3.5 Å resolution. *Science* **310**: 827–34.

Serganov A., Keiper S., Malinina L., Tereshko V., Skripkin E., Hobartner C., Polonskaia A., Phan A.T., Wombacher R., Micura R., Dauter Z., Jaschke A., and Patel D.J. 2005. Structural basis for Diels-Alder ribozyme-catalyzed carbon-carbon bond formation. *Nat. Struct. Mol. Biol.* **12**: 218–224.

Serganov A., Yuan Y.R., Pikovskaya O., Polonskaia A., Malinina L., Phan A.T., Hobartner C., Micura R., Breaker R.R., and Patel D.J. 2004. Structural basis for discriminative regulation of gene expression by adenine- and guanine-sensing mRNAs. *Chem. Biol.* **11**: 1729–1741.

Shi H. and Moore P.B. 2000. The crystal structure of yeast phenylalanine tRNA at 1.93 Å resolution: a classic structure revisited. *RNA* **6**: 1091–1105.

Stahley M.R. and Strobel S.A. 2005. Structural evidence for a two-metal-ion mechanism of group I intron splicing. *Science* **309**: 1587–1590.

van Batenburg F.H., Gultyaev A.P., Pleij C.W., Ng J., and Oliehoek J. 2000. PseudoBase: a database with RNA pseudoknots. *Nucleic Acids Res.* **28**: 201–204.