RNA Modeling Using the MC-Fold and MC-Sym Pipeline

Marc Parisien and François Major

Institute for Research in Immunology and Cancer (IRIC),
Department of Computer Science and Operations Research,
Université de Montréal, PO Box 6128, Downtown Station,
Montréal, Québec, Canada H3C 3J7

This manual describes the modeling of RNA using the MC-Fold and MC-Sym pipeline. The manual is partitioned into sections which address specific modeling tasks.

Figure 1: Molecular modeling of an argonaute silencing complex. *Thermus thermophilus* argonaute (PDB file 3F73 [1]; grey) complexed with a mature micro-RNA (miRNA; yellow) and it's target messenger-RNA (mRNA; blue). The protein and nucleotides 1 to 8, and 19 to 21 of the guide strand have been taken from the PDB file. Other nucleotides, including the mRNA, have been built with the MC-Sym computer program using Nucleotide Cyclic Motifs. The chosen model is the one that minimizes atomic clashes between the protein and the RNA duplex.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Modeling RNA structure allows one to resume all known structural data into a coherent construct. It can also suggest new experiments to perform to probe the RNA's structure, and to suggest bio-molecular operating schemes.

Here, we will explain how to model an RNA structure using the MC-Fold and MC-Sym pipeline [2], by explaining the many steps of the modeling process.

The web portal of the MC-Fold and MC-Sym pipeline is here:

http://www.major.iric.ca/MC-Pipeline/

In this manual you will find these sections:

- How to use MC-Fold. MC-Fold is a program that takes for input an RNA sequence and produces for output a list { } of sub-optimal secondary structures:

```
sequence -> { secondary structure }
```

- How to use MC-Cons. MC-Cons is a program that takes for input a list { } of tuples ( ) of sequences and their list { } of associated sub-optimal secondary structures, and produces for output a list { } of tuples ( ) of sequences and one of their associated secondary structures, such that the chosen secondary structures maximize their resemblance to all others:

```
{ ( sequence, { secondary structure } ) } ->
    { ( sequence, secondary structure ) }
```

- How to use MC-Sym. MC-Sym is a program that takes for input a tuple ( ) of sequence and it's secondary structure, and produces for output a list { } of tertiary structures:

```
( sequence, secondary structure ) -> { tertiary structure }
```

- How to analyze MC-Sym's results, the set of 3D structures generated by MC-Sym, specially with respect to experimental probing data.

From these definitions, one can see that the output of one program serves as the input of the next one, hence the pipeline:

```
MC-Fold | MC-Sym
MC-Fold | MC-Cons | MC-Sym
```

Chapter 2

Using MC-Fold

MC-Fold is a computer program that predicts RNA secondary structures from sequence [2]. It's work flow is depicted in Figure 2.1. The program is accessible on-line through a web server at this address:

http://www.major.iric.ca/MC-Fold/

RNA secondary structure prediction is a field in constant ebullition [3]. The most prominent approach to structure prediction is the use of Turner's thermodynamics tables combined to a nearest neighbor model [4]. This approach seems the most promising because it is based on a biophysical model. However, despite decades of parameter compilation, the problem of structure prediction remains, mainly because the contributions of non-canonical base pairs are ill-characterized. MC-Fold attempts to take into account the in-stem non-canonical base pairs, and suites of, into a unified energetic framework. Statistics from the Protein Data Bank [5] on Nucleotide Cyclic Motifs (NCMs [2]) are used to derive a pseudo-potential energy function. The NCM naturally embraces all base pair types. MC-Fold is thus of the knowledge-based type, like CONTRAfold [6], as opposed to physics-based types like Mfold [7], Sfold [8], RNAfold [9] and RNAstructure [10]. A brief list of on-line resources for secondary structure prediction ca be found in Table 2.1.

Program	URL	Reference
ILM	http://cic.cs.wustl.edu/RNA/	[11]
Sfold	http://sfold.wadsworth.org/srna.pl	[8]
Pfold	http://www.daimi.au.dk/ compbio/pfold/	[12]
Mfold	http://frontend.bioinfo.rpi.edu/applications/mfold/cgi-bin/rna-form1.cgi	[7]
VSfold	http://www.rna.it-chiba.ac.jp/vsfold5	[13]
RNAfold	http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi	[9]
CONTRAfold	http://contra.stanford.edu/contrafold/	[6]
RNAstructure	http://rna.urmc.rochester.edu/rnastructure.html	[10]

Table 2.1: On-line secondary structure prediction resources. The list is not exhaustive.

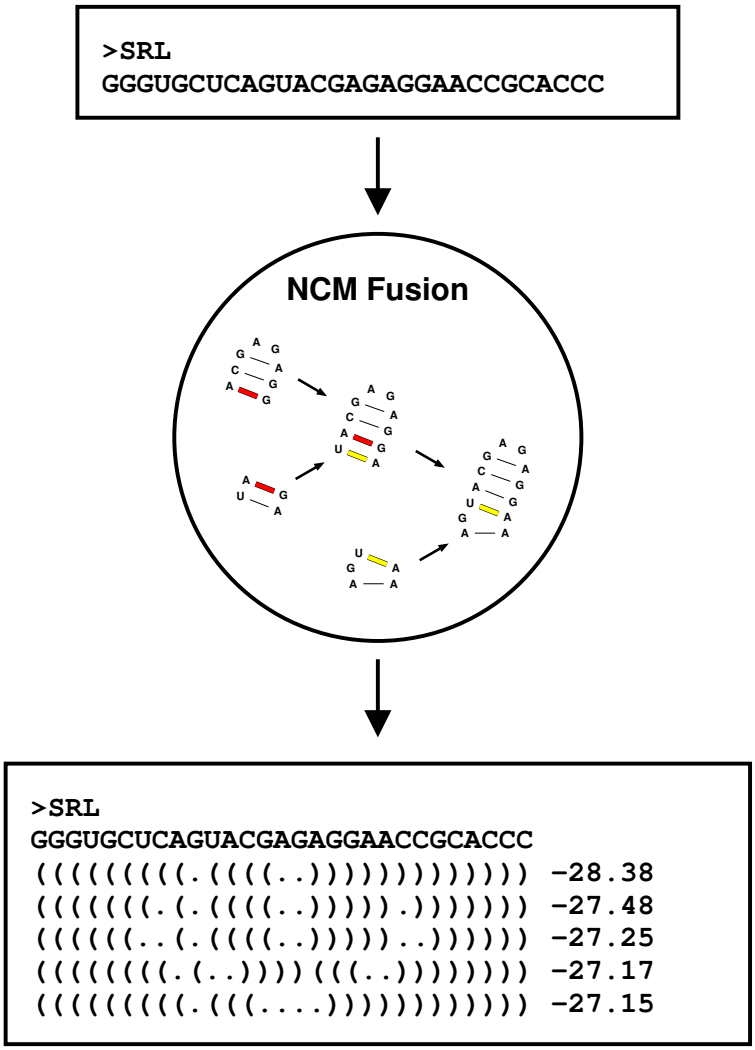


Figure 2.1: MC-Fold work flow. The user specifies the RNA sequence along it's name as an input to MC-Fold (top box). MC-Fold's Nucleotide Cyclic Motif (NCM) fusion process generates and scores secondary structures (middle circle). The output of MC-Fold is a list of secondary structures, sorted by decreasing energy values (bottom box). The top structure if the minimum free energy structure.

2.1 Submitting a sequence

Submitting a sequence to MC-Fold is as easy as 1-2-3:

1. Fill the *Sequence* input field of the "**1. Sequence**" section with the RNA sequence of your choice. The sequence should be no longer than 150 nucleotides. Valid symbols are A, C, G, U and T (converted to U). Let's try the following sequence:

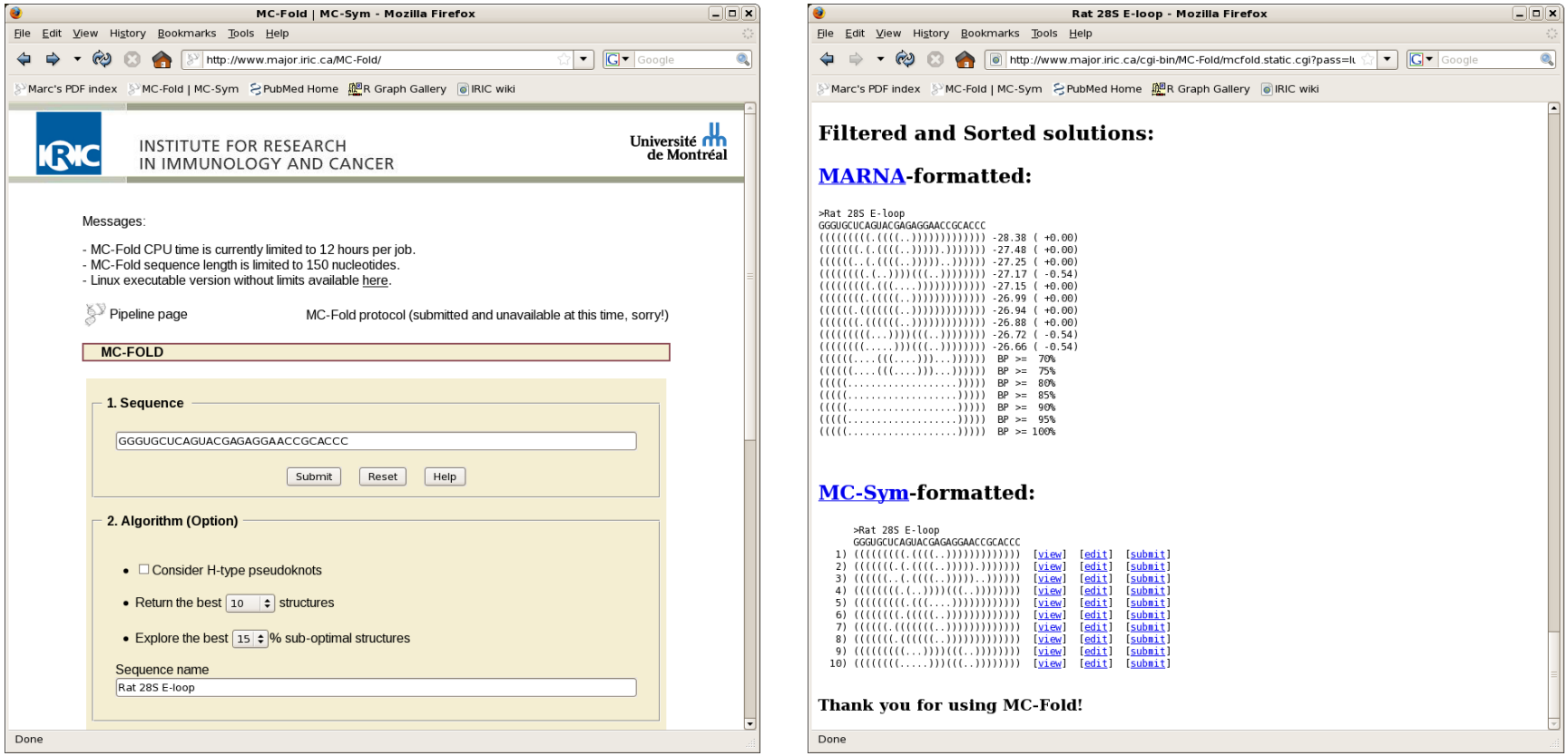
   ```
   GGGUGCUCAGUACGAGAGGAACCGCACCC
   ```

2. Fill the *Sequence name* input field of the "**2. Algorithm (Option)**" section. The sequence is that of:

   ```
   Rat 28S E-loop
   ```

   At this point, your web form should look similar to Figure 2.2a.

3. Click the **Submit** button.



(a) (b)

Figure 2.2: MC-Fold web form. **(a)** MC-Fold input page. The sequence and sequence name input fields are filled. We would also like to have the top 10 best structures. All other options are at their default values. **(b)** MC-Fold output page. The MC-Sym-formatted section allows for further analysis via the three web links `[view]`, `[edit]`, `[submit]`.

2.2 Analyzing the results

Browse to the bottom of MC-Fold's results page, which should look like Figure 2.2b. In the **MC-Sym-formatted** section you will find the sub-optimal structures, ranked by energy from best at position 1. Besides each secondary structures there are three web links:

1. [view] will render the secondary structure in a visually appealing format. Pseudoknots are not rendered. The renderer is a slightly modified version distributed with CONTRAfold [6].
2. [edit] will bring a web form to tailor MC-Sym's script.
3. [submit] will submit the secondary structure to MC-Sym.

In the **MARNA-formatted** section you will also find the sub-optimal structures, ranked by energy from best at position 1. This section is custom-tailored for the MARNA web server [14] and for MC-Cons (see Chapter 3). Base pair conservation levels, from 70% to 100% are also shown. This gives an idea about the dynamics of the structure. It can also be seen as the equivalent of the base pair probabilities [15].

2.3 Using the options

In section "**2. Algorithm (Option)**" of MC-Fold's web form there are several options to tinker with:

- If you suspect that your sequence folds into an H-type pseudoknot [16], then check the *Consider H-type pseudoknots* option. Pseudoknots of the H-type are the most common, and are of the form ABAB [17]. Also, increase the percentage of sub-optimal structures explored.
- MC-Fold can produce more or less sub-optimal structures, and keep the best N structures produced. Adjust the *Return the best N structures* option to suit your needs. It is better to produce more sub-optimal structures if one intends to use MC-Cons after. MC-Fold may produce less than the number N specified, depending on the percentage of sub-optimal structures explored.
- The sub-optimal search space can be constrained within a percentage of the minimum free energy structure, as MC-fold makes use of the Waterman-Byers algorithm [18, 19]. Because the exploration has an exponential time complexity, increasing this value can have a dramatic effect on MC-Fold's run time.

In section "**3. Structural Constraints (Option)**" of MC-Fold's web form one can provide a structural mask, which forces certain nucleotides to be either paired or unpaired. This option should be used if one is sure of the paired/unpaired status of certain nucleotides. Using this option will restrict the conformational search space, and the time to explore it. These structural constraints are always respected. If one wants to coax a particular nucleotide to be unpaired, but not necessarily, then the next option should be used.

Finally, section "**4. Single-Stranded Chemical/Enzymatic Reactivity Data (Option)**" is used to mark the reaction intensity of nucleotides to single-stranded probing. The most widely used RNA structure probes are either chemical or enzymatic (see [20] for a review). Usefulness of low-resolution probing data is shown here [10]. Confidence in single-strandedness is increased by the consensus attack of different single-stranded probes. Notably, of the chemical probes, consider SHAPE [21], which cleavage activity corresponds to backbone flexibility. Experiment with different masks to see if the solutions are the same. In Figure 2.3 we used the SHAPE data published by Weeks' group [21] to predict the secondary structure of the tRNA. Here, we only considered adjacent nucleotides which featured moderate to high reactivity.

```
>tRNA ASP
.................xx.............xxxx.................x..................  High
...............xx..x................xx................xx..................  Medium
GCCGUGAUAGUUUAAUGGUCAGAAUGGGCGCUUGUCGCGUGCCAGAUCGGGGUUCAAUUCCCCGUCGCGGCGC
(((((((..(((((......))))).(((((.......))))))...((((((...))..)))))))))).. native   2TRA    RNK TP FP FN Mthw
(((((((((((((.......))))))).(((((.......))))).....((((((...))..)))))))))).. -59.94 ( -1.45)    1 18  6  6 75.0
(((((((.(((((.......))))).(((((.......))))))...((((((...))..)))))))))).. -59.93 ( -1.04)    2 19  5  5 79.2
(((((((((((((......)))))).(((((.......))))).)))))).((((((...))..)))))).. -59.76 ( -0.85)    3 23  3  1 92.1
(((((((.(.(((.......))))).(((((.......))))))..((((((...))..)))))))))).. -59.57 ( -0.81)    4 19  5  5 79.2
(((((((((((((........))))).(((((.......))))).)))))).((((((...))..)))))).. -58.94 ( -0.85)    5 22  3  2 89.8
(((((((..(((.......))))).(((((.......))))))..((((((...))..)))))))))).. -58.92 ( -0.98)    6 19  5  5 79.2
(((((((.(.(((.......))))).(((((.......))))).).((((((...))..)))))))))).. -58.79 ( -0.81)    7 19  5  5 79.2
(((((((..(((((......))))).(((((.......))))))...((((((...))..)))))))))).. -58.63 ( -0.89)    8 24  0  0 100
(((((((((((((.......)))).((((((.......))))).)))))).((((((...))..)))))).. -58.45 ( -0.77)    9 18  7  6 73.5
(((((((...(((.......))))).(((((.......))))))..((((((...))..)))))))))).. -58.43 ( -1.27)   10 23  1  1 95.8
```

Figure 2.3: MC-Fold predictions for the yeast $tRNA^{ASP}$. The top ten structures generated by MC-Fold for the Yeast tRNA under SHAPE constraints are shown. The native structure (PDB file 2TRA) ranks 8^{th} (Matthews coefficient ratio of 100%). The numbers in parenthesis represent the energy contributions of the coaxial stacking. The nucleotides marked with an 'x' under the "High" SHAPE constraints have an 8 kcal/mol penalty if found paired; 4 kcal/mol for "Medium". Nucleotide 47 is absent.

Chapter 3

Using MC-Cons

MC-Cons is a computer program that assigns an RNA secondary structure among sub-optimals for many sequences, such that the assigned secondary structures maximize the resemblance to all others [2]. It's work flow is depicted in Figure 3.1. The program is accessible on-line through a web server at this address:

http://www.major.iric.ca/MC-Cons/

The consensus structural assignment done here is different in spirit of those programs that try to build a consensus secondary structure, like Dynalign [22], MARNA [14] or RNAshapes [23]. The consensus secondary structure often lacks many base pairs, thus more difficult to project in 3D space. Furthermore, MC-Cons is able to partition the sequences into many structural classes, as shown with the Iron Responsive Element hairpin (between V- and W-bulged stems, see the Supplementary Information of [2]). A brief list of on-line resources for consensus structure prediction ca be found in Table 3.1.

Program	URL	Reference
MARNA	http://www.bioinf.uni-freiburg.de/Software/MARNA/index.html	[14]
RNAshapes	http://bibiserv.techfak.uni-bielefeld.de/rnashapes/	[23]
RNAalifold	http://rna.tbi.univie.ac.at/cgi-bin/RNAalifold.cgi	[24]
Dynalign	http://rna.urmc.rochester.edu/dynalign.html	[22]
PARTS	http://rna.urmc.rochester.edu/parts.html	[25]

Table 3.1: On-line consensus structure prediction resources. The list is not exhaustive.

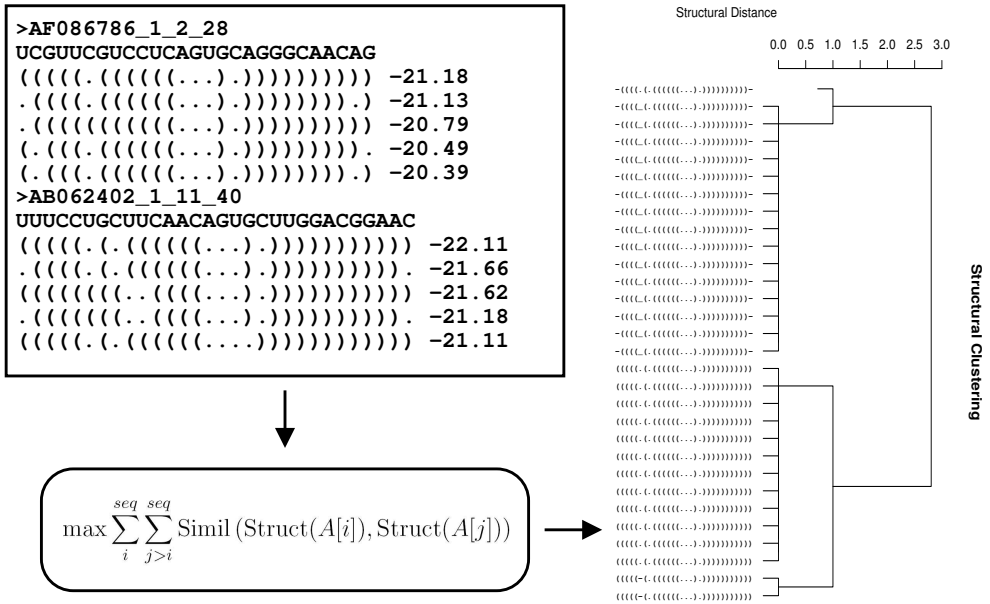


Figure 3.1: MC-Cons work flow. The user specifies many RNA sequences with their list of sub-optimal, alternative, secondary structures as an input to MC-Cons (upper-left box). MC-Cons assigns an alternative structure $A[i]$ to each sequence i, such that their pairwise similarity is maximized (bottom-left oval). The output of MC-Cons is a list of structures which have been assigned to each sequence (right pane). The structures can be clustered into structural classes.

3.1 Preparing the data

MC-Cons performs best when:

- The sequences are short, i.e. less than 100 nts.
- The sequences are of the same lengths, +/- a few ins and dels.
- There are many (>2), but not too many (<30) sequences.

Before using MC-Cons we need to prepare it's input. Consider these five sequences taken from Figure 3B of dos Santos et al. 2008 [26]:

```
>D_melanogaster
UGCUUGCAUACUGCUUUGGCCAGGACCAAAACGUAUGCGAAGUG
>D_yakuba
UGCUUGCACACUGCUUUGGCCAGGACCAAAACGUGUGCGAAGUG
>D_prosaltans
UACUUGCAUUUGGCUUGGGCCUGGACCCUAAGCAAAUGUCAGGUU
>D_hydei
UGCUUGCAUUUCCAAUUUGCCCAGCUGCAAAUCGCAUGUUAAGCA
>Z_tuberculatus
UGCUUGCAUUGCGCUUUGCUCGGCUGCAAAGCAAAUGUUAAGCA
```

For each sequence, get a list of sub-optimal secondary structures using MC-Fold (see Chapter 2). Here, consider the top 100 sub-optimal structures for each sequence. It is better to consider many more sub-optimals in MC-Fold, then let MC-Cons progressively deal with the top 10, 25, etc... Copy-and-paste all the **MARNA-formatted** sections of all MC-Fold runs in a text file[1] named `wle3.marna`. The content of this file should look like the one in Figure 3.2.

[1] for Windows users; use Wordpad instead of Microsoft Word.

```
>D_melanogaster
UGCUUGCAUACUGCUUUGGCCAGGACCAAAACGUAUGCGAAGUG
((((((((((.((((((((((..))))))))))))))).)))))  -43.81 ( +0.00)
... [98 structures] ...
.((((((((((..(((((((((..)))))))).))))))).)))).  -38.79 ( +0.00)
>D_yakuba
UGCUUGCACACUGCUUUGGCCAGGACCAAAACGUGUGCGAAGUG
((((((((((.((((((((((..))))))))))))))).)))))  -44.22 ( +0.00)
... [98 structures] ...
.((((.(((((.((((((((((..))))))))))))))..)))).  -39.21 ( +0.00)
>D_prosaltans
UACUUGCAUUUGGCUUGGGCCUGGACCCUAAGCAAAUGUCAGGUU
(((((((((((((((((((((..)))))))))))))))).)))))  -45.05 ( +0.00)
... [98 structures] ...
.((((((((((((.(((((((..))))))).))))))).)))))).  -41.31 ( +0.00)
>D_hydei
UGCUUGCAUUUCCAAUUUGCCCAGCUGCAAAUCGCAUGUUAAGCA
((((((.((((((((((((((....))))))))))))))))))))  -39.83 ( +0.00)
... [98 structures] ...
(((((((((..(((((((((..((..)))))))))))))).)))))  -36.76 ( +0.00)
>Z_tuberculatus
UGCUUGCAUUGCGCUUUGCUCGGCUGCAAAGCAAAUGUUAAGCA
((((((((((.(((((((((....))))))))))))))).)))))  -45.21 ( +0.00)
... [98 structures] ...
((((((((((.(((((((..((..)))))))))))))).)))))  -40.52 ( +0.00)
```

Figure 3.2: MC-Cons input file `wle3.marna`. The file contains five sections because it features five sequences. Each section is composed of 1. the sequence name, preceded with the '>' character, 2. the actual sequence and 3. a list of sub-optimal secondary structures associated with the sequence.

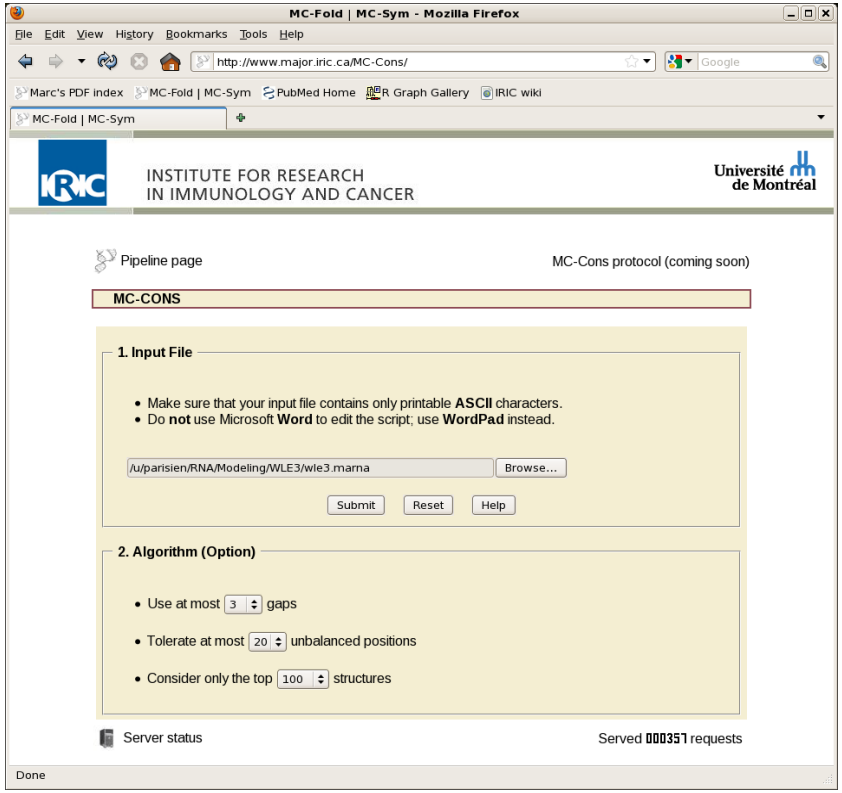
3.2 Submitting the data

Submitting a multiple-sequences data set to MC-Cons is as easy as 1-2-3:

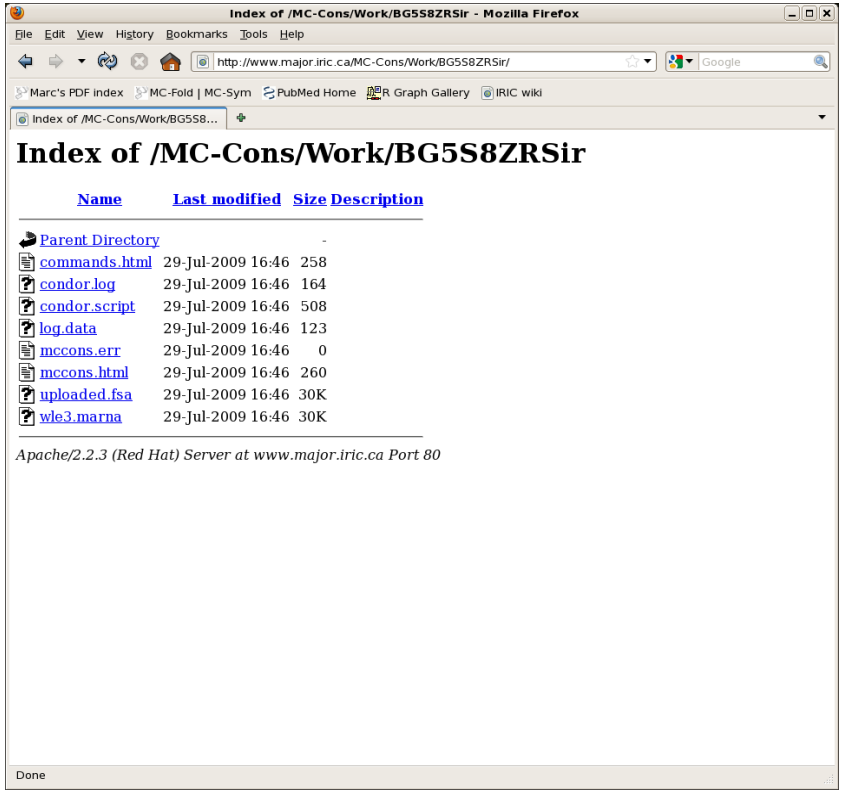
1. Fill the *file* input field of the "**1. Input File**" section by clicking the **Browse...** button and locating MC-Cons' input file `wle3.marna`.
2. Set the number of structures to the top 100 in the "**2. Algorithm (Option)**" section.

   At this point, your web form should look similar to Figure 3.3a.
3. Click the **Submit** button.

From there, the web server will create a working directory and redirect the web browser to this folder. Figure 3.3b shows such a working directory, whose 10-digit key is BG5S8ZRSir. This web page can be bookmarked for later use. The URL of this web page can be exchanged between researchers in a collaborative modeling effort, and is accessible across the world via the Internet. The keys are generated randomly from a mixture of letters and numbers, hence offending substrings can appear fortuitously. Directory keys are unique to each submit commands; the key you will obtain will assuredly be different from the one shown here.



(a)



(b)

Figure 3.3: MC-Cons web form. **(a)** MC-Cons input page. The location of the `wle3.marna` file is given. We would also like to consider the top 100 best structures of each sequence. All other options are at their default values. **(b)** The MC-Cons working directory, whose key is BG5S8ZRSir. This directory is automatically attributed by the web server for each MC-Cons run.

3.3 Analyzing the results

The results of MC-Cons are laid down in the `mccons.html` file, which is found in the working directory attributed to your MC-Cons run. Figure 3.3b shows a directory listing in which the file `mccons.html` appears. Simply click on the web link of the file.

Since the MC-Cons job is queued on our web server, and that the results can be delayed depending on the server's load, the `mccons.html` file can be empty or partially filled at any moment. Simply use the refresh command of the web browser to update the content of the file. The MC-Cons job is completed when the message **Thank you for using MC-Cons!** is found at the very bottom of the `mccons.html` page.

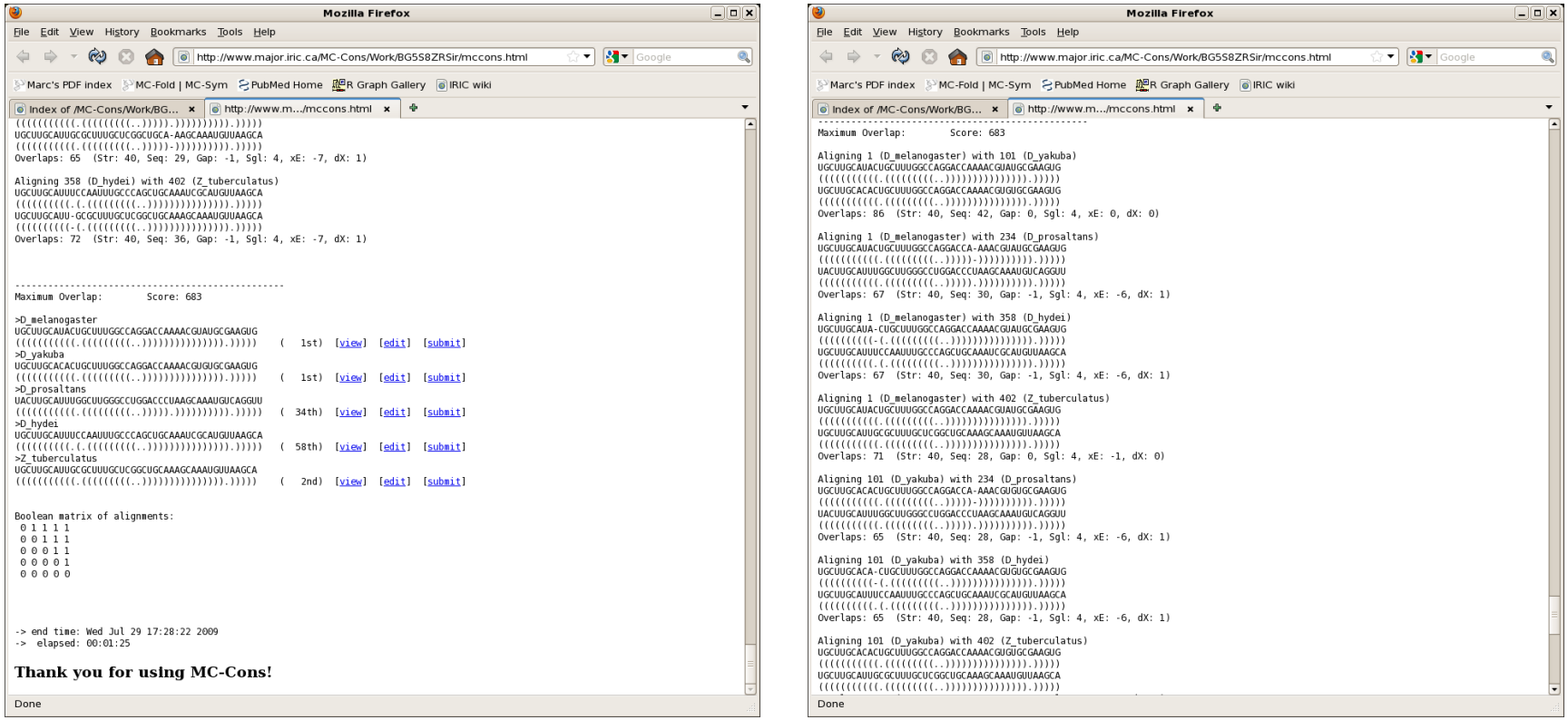
The first thing to inspect is the Boolean matrix of alignments, found at the bottom of the `mccons.html` file. Figure 3.4a shows such a matrix. In our case, the upper-triangular matrix has only '1' as entries, which signifies that all sequences were aligned to all others. It could happen that one of the entries in the matrix would be '0'. In that case, none of the ith row sub-optimal structures were able to be aligned on none of the jth column's sub-optimals. This situation could be tolerable, as long as all sequences of the same structural class align to one another. Here, we see that MC-Cons made only one structural class, and all of the sequences align to one another.

Just over the Boolean matrix of alignments is the consensus structural assignment, that is, the assignment of a sub-optimal secondary structure to each sequence picked from it's list of sub-optimal structures. Here, our optimal assignment has a score of 683 (Figure 3.4a). for each sequence MC-Cons has assigned a secondary structure. Besides the secondary structures you will find, in parenthesis, the relative rank of the picked structure among it's sub-optimals. For instance, the D_prosaltans secondary structure ranks 34th within it's sub-optimals list. The presence of highly ranked secondary structures in the assignment (1st and 2nd) increases the confidence in the results.

Besides each secondary structure you will also find three web links: `[view]`, `[edit]`, and `[submit]` (see Figure 3.4a):

1. `[view]` will render the secondary structure in a visually appealing format. Pseudoknots are not rendered. The renderer is a slightly modified version distributed with CONTRAfold [6].
2. `[edit]` will bring a web form to tailor MC-Sym's script.
3. `[submit]` will submit the secondary structure to MC-Sym.

Pairwise structural alignments between all sequences are also provided in the `mccons.html` file. It shows which structural features align together. The Figure 3.4b displays a sample of our pairwise alignments.



(a) (b)

Figure 3.4: MC-Cons results. **(a)** The Boolean matrix of alignments, along with the consensus structural assignment. **(b)** A sample of the pairwise structural alignments.

Finally, a clustering of the secondary structures can be obtained:

1. Open the `commands.html` file in your working directory.
2. Click the **CLUSTER** web link.
3. Choose the image format most appropriate for your needs (vector or raster).

The results of the MC-Cons structural clustering is shown in Figure 3.5.

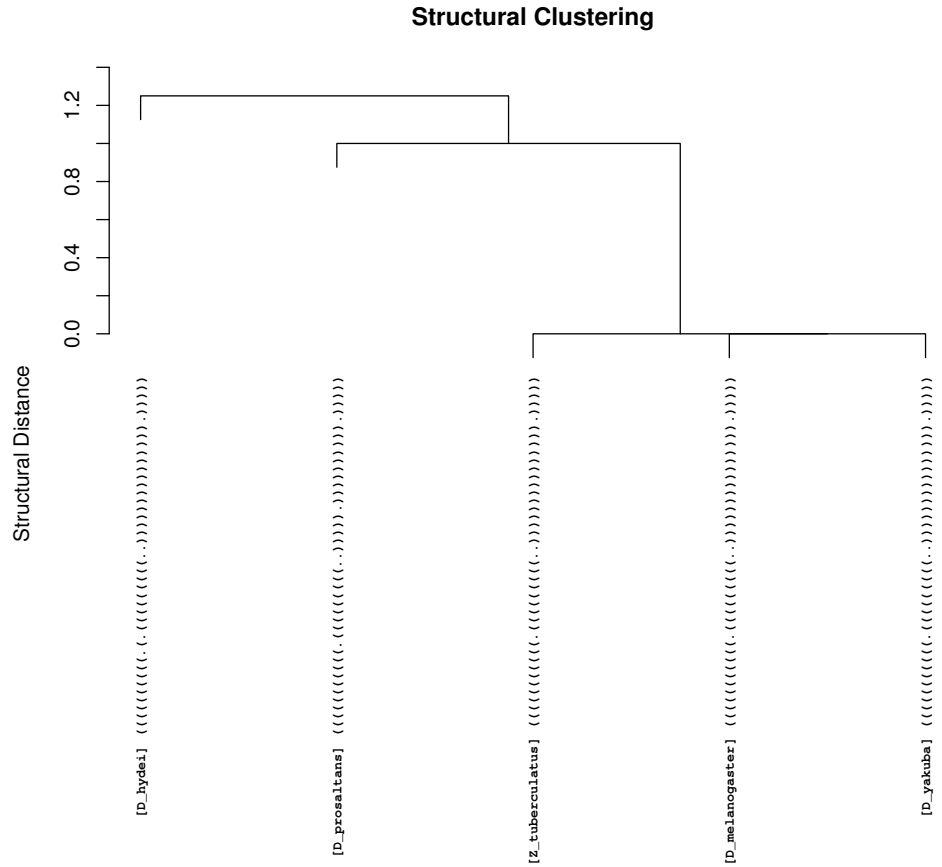


Figure 3.5: MC-Cons clustering of the consensus structural assignment. The clustering is based on the tree-edit distance between the secondary structures, as reported by the computer program RNAdistance of the Vienna package [27]. The clustering is made using the R statistical package [28].

Chapter 4

Using MC-Sym

MC-Sym is a computer program that predicts all-atoms RNA tertiary structure from sequence and secondary structure [29, 30]. It's work flow is depicted in Figure 4.1. A recent upgrade to MC-Sym now makes use of Nucleotide Cyclic Motifs (NCM) [2]. The program is accessible on-line through a web server at this address:

http://www.major.iric.ca/MC-Sym/

As it has recently been found to be pervasively transcribed [31], and the many critical roles performed within the cell, specially in gene maintenance and regulation [32], RNA is more than ever under the spotlight of computational and experimental scrutiny in order to crack it's folding code. Indeed, the 3D structure of a bio-polymer confers it's function [33]. It is without surprise to see an emerging interest in RNA 3D structure prediction [34]. A brief list of on-line resources for tertiary structure prediction can be found in Table 4.1.

Program	URL	Reference
FARNA	https://www.rosettacommons.org/software/index.html	[35]
NAST	https://simtk.org/home/nast	[36]
RNA2D3D	http://www-lmmb.ncifcrf.gov/ bshapiro/software.html	[37]
iFoldRNA	http://troll.med.unc.edu/ifoldrna/	[38]
BARNACLE	http://sourceforge.net/projects/barnacle-rna/	[39]
ASSEMBLE	http://www.bioinformatics.org/assemble/	[40]

Table 4.1: On-line tertiary structure prediction resources. The list is not exhaustive.

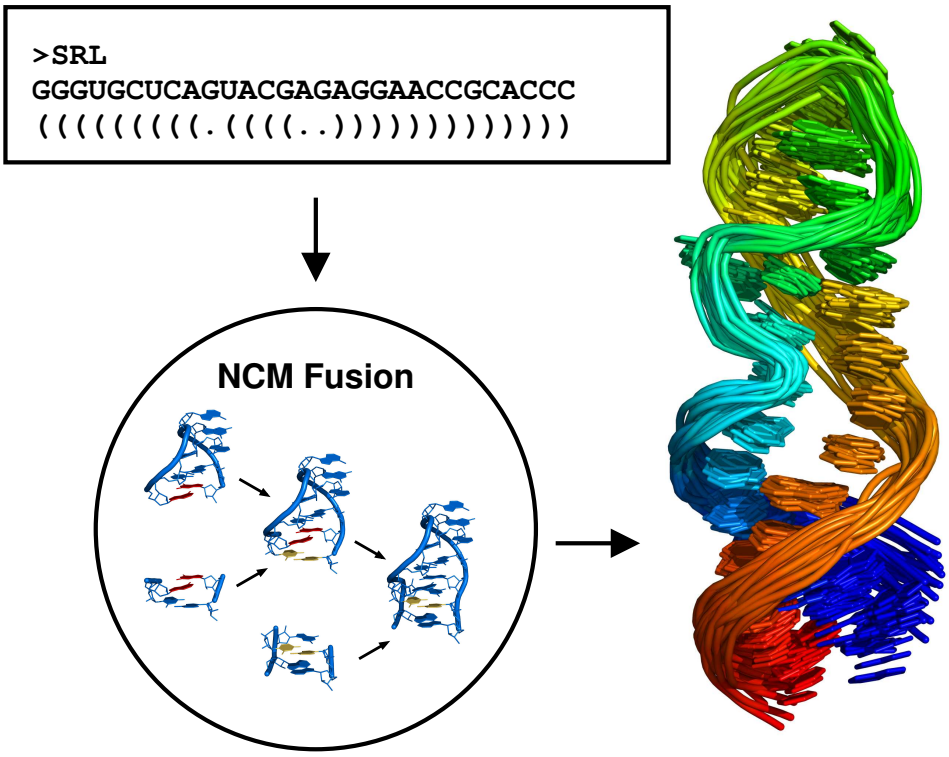


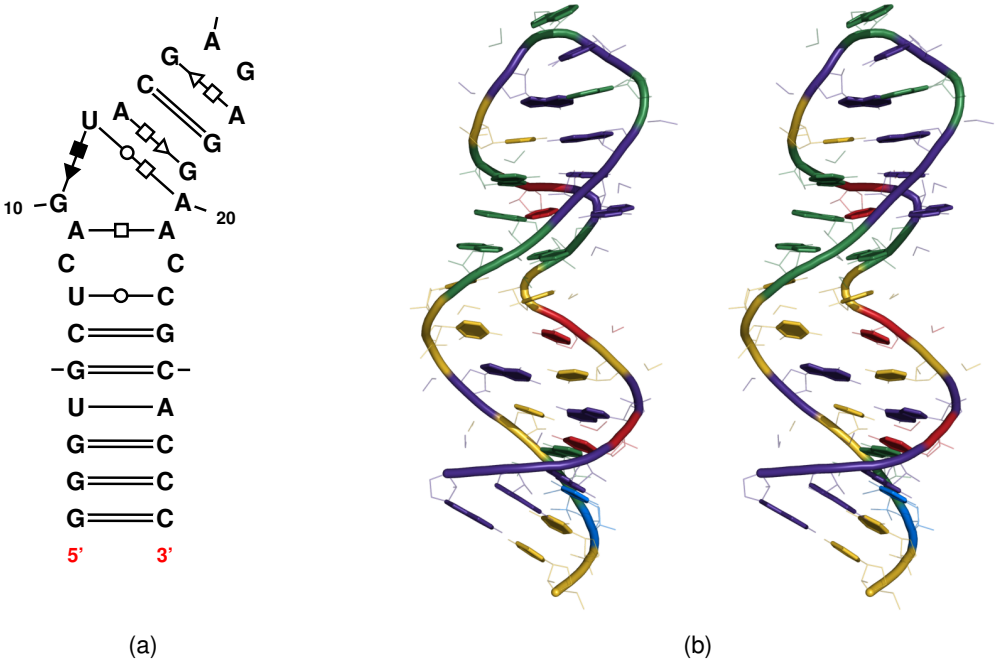
Figure 4.1: MC-Sym work flow. The user specifies the RNA sequence along it's secondary structure as an input to MC-Sym (upper-left box). MC-Sym's Nucleotide Cyclic Motif (NCM) fusion process generates tertiary structures (bottom-left circle). The output of MC-Sym is a list of tertiary structures (right pane).

4.1 A first modeling session

A typical MC-Sym modeling session consists of one or multiple iterations of:

1. Fabricating and editing an MC-Sym script (see section 4.3).
2. Generating 3D structures using MC-Sym.
3. Analyzing the structure set (see chapter 5).

Here, as a first modeling session, we will build 3D models of the sarcin-ricin loop (SRL) from Rat 28S ribosomal RNA [41] (Figure 4.2)(PDB file 430D). We choose this molecule because it highlights the strengths of the MC-Fold and MC-Sym pipeline, as it is able to reproduce all it's nucleotide interactions at atomic precision.



(a) (b)

Figure 4.2: Rat 28S sarcin-ricin loop (PDB file 430D). **(a)** Secondary structure. The image was produced with the on-line computer program MC-Sketch (Mokdad & Major, 2009 unpublished. http://www.major.iric.ca/~mokdada/mcsketch/). **(b)** Stereo-view of the tertiary structure. Nucleotides are colored as follow: green, adenosine; purple, guanosine; red, uracil and yellow, cytosine. Image produced with PyMOL [42].

4.1.1 Via MC-Fold

In this section we will see how to obtain 3D structures from sequence only.

1. Use MC-Fold (chapter 2) to obtain the secondary structure of the RNA molecule named SRL. It's sequence is:

   GGGUGCUCAGUACGAGAGGAACCGCACCC

2. In the results page returned by MC-Fold, browse to the last **MC-Sym-formatted** section.
3. Click the `[submit]` link next to the rank 1 secondary structure. This action will submit this sequence and secondary structure directly to the MC-Sym web server.

4.1.2 Right after a submission

From there, the web server will create a working directory and redirect the web browser to this folder. Figure 4.3a shows such a working directory, whose 10-digit key is Yr2OJN1xBp. This web page can be bookmarked for later use. The URL of this web page can be exchanged between researchers in a collaborative modeling effort, and is accessible across the world via the Internet. The keys are generated randomly from a mixture of letters and numbers, hence offending substrings can appear fortuitously. Directory keys are unique to each submit commands; the key you will obtain will assuredly be different from the one shown here.

After a while refresh the web page of your working folder. MC-Sym will dump there the structures as soon as they are generated. Each model is assigned a sequential number starting from 0001, and is stored in a zipped file in the PDB format. Figure 4.3b shows a working folder in which MC-Sym has deposited some models.

By associating files of the type ".pdb.gz" to a molecular display application (like Rasmol) in your web browser, a simple click on a model will prompt it's download and it's display for an instant visual inspection.

MC-Sym will stop generating models when either one of these two conditions is met: 1. after 30 minutes or 2. when 1000 structures are generated. These instructions are found in the MC-Sym input script.

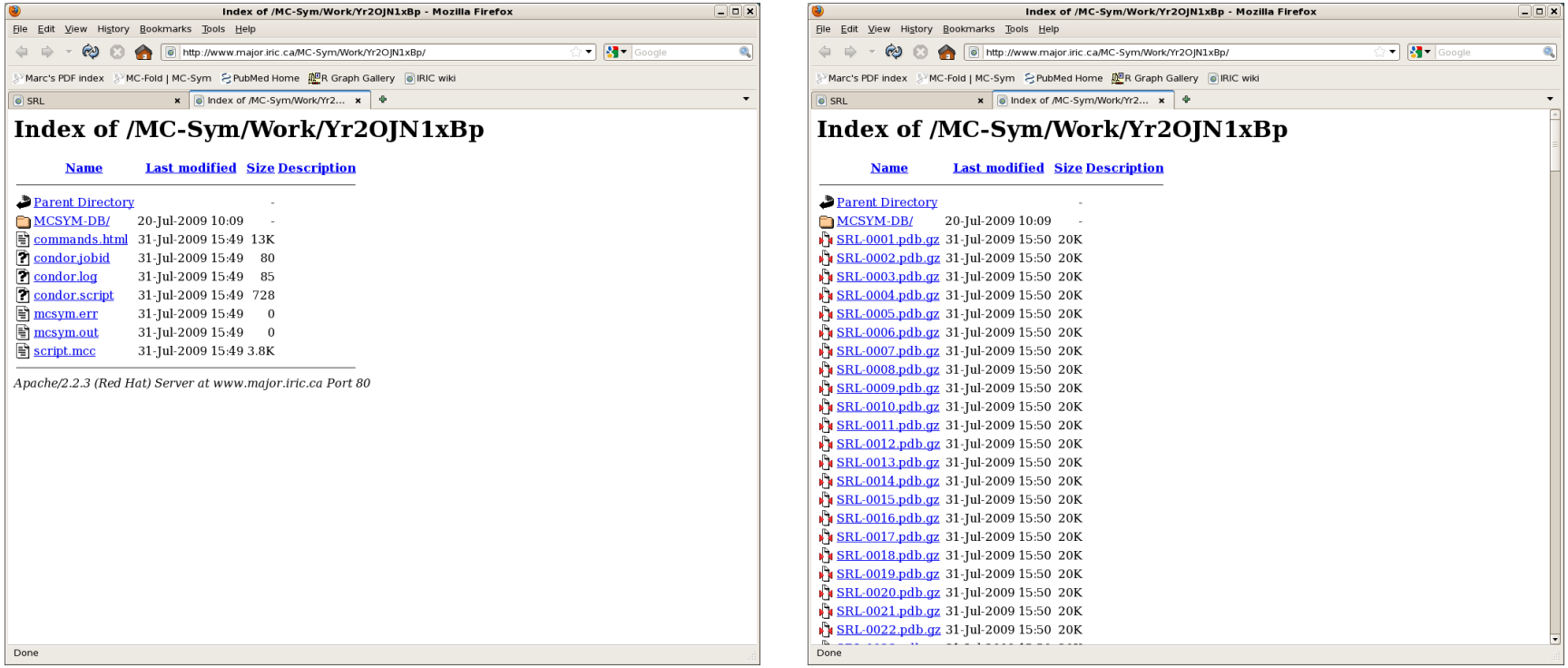


Figure 4.3: MC-Sym results page. **(a)** The page just after a submit command. The MC-Sym input script is `script.mcc`. **(b)** The page after MC-Sym has generated some 3D models. Each model is assigned a sequential number starting from 0001, and is stored in a zipped file in the PDB format.

4.1.3 Via the MC-Sym script generator

In this section we will see how to obtain 3D structures from sequence and from the knowledge of the secondary structure, by making use of the MC-Sym script generator. In order to ease the passage from 2D to 3D, it is imperative that each unpaired nucleotide to model has been considered as a base pair partner. The modeling of classical secondary structures, i.e. structures deprived of in-stem non-canonical base pairs, is much more difficult, because of the many degrees of freedom an unpaired nucleotide has. Hence, we encourage the use of MC-Fold to obtain the secondary structure. Furthermore, hairpin heads are also found structured, and this should be exploited. Consider the T-loop motif in the tRNA (a lonepair triloop [43, 44]), the isolated base pair in the Iron Responsive Element (see [45]) or in the HIV-1 TAR hairpin (see [46]). As for multi-branched loops, there are too few solved in the PDB. However, one might consider studies (like [47]) or databases (like [48]).

1. Browse to the MC-Fold and MC-Sym pipeline page:
   http://www.major.iric.ca/MC-Pipeline/
2. Locate the section **MC-SYM SCRIPT GENERATOR**.
3. Type-in or copy-and-paste your sequence and it's secondary structure. Let's try this:

   ```
   >SRL
   GGGUGCUCAGUACGAGAGGAACCGCACCC
   ((((((((((.((((..))))))))))))))
   ```

   At this point, your web form should look similar to Figure 4.4, in which the input text area has been filled.
4. Click the **Submit** button. This action will submit this sequence and secondary structure directly to the MC-Sym web server.

We invite the reader to visit section 4.1.2 for further instructions.

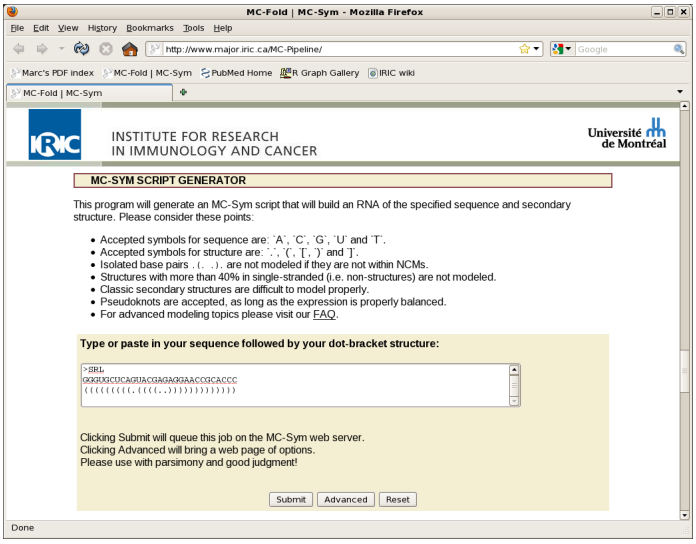


Figure 4.4: MC-Sym script generator.

4.1.4 Polishing the results

Now that MC-Sym has generated some 3D models we need to polish them a bit before doing anything else. Even though MC-Sym uses fragments from the PDB, the backbone of the RNA is entirely reconstructed. Because of this, the backbone is often rendered as discontinuous stretches when the structure is inspected in a molecular viewer. To palliate to this situation, we can minimize the models to an acceptable degree, depending on their sizes and further use.

To polish your 3D structures:

1. Locate the file `commands.html` found in your working directory, and click on it.
2. Browse to the section named **Minimization**.
3. Check one of these two options: **Relieve** or **Refine**.
4. Click the **Minimize** button.

This action will perform a minimization procedure on all of the models. The minimization stop criteria depends on the chosen action: **Relieve** will stop when the force's root-mean-square (RMS) is below 100 Kcal/mol/Å, **Refine** when 5 Kcal/mol/Å. Although the stop criteria is far from the 10^{-3} value required in computational chemistry, it is sufficient for our modeling purposes. The minimization is performed using the Tinker molecular modeling package [49, 50], with a steepest-descent search on the Amber '99 force-field [51]. Typically, one could first **Relieve** the structures, then choose a few models, then **Refine** the chosen models. The **Anneal** option should only be used when a handfull of structures are to be minimized. Furthermore, **Anneal** performs an unrestrained minimization using low-temperature molecular dynamics.

4.1.5 Choosing one solution

After polishing the structures, one can score them in order to rank them by their internal energy. Then, the structure with the best energy could be chosen as to represent the 3D fold of the sequence.

1. Locate the file `commands.html` found in your working directory, and click on it.
2. Browse to the section named **Analysis**.
3. Check the **Score** option.
4. Click the **Analyze** button.

The scoring function makes use of the Amber'99 force-field [51], to evaluate the van der Waals and the electrostatics (H-bonds) between nucleobase atoms (non-bonded interactions only).

4.2 Obtaining an MC-Sym script

There are three ways to obtain an MC-Sym script. The means to obtain a script differ in how one derives the secondary structure:

1. via MC-Fold (see chapter 2). Figure 2.2b shows a list of secondary structures.
   → Click the `[edit]` web link.

2. via MC-Cons (see chapter 3). Figure 3.4a shows a list of secondary structures.
   → Click the `[edit]` web link.

3. via the MC-Sym script generator (see section 4.1.3). Here, the secondary structure is supplied by the user.
   → Click the **Advanced** button.

In all cases, a new web form will appear to custom tailor your MC-Sym script. The web form should look like the one in Figure 4.5. The options are described in the next sections. Set the options to use, then click the **Download** button. Save the script in your computer using the `.mcc` extension.

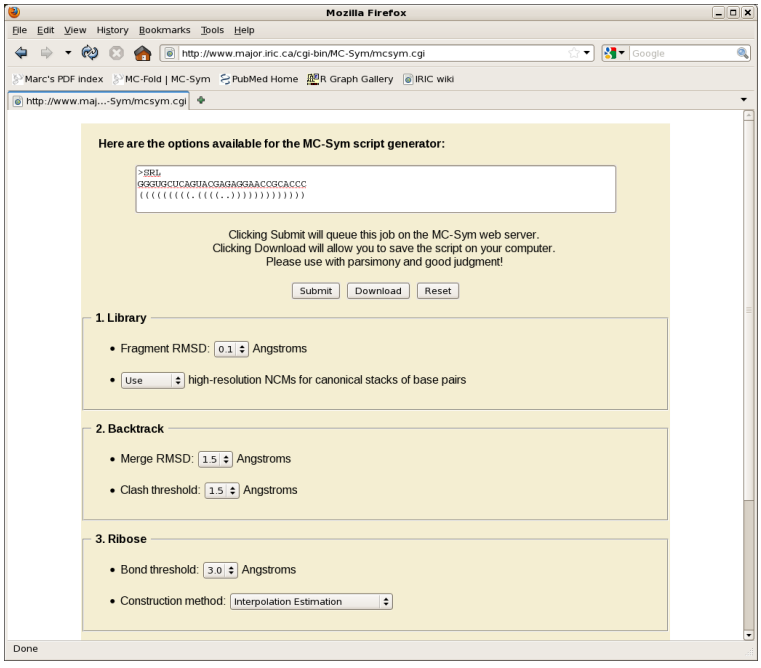


Figure 4.5: MC-Sym script generator form. The form allows to custom tailor an MC-Sym script.

4.2.1 Using the options

Here, we provide short descriptions for each the options of the MC-Sym script generator.

1. **Library**. MC-Sym makes use of fragments from the PDB, and in particular Nucleotide Cyclic Motifs (NCM). These fragments can resemble one another up to a certain point.
   - To remove from MC-Sym's library the fragments which are too similar, set the **Fragment RMSD** threshold. The higher the value, the less number of fragments will be loaded into MC-Sym. The default value will prevent identical fragments to be loaded.
   - *Use* **high-resolution NCMs** for stacks of canonical base pairs. This will limit the conformational search space, and enhance the quality of generated structures. High-resolution NCMs are taken from crystal structures at resolutions better or equal 2.0Å.
2. **Backtrack**. MC-Sym employs the backtracking algorithm to assemble complete 3D structures.
   - The quality of the welding of two consecutive NCM is controlled through the **Merge RMSD** threshold. The default threshold value is such that the merging of similar base pair types at the merged edges are encouraged.
   - The **Clash threshold** is used to control the progression of fragment assembly on the partial structures. It prevents non-bonded atoms to be too close to one another, hence preventing steric clashes.
3. **Ribose**. MC-Sym first positions the nucleobase atoms, then proceeds to an automatic sugar/backbone reconstruction procedure that threads consecutive nucleobases.
   - The **Bond threshold** controls the error on covalent bonds in the backbone. The default value allows for modeling error and for proper backbone conformation recapitulation by a steepest descent minimization.
   - Two **Construction methods** are available. *Interpolation Estimation* (Thibault & Major, unpublished) is fast but generates less precise sugar pucker modes, while *Cyclic Coordinate Minimization* (Lemieux & Major, unpublished) is more precise but is as much as three times slower (because of the multi-step optimization phase).
4. **Explore**. MC-Sym considers these options when exploring the conformational space:
   - Two **Exploration methods** are available. *Probabilistic* allows for back-jumps and random domain assignments. It generally produces solutions faster. It also allows for a thorough sampling of all backtrack variables (library). *Exhaustive* is the classic backtrack algorithm. It will detect construction dead-ends. It is also more efficient at building highly constrained models.
   - The **Model limit** controls the number of models generated. The higher the value the better the choice of structures.
   - The **Time limit** controls the run time of MC-Sym.

- The **Model diversity** threshold controls the resemblance of generated models. If the threshold value is too low then all generated structures are dumped, and the model limit threshold is fast attained. If too high of a value, then very few models are dumped, making MC-Sym explore more of the conformational space. For small hairpins of 30 nucleotides we use 0.5Å. For a 76 nucleotides structure like the tRNA we use between 1 to 3Å(i.e. the larger the structure, the larger should be this value).

4.3 Editing an MC-Sym script

Often, the script generated by the MC-Sym script generator is insufficient or inappropriate. Insufficiency would come, for example, in the call to explicitely line up a base triple, or to restrict two atoms to be within a certain distance. Inapropriateness would come, for example, in the fact that a generated script could contain a dead-end, i.e. a backtrack step which would never be surpassed. The modeling of large, multi-branched, or pseudoknotted RNA structures will require a breakdown of the original script, as it would contain too many backtracking variables (or fragments). A divide-and-conquer approach has a better chance of producing 3D models. Hence the need to edit an MC-Sym script. Figure 4.6 shows the various sections, and their relative ordering, of an MC-Sym script. Each section will be detailed in the following texts.

```
// ==================== Sequence ====================
sequence( r A1 ... )


// ==================== Relation ====================
relation( ... )


// ==================== Library ====================
ncm_01 = library( ... )
ncm_02 = library( ... )
...


// ===================== Backtrack =====================
structure = backtrack
(
    ncm_01
    merge( ncm_02  1.5 )
    ...
)


// ================== Distance Restraints ==================
distance( ... )  // coaxial stacking


// ================== Backtrack Restraints ==================
clash( ... )
backtrack_rst( ... )


// ================== Ribose Restraints ==================
ribose_rst( ... )


// ================== Exploration Initialization =========
explore( ... )
```

Figure 4.6: Sections of an MC-Sym script.

4.3.1 Sequence

The **sequence** section of an MC-Sym script defines the RNA sequence to model. The sequence can feature nucleotides which are not explicitely modeled in 3D, but all modeled nucleotides must be declared in the sequence section. The **sequence** command comprises four parts:

```
sequence( r A1 ACGUACGU )
11111111  2 33 44444444
```

1. The **sequence** command.
2. The **r** indicates a modeling of an RNA sequence.
3. The **A1** indicates the chain ID to model, here A, and the sequential number of the first nucleotide, here 1.
4. Then follows the RNA sequence to model, here **ACGUACGU** .

To model a multiple-chain structure use as many **sequence** commands as there are chains to model. For instance, consider an RNA duplex:

```
sequence( r A1  CCCCAAAA )
sequence( r B11 UUUUGGGG )
```

Here, two chains are modeled: A and B. The first nucleotide in chain A is 1, and in B, 11.

4.3.2 Relation

The **relation** section of a script advises MC-Sym about the relationship between two (or more) nucleotides, and their corresponding relation type. Nucleotides are addressed by their chain ID and their sequential nucleotide number, as defined in the **sequence** section. Nucleotides declared in the **relation** section can be ultimately used in the **backtrack** section. Here are a few examples of nucleotide relations declarations, enclosed in a single **relation** command:

```
relation
(
   A1 A10 { pairing and XIX       } 10
   A5:A7  { adjacent_5p and stack } 10%
)
```

Here, the first line specifies that nucleotides A1 is paired with A10 in a Saenger type XIX base pair (a canonical Watson-Crick G=C). Only ten examplars from the base pairing library are

to be used. The second line indicates that nucleotides A5 through A7 (by the presence of the colon symbol) are to be stacked. Ten percent of the examplars of the stacking library should be considered. For base pairs, the Saenger [52], the Leontis-Westhof [53] and the Lemieux-Major [54] nomenclatures are implemented.

4.3.3 Library

The **library** section of a script declares 3D fragments to be loaded into MC-Sym under a name tag. The structures generated by MC-Sym will be ultimately composed of these fragments. Nucleotide Cyclic Motifs (NCM), and single-stranded stretches from 2 to 4 nucleotides long, can be loaded from a specific folder, rooted at **MCSYM-DB**. Fragments may also be imported from a previous modeling session, to append nucleotides on partial structures. A typical **library** command looks like this:

```
ncm_13 = library(
    pdb( "MCSYM-DB/4/GAGA/*_1.pdb.gz" )
    #1:#4 <- A14:A17
    rmsd( 0.5 sidechain && !( pse || lp || hydrogen ) ) )
```

The first line declares a fragment whose name is `ncm_13`. The second line instructs MC-Sym where and which files to upload. The third line says that the first to the fourth nucleotides in the files are to be mapped to nucleotides 14 through 17 of chain A in our model. Here, notice the relative nucleotide addressing in the uploaded fragments using the pound symbol '#'. This is because GAGA fragments from the PDB have different absolute numbering, depending of which PDB file they come from (and perhaps multiple of these in a same PDB file). If the nucleotides loaded have the same chain ID and sequential numbering as the sequence to be modeled, then this line can be omitted. The last line instructs MC-Sym that the loaded fragments must be at least 0.5Å of RMSD to one another. This is to prevent from loading multiple copies of the same fragment (which can appear in different PDB file names).

Here's how to load partial structures from a previous modeling session, whose directory keys were T2EJQ0uKJD and WlDkaYY4OD:

```
axis_1 = library(
    pdb( "../T2EJQ0uKJD/axis1-*.pdb.gz" ) )
axis_2 = library(
    pdb( "../WlDkaYY4OD/axis2-*.pdb.gz" ) )
```

We have assumed that the imported nucleotides have the proper numbering and chain ID, and that the fragment instances are all distinct in RMSD (because we do not check them against one another).

4.3.4 Backtrack

The **backtrack** section of an MC-Sym script lays out the assembly order of the nucleotides and fragments, just like your new IKEA's furniture building instructions.

The most important rule to remember is that the building order should be a connected spanning tree of the graph in which nodes are the nucleotides or fragments, and edges are the relations between them. That is, a nucleotide or a fragment may be placed at most once, but the same nucleotide or fragment can place many others.

Here are the instructions to add nucleotides and fragments:

- Use **merge** to append a fragment on a partial structure. Fragments are loaded in the **library** section. At least one nucleotide in the merged fragment must have been previously placed. These nucleotides will then serve as anchor points for the welding of the new fragment. The last parameter of a **merge** command is the maximum RMSD for a merge to be successfull. For instance, consider these two fragments:

```
ncm_12 = library(
    pdb( "MCSYM-DB/2_2/CGAG/*.pdb.gz" )
    #1:#2, #3:#4 <- A13:A14, A17:A18 )
ncm_13 = library(
    pdb( "MCSYM-DB/4/GAGA/*_1.pdb.gz" )
    #1:#4 <- A14:A17 )
```

  Then the following **merge** instruction is valid because nucleotides A14 and A17 are common to both fragments:

```
ncm_12
merge( ncm_13 0.5 )
```

- Use **place** to append a fragment on a partial structure. Here, none of the nucleotides of the placed fragment have been previously addressed, so we cannot use **merge**. Therefore, the positioning of the new fragment will be done via a relation (base pairing, stacking, nucleotide adjacency, etc...) with respect to an already placed nucleotide. The relation between the two nucleotides must be declared explicitely in the **relation** section. for instance, consider two helical fragments, axis_1 and axis_2, that are coaxially stacked between nucleotides A39 in axis_1 and A40 in axis_2:

```
relation(
    A39 A40 { adjacent_5p and stack } 25%
)
```

  Then the following **place** instruction is valid because nucleotides A39 and A40 have an explicit (stacking) relation:

```
axis_1
place( A39 A40 axis_2 )
```

- Use the nucleotide place operator **(...)** to position a nucleotide with respect to another. Again, the relation between the two nucleotides must be declared explicitely in the **relation** section. Consider these nucleotide relations:

```
relation(
    A72:A76 { adjacent_5p and stack } 10
)
```

  Then the following **(...)** instructions are valid:

```
( A72 A73 A74 A75 A76 )
```

  Which could be broken down (or any subgrouping of) to, say:

```
( A72 A73 )
( A73 A74 A75 )
( A75 A76 )
```

  Meaning that A72 places A73, then A73 places A74, A74 places A75, etc.

  N.B. The difference between the place operator **(...)** and the **place** instruction is that, even though both addresses two nucleotides and their relation between them, the later will also move all other nucleotides of the same fragment as the nucleotide beeing placed.

4.3.5 Distance

Use as many **distance** commands in an MC-Sym script as imposed by the model. Their use will guide and restrict the conformational space, and hence the time to explore it. Only adjacent nucleotides are under implicit distance constraints (via backbone closure). Look-ahead distance constraints are encouraged to be implemented when modeling large hairpin loops (or bridging two parts of a molecule using single-stranded stretches). Coaxial stacking between two helices can be simulated using a distance constraint. consider these examples:

```
distance( A63:PSY A64:PSY 0.0 5.0 )
distance( A11:C1' A13:C1' 0.0 13.6 )
```

In the first example, the distance between PSY pseudo-atoms (center-of-nucleotides) of A63 and A64 must be found within 5Å. In the second example, a look-ahead distance restraint between the C1' atoms of A11 and A13 is shown.

4.3.6 Others

The **clash**, **backtrack_rst** and **ribose_rst** sections seldom need editing. In the **explore** section, one can change:

- model_limit
- time_limit
- rmsd

All these are described fully in section 4.2.1.

4.4 Submitting an MC-Sym script

Submitting a script to MC-Sym is as easy as 1-2-3:

1. Browse to MC-Sym's input web form:
   http://www.major.iric.ca/MC-Sym/
2. Fill the *file* input field of the "**1. Input File**" section by clicking the **Browse...** button and locating your MC-Sym script.

   At this point, your web form should look similar to Figure 4.7.
3. Click the **Submit** button.

We invite the reader to visit section 4.1.2 for further instructions. Options are further described in the following section.

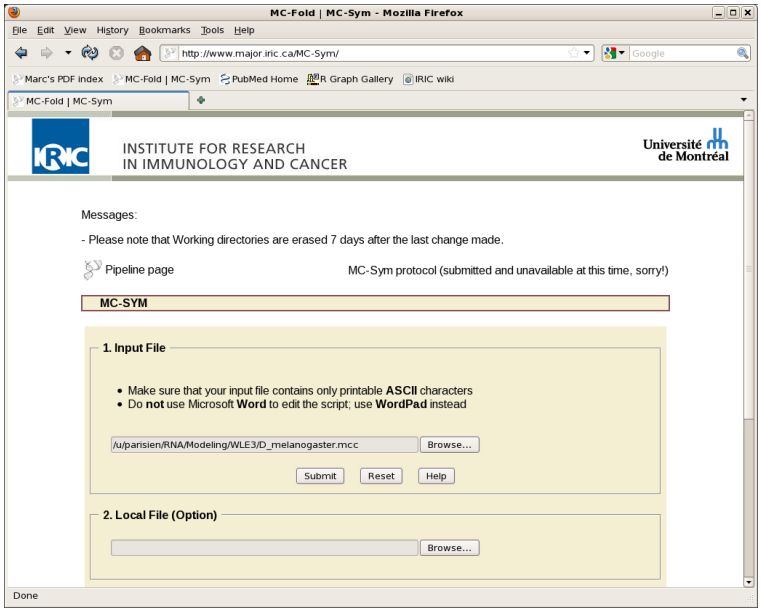


Figure 4.7: MC-Sym web form. Section **1. Input File** has been filed with the file to upload as the MC-Sym script.

4.4.1 Using the options

Here we describe the options of the MC-Sym web input form.

1. **Input File**. Locate your MC-Sym script.
2. **Local File (Option)**. Locate any other file to upload in your working folder. This file can be a reference PDB file to compare the models against with. Or, it can be a PDB file from which parts will be loaded into MC-Sym for modeling. If the **1. Input File** section is left empty, and a **Directory key** is provided, multiple files can be uploaded from this web form into your working folder.
3. **Directory Key (Option)**. Leave this field empty for MC-Sym to automatically create and assign to you a new working folder, typical of a fresh modeling session. Provide a 10-digit key to resume your modeling session in the same working folder. The working folder must have already been created by the same user (identified by it's IP address).
4. **Email Address (Option)**. Provide your email address if you'd like to be informed of MC-Sym's execution completion. This is usefull for long or timeless jobs as the email will provide a reminder of the modeling task. The email will not contain the directory key for which the job has completed, so care must be taken to bookmark the working folders.

4.5 A second modeling session

For our second modeling session we will turn our attention towards human U65 H/ACA small nucleolar RNA and it's substrate rRNA [55] (PDB file 2P89, Figure 4.8b). We will adopt a divide-and-conquer approach, modeling the two main axes separately (Figure 3E of [55], Figure 4.8ac), then assembling these axes in a complete structure. Axis 1 comprises the coaxially stacked stems P1 and P1S, while axis 2 comprises the coaxially stacked P2 and P2S. The coaxial stackings have been deduced from the NMR data.

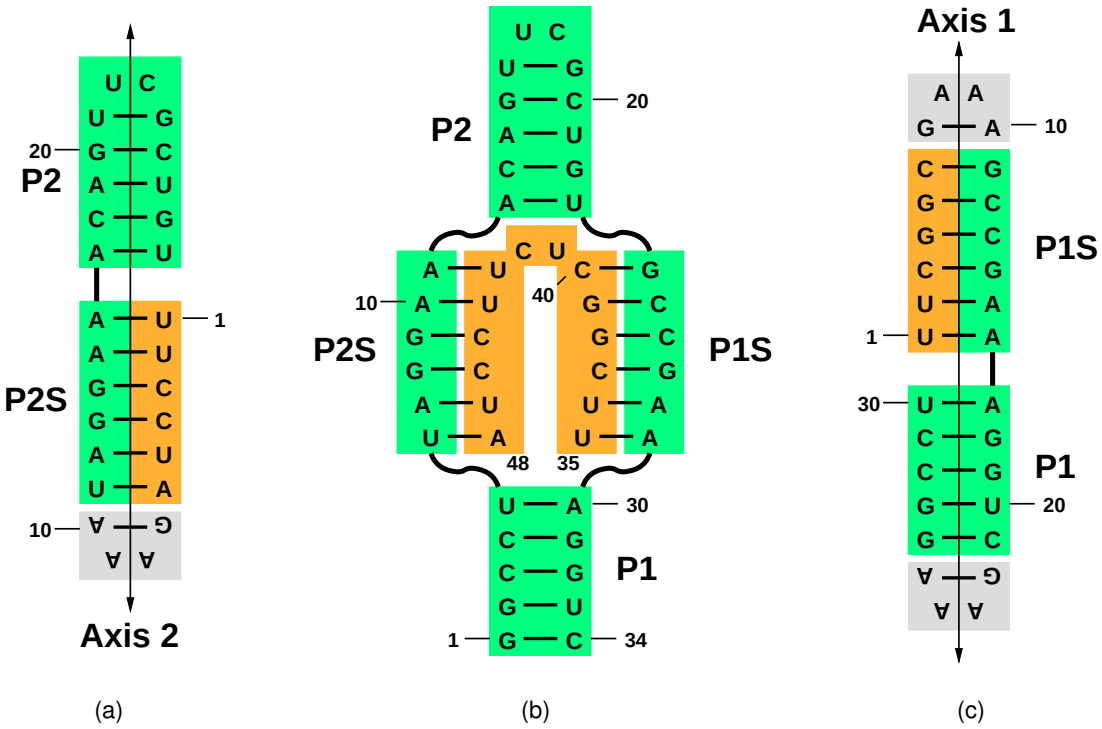


Figure 4.8: Modeling of human U65 H/ACA. **(b)** Human U65 H/ACA (green: chain A) and it's substrate (orange: chain B). Stems P1, P1S, P2 and P2S are identified. **(a)** Modeling of axis 2, which comprises the coaxially stacked stems P2 and P2S. An extra GAAA tetraloop (grey box) serves to make a continuous chain (nts 1-28). **(c)** Modeling of axis 1, which comprises the coaxially stacked stems P1 and P1S. Two extra GAAA tetraloops (grey boxes) serve to make a continuous chain (nts 1-30).

4.5.1 Modeling axis 1

The main difficulty here is to deal with the double-stranded RNA sequence (chains A and B), as the MC-Sym script generator operates with only one RNA strand. The trick to employ is to cap the blunt ends of the double-helices with a GAAA tetraloops (grey boxes in Figures 4.8ac), allowing us to move from one strand to the other with a continuous RNA segment. The extra nucleotides of the tetraloops will not be modeled, and thus the helices will retain their blunt ends. The axes can be loaded afterwards and their nucleotides sequential numbers can be remapped to their original schemes.

Generating the MC-Sym script

To model axis 1 (Figure 4.8c) follow these steps:

1. Browse to the MC-Pipeline web page:
   http://www.major.iric.ca/MC-Pipeline/
2. Locate the section **MC-SYM SCRIPT GENERATOR**.
3. Copy-and-paste the following sequence and secondary structure

```
>axis1
UUCGGCGAAAGCCGAAAGGUCGAAAGGCCU
(((((((..)))))))((((((..))))))
```

4. Click on the **Advanced** button.
5. Locate the **4. Explore** section of the web form.
6. Change **Model limit** to 9999.
7. Change **Time limit** to 1 hour.
8. Change **Model diversity** to 0.5 Angstroms.
9. Click on the **Download** button.

Save the script in your computer with the name `axis1.mcc`. Can you spot the extra GAAA tetraloops in the sequence? Can you also spot the sequence parts of the P1 and the P1S stems (hint: the sequence starts with nucleotide 35)? Use Figure 1B of [55] to help you see clearly what pieces we are modeling.

In the **explore** section, the RMSD is low enough (0.5Å) that the axis will be properly sampled, and 9999 models will be generated within the hour. When modeling, it's better to have too much models than too few; we then have a better choice! Clearly, we can't use them all, and we'll see how to weed out a few of the models.

Editing the MC-Sym script

Now that we have generated an MC-Sym script, we have to edit it to remove the two extra GAAA tetraloops. Open the `axis1.mcc` file with a text editor[1]. Perform the following modifications:

1. In the **Library** section, remove the **library** commands for NCMs: `ncm_06`, `ncm_07`, `ncm_12` and `ncm_13`.
2. In the **Backtrack** section, remove the **merge** commands for NCMs: `ncm_06`, `ncm_07`, `ncm_12` and `ncm_13`.
3. Uncomment the **distance** restraint between nucleotides A17 and A16, to force the coaxial stacking of P1 and P1S.

Generating 3D models

Now, submit your script `axis1.mcc` to the MC-Sym web server (see Section 4.4). Wait until MC-Sym has finished it's job (either the 1 hour time limit expires, or it has generated 9999 structures). Don't forget to take note of your 10-digit working directory key, as we will use it later (our key is utGn9Xa1HM).

[1] for Windows users; use Wordpad instead of Microsoft Word.

4.5.2 Modeling axis 2

Generating the MC-Sym script

We will build axis 2 (Figure 4.8a) using the same scheme as for axis 1; by the use of GAAA tetraloops. The two axes can be assembled in parallel, as one doesn't depend on the other. Here, the sequence and secondary structure for the P2-P2S axis 2 is:

```
>axis2
UUCCUAGAAAUAGGAAACAGUUCGCUGU
(((((((..)))))))((((((..))))))
```

Here, the sequence starts at position 43 of the rRNA substrate. Notice the presence and the position of the UUCG tetraloop. Generate it's script and save it in your computer with the name `axis2.mcc`.

Editing the MC-Sym script

Now that we have generated an MC-Sym script, we have to edit it to remove the extra GAAA tetraloops. Open the `axis2.mcc` file with a text editor. Perform the following modifications:

1. In the **Library** section, remove the **library** commands for NCMs: `ncm_06` and `ncm_07`.
2. In the **Backtrack** section, remove the **merge** commands for NCMs: `ncm_06` and `ncm_07`.
3. Uncomment the **distance** restraint between nucleotides A17 and A16, to force the coaxial stacking of P2 and P2S.

Generating 3D models

Now, submit your script `axis2.mcc` to the MC-Sym web server (see Section 4.4). Wait until MC-Sym has finished it's job (either the 1 hour time limit expires, or it has generated 9999 structures). Don't forget to take note of your 10-digit working directory key, as we will use it later (our key is iCiAecXM8z).

4.5.3 Assembling the axes

preparing the axes

Before assembling the two axes in 3D space, we need to weed out a few of these, such that the final structures will be built from energetically favored axes. Proceed through the following steps, once MC-Sym has finished to generate the 3D structures of both axes:

1. In the working directory of axis 1, locate and open the `commands.html` file.
2. In the **Analysis** section, check the **P-Score** option, then click the **Analyze** button.

We will use the P-Score as a measure of the quality of the generated models. The P-Score evaluates the A-RNA likelyness of the structure (using the positions of the phosphate atoms only). Repeat the same procedure for axis 2 (this can be done in parallel to axis 1). Going on:

1. In the working directory of axis 1, re-locate and open the `commands.html` file.
2. In the **SQL Data Query** section, copy-and-paste this SQL query:
   `DESCRIBE utGn9Xa1HM;`

   Note that your directory key is most likely different from ours; please use yours.
3. Click the **Submit** button.
4. In the **SQL Data Query** section again, copy-and-paste this SQL query:
   SELECT `*` **FROM** `utGn9Xa1HM` **ORDER BY** `PScore` **ASC LIMIT** `1000;`
5. Check the **Delete all models not selected** option.
6. Click the **Submit** button.

Refresh your working directory; it should now contain the top 1000 3D structures. Repeat for axis 2.

Preparing the MC-Sym script

We now need to load the two axes, remap them to their proper nucleotide numbers, and assemble them to make complete models. The whole script, named `assemble.mcc`, is shown in Figure 4.9.

The script has these notable features:

- The script models two RNA sequences, and the nucleotide numbering is to match that found in PDB file 2P89.
- The two axes will be positioned one relative to the other using an adjacency relation between nucleotides 23 and 24 of chain A.
- Axes 1 and 2 are loaded in MC-Sym. Notice how the nucleotides numbers are remapped to fit the new sequence commands. Adapt the script to make MC-Sym load your axes instead (i.e. change the 10-digit directory keys to those you obtained).
- We use tri-nucleotide fragments, named `bridge`, to complete the structure.

Submitting the MC-Sym script

Submit your MC-Sym script, along with the PDB file 2P89 (to compare your models with the solution structure) (see section 4.4). After a few minutes, MC-Sym should have produced a few models. Let MC-Sym run for about 24 hours to obtain a few thousand models. To stop MC-Sym:

1. In the working directory of the complete structure, locate and open the `commands.html` file.
2. In the **Maintenance** section, click the **CANCEL** web link; this will unqueue MC-Sym's job and stop it.

Notice that we haven't used any of the NMR data collected for this complex, except for the coaxial stackings of P1-P1S and of P2-P2S, which could have potentially been predicted from sequence [56] (by maximizing the van der Waals of adjacent purines, like A11-A12, and A29-A30).

4.5.4 Post MC-Sym

Polish your models, by using the **relieve** option in the **Minimization** section, then click the **Minimize** button. Minimizing two thousand structures should take less then 2 hours. Refining all your models at this stage would be a waste of time. Proceed to chapter 5 to analyze your models.

```
// ==================== Sequence ====================
sequence( r A1  GGCCUUAGGAAACAGUUCGCUGUGCCGAAAGGUC )
sequence( r B35 UUCGGCUCUUCCUA                     )
//              12345678901234567890123456789012345
//                       1         2         3

// ==================== Relations ====================
relation
(
    A23 A24 { adjacent_5p && !stack } 95%
)

// ==================== Library ====================
axis_1 = library(
    pdb( "../utGn9Xa1HM/axis1-????.pdb.gz" )
    #1:#6, #7:#17, #18:#22 <- B35:B40, A24:A34, A1:A5 )

axis_2 = library(
    pdb( "../iCiAecXM8z/axis2-????.pdb.gz" )
    #1:#6, #7:#24 <- B43:B48, A6:A23 )

bridge = library(
    pdb( "MCSYM-DB/ss3/UCU/*.pdb.gz" )
    #1:#3 <- B41:B43 )

// ==================== Backtrack ====================
structure = backtrack
(
    axis_1
    place( A24 A23 axis_2 )
    merge( bridge  1.5 )
)

// ==================== Backtrack Restraints ====================
clash
(
    structure
    1.5 !( pse || lp || hydrogen )
)
backtrack_rst
(
    structure
    width_limit  = 25%,
    height_limit = 33%,
    method       = probabilistic
)

// ==================== Ribose Restraints ====================
ribose_rst
(
    structure
    method    = estimate,
    pucker    = C3p_endo,
    glycosyl  = anti,
    threshold = 2.0
)

// ==================== Exploration Initialization =========
explore
(
    structure
    option(
        model_limit = 9999,
        seed        = 3210 )
    rmsd( 0.5 sidechain && !( pse || lp || hydrogen ) )
    pdb( "U65S" zipped )
)
```

Figure 4.9: MC-Sym script of human U65 H/ACA [55]. The script loads two axes that have been previously modeled, and assembles them in a complete structure.

4.6 The conformational search space

In our research group, the RNA modeling problem is formulated as a Constraint Satisfaction Problem (CSP). A CSP is defined from three finite sets [57, 58]: the set of variables $V = \{v_1, v_2, ..., v_n\}$, the set of domains $D = \{d_1, d_2, ..., d_n\}$, and the set of constraints $C = \{c_1, c_2, ..., c_m\}$. In modeling, the variables v_i are the nucleotides, the NCMs, or any other fragments that are uploaded in MC-Sym. Each variable v_i is assigned a domain d_i, which consist of actual 3D instances of the variable. Hence, the third NCM fragment of the ith variable is $d_{i,3}$. The size of a domain is it's number of elements: $|d_i|$. Thus, the conformational search space has a total size of:

$$\prod_{i=1}^{n} |d_i| \tag{4.1}$$

For example, if our modeling task consists of $n = 3$ NCMs (variables), and that we have two, three and four 3D instances for each NCM, then our search space has a size of:

$$\prod_{i}^{n} |d_i| = 2 \times 3 \times 4 = 24 \tag{4.2}$$

That is, we could potentially end up with 24 different solutions. In the CSP formulation, the constraints can always bring down this number of solutions. In practice, an RNA duplex featuring 11 stacked base pairs could be made using 10 NCMs. If each NCM has 100 3D instances from the PDB, that could potentially yield 100^{10} or 10^{19} solutions!

MC-Sym uses the backtracking algorithm to walk the search space. It will fabricate a total number of partial structures:

$$\sum_{i=1}^{n} \prod_{j=1}^{i} |d_j| \tag{4.3}$$

In other words, there are many more sub-structures than the total number of structures! Therefore, when modeling with MC-Sym, is it extremely important to keep the number of variables, and the domain sizes, down. Following our little example:

$$\sum_{i=1}^{n} \prod_{j=1}^{i} |d_j| = (2) + (2 \times 3) + (2 \times 3 \times 4) = 32 \tag{4.4}$$

Here, we will show how to control the domain sizes, using different NCM sets.

4.6.1 Controling the search space

If we can control the number of actual 3D instances found in the PDB for each NCM we model, then we can control the search space size and the time to travel it. Furthermore, some basepairs require more attention than others (a C=G pair is most likely *cis W/W*, but an A=U pair could be *cis W/W* or *trans H/W*), so it would be interesting to sample more certain regions (say assymetric internal loops) than others (say stacks of canonical base pairs).

The stack of basepair, termed the 2_2 NCM, is, by far, the most common block found in RNA structures. It is the founding piece for A-RNA helices. It thus has the most rich set of subsets, to effectively bound the search space.

Here's a list of the NCM sets that can be loaded in MC-Sym, in increasing number of instances. Note that some apply only to the 2_2 NCM:

- ***_t**: the theoretical set. It applies only to 2_2 NCMs which have A=U or G=C base pairs only. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/2_2/UCGA/*_t.pdb.gz" ) ...  )
  ```

  It's purpose is to provide only one 3D instance for that NCM, reducing the conformational search space. Useful to model large RNAs, or to force MC-Sym to sample other parts of the RNA structure. One can use the R20 sets instead (see below); which would also add the G=U base pair in a strict set with few, but "high quality", NCMs. Two problems arise when using the ***_t** sets; 1) consecutive pyrimidines won't be stacked, at least on the 5' strand. 2) base pairs won't be propelled.

- ***Rr***: 2_2 NCMs that come from structures solved using X-ray only, with a resolution better or equal to **r**. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/2_2/GAGA/*R20*.pdb.gz" ) ...  )
  ```

  This set is only available for 2_2 NCMs. Here, **r**=R20 are those NCMs from resolutions better or equal to 2.0A. Available sets are R10, R15, R20, R25, R30 and R35, for resolutions better or equal to, respectively, 1.0, 1.5, 2.0, 2.5, 3.0 and 3.5 Angstroms. Hence, by using R35, you also use those lower than 3.5A, that is R30 downto R10. Theoretical NCMs featuring stacks of Watson-Crick base pairs could be replaced with the R20 sets, for example. Don't expect to find your favorite 2_2 NCM in the R10 set (i.e. solved at a resolution better than 1.0 Angstrom)!

- ***R***: 2_2 NCMs that come from structures solved using X-ray only, regardless of resolution. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/2_2/GAGA/*R*.pdb.gz" ) ...  )
  ```

- ***_1**: observed set, found in the PDB with the actual sequence. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/4/GGAA/*_1.pdb.gz" ) ...  )
  ```

  It's purpose is to provide 3D instances for that NCM that can be found in the PDB with the actual modeled sequence. Useful to model parts of an RNA structure which are known to adopt particular folds, like the GNRA- or UNCG-tetraloops, or base pair steps with proper and suited helical parameters [59, 60, 61, 62].

- ***_x**: artificial set, built from backbone templates. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/4_2/GUUCGC/*_x.pdb.gz" ) ...  )
  ```

  It's purpose is to provide 3D instances for NCMs with few or none examplar in the PDB. They are built from a database of backbone templates onto which the flanking base pairs were fitted, and any other nucleobases mutated. Useful to model any NCMs which cannot be found in the PDB. As more nucleotides are implicated in the NCM, the number of backbone templates diminishes rapidly. Hence, a user should opt for other modeling options, like modeling internal loops with single-stranded regions. Assymetrical loops can also be modeled from a motif search in the PDB, as K-turns [63] or C-loops [64]. The artificial set construction is described in details here [2].

- *: a combination of all other sets. For example:

  ```
  ncm_01 = library( pdb( "MCSYM-DB/4/UACG/*.pdb.gz" ) ...  )
  ```

  However, MC-Sym will load in that order: 1) the theoretical set (if any), 2) the observed set (if any), and 3) the artificial set.

The MC-Sym automatic script generator ponders with these different sets depending on the type of NCM and it's sequence.

4.6.2 Debugging a dead-end

Sometimes, MC-Sym is prevented from assembling complete structures because of an inapropriate suite of NCM domain sets. Imagine trying to model this duplex:

```
5′- C  AC -3′
3′- GCCUG -5′
```

Using these two NCM domains:

```
 ncm_01 = library( pdb( "MCSYM-DB/2_4/CAUCCG/*.pdb.gz" ) ...  )
ncm_02 = library( pdb( "MCSYM-DB/2_2/ACGU/*R20*.pdb.gz" ) ...  )
```

A dead-end is created by using an NCM set wich is too strict and cannot weld on any instances of the previous NCM set. Here the use of the 2_2 ***R20*** set makes the dead-end. By increasing the number of instances in ncm_02, we increase the chance of a successful welding. Thus, one could use the next best set to ***R20***:

```
ncm_02 = library( pdb( "MCSYM-DB/2_2/ACGU/*_1.pdb.gz" ) ...  )
```

Which, instead of considering instances from X-ray solved structures, we will consider any NCM observed in the PDB.

Chapter 5

Analyzing MC-Sym's results

This chapter is dedicated to the analysis of MC-Sym's results. When modeling an RNA one has to evaluate the fitness of a model. The following sections will describe in details why and how to assess a model, and in particular with respect to experimental probing data.

5.1 Comparing with a solution structure

The only way to asses and parameterize a modeling engine is by comparing it's results to solved structures. The *de facto* standard in structural comparison is the root mean squared deviation (RMSD) [65]. However, other measures were devised and are equally interesting:

- **GDT-TS** (Global Distance Test) [66, 67].
  Number of atoms in a model within 1, 2, 4 and 8Å from their respective positions in the solution structure, once optimally aligned. That number is then divided by four times the total number of atoms in the model. Hence, the GDT-TS score is a fraction ranging from zero (complete modeling failure) to one (complete modeling success). Values above 0.5 could be considered as modeling feats (because, at the worst, the two upper shells at 4 and 8Å would now be filled).
- **INF** (Interaction Network Fidelity) [68].
  RNA molecules most likely undergo conformational changes under Brownian motion, hence affecting the RMSD. However, the network of hydrogen bonds between nucleobases is perhaps more robust, and a better measure of RNA modeling success. The INF is simply the Matthews correlation coeficient [69, 70] between the nucleobase interactions (base pairing and stacking) in the model compared to the solution structure. Values range between zero (complete modeling failure) to one (complete modeling success).
- **DI** (Deformation Index) [68].
  This is simply the RMSD corrected by the INF: DI = RMSD / INF. In other words, RMSD is meaningless if the model does not reproduce the nucleobase interactions.

5.1.1 Uploading a solution structure

There are two ways to supply a solution structure against which the models will be compared to, depending on whether the models are already generated or not.

1. By uploading the reference structure along with the MC-Sym script (see sections 4.4 and 4.4.1): when submitting an MC-Sym script to the web server, provide at the same time the reference structure in the **2. Local File (Option)** section.
2. By uploading the reference into a specified working directory key (see section 4.4.1): when supplying a working directory key in the **3. Directory Key (Option)** section, provide at the same time the reference structure in the **2. Local File (Option)** section, and leave empty the **1. Input File** section.

It is best if the solution structure:

- does not contain any modified nucleotides (i.e. HETATM rows).
- has the same chain ID and sequential nucleotide numbering as the models.

Hence, the solution structure may need extra attention in order to provide for the structural comparison.

5.1.2 Using the solution structure

When uploading a reference structure, the MC-Sym web server will update the `commands.html` file in your working directory to include new structural comparison commands and options:

- An **RMSD** section: simply click the **RMSD** button.
- An **INF** option in the **Analysis** section.
- A **GDT-TS** option in the **Analysis** section.

Once the RMSD and the INF have been computed for all models, the Deformation Index (**DI**) can be obtained via an SQL data query:

1. Locate and open the file `commands.html` in your working directory.
2. Browse to the **SQL Data Query** section.
3. Copy-and-paste the following SQL statement, by adapting the 10-digit directory key to your needs:
   SELECT `*, RMSD/'INF-ALL' AS DI FROM Yr2OJN1xBp ORDER BY DI;`
4. Click the **Submit** button.

5.2 Choosing structures

Generating a structural ensemble for an RNA molecule is one thing, choosing one or a few structures from the pool as the representative fold is another, specially in the absence of any experimental probing data. Unfortunately, energy evaluation from a molecular mechanics force-field (MMFF) will not uncover the best fold, because 1. the force-field is too "local" in scope (doesn't see the big picture), and 2. the MC-Sym models, even refined, still have defects which prevent them to be properly scored by an MMFF. Hence, we must rely on a more *ad hoc* selection procedure.

It is extremely informative to compare how RNA geometrical data behaves against the RMSD for a given decoy set and it's reference structure. Consider the pool of 3D structures generated in section 4.5, for the human H/ACA U65 and it's substrate.

5.2.1 Base entropy

An interesting observation made by Laederach and co-workers is that "it is often possible to find an orientation such that only the edges of most bases are visible" [71]. They also provided an operational definition of the base entropy, implemented here in the pipeline. Hence, we can evaluate the bipolarity or the coplanarity of bases within any 3D model:

1. Open the `commands.html` file in your working directory.
2. Choose the **Entropy** option in the **Analysis** section, then click on the **Analyze** button.

Once the entropy analysis is done, we will look at how it behaves compared to the RMSD (which must have previously been computed, see section 5.1):

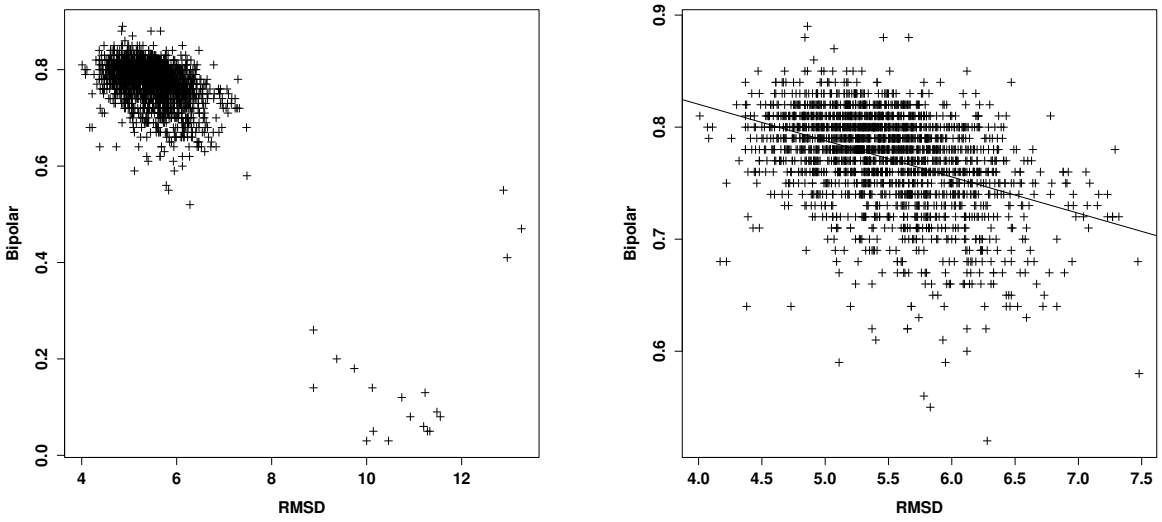
1. Open the `commands.html` file in your working directory, again.
2. In the **SQL Data Query** section, copy-and-paste the following SQL command. Make sure to adapt your 10-digit directory key:
   SELECT `RMSD, Bipolar` **FROM** `qQHxGzuIff;`
3. Check the **Show results in 2D graphic** option.
4. Click the **Submit** button.

Here, you should see a plot of Bipolar against the RMSD, similar to Figure 5.1a. Let's zoom into the portion of the graph where the RMSD is lower than 8Å, and add the correlation curve:

1. In the **SQL Data Query** section, copy-and-paste the following SQL command. Make sure to adapt your 10-digit directory key:
   SELECT `RMSD, Bipolar` **FROM** `qQHxGzuIff` **WHERE** `(RMSD<8);`

2. Check the **Show results in 2D graphic** option.
3. Check the **Show also the correlation** option.
4. Click the **Submit** button.

Now, we see that bipolarity doesn't correlate well with RMSD (we obtain an adjusted R-squared of 0.16, thus a Pearson's correlation coefficient of $P = \sqrt{0.16} = 0.4$) (see Figure 5.1b), but low RMSD structures have a high bipolarity content (say, above 0.8).



(a) (b)

Figure 5.1: Bipolarity against RMSD for human H/ACA MC-Sym models. **(a)** The plot obtained from all models. **(b)** The plot obtained from the region where the RMSDs are lower than 8Å. The oblique line is a linear best-fit curve of the points.

5.2.2 Radius of gyration

The radius of gyration of a structure is another geometrical variable to watch. Some favor compact RNA models by penalyzing proportionaly to the radius of gyration [35]. Since RNA obeys the Flory theory, the radius of gyration grows as $5.5N^{1/3}$, where N is the number of nucleotides in the sequence [72]. In our case, N=48, so we expect to have a radius of gyration of about 20Å. To measure the radius of gyration of models:

1. Open the `commands.html` file in your working directory.
2. In the **Global Geometry** section, choose the **Radius of gyration** option.
3. Enter 0 (zero) as the **Target value**.
4. Click the **Submit** button.

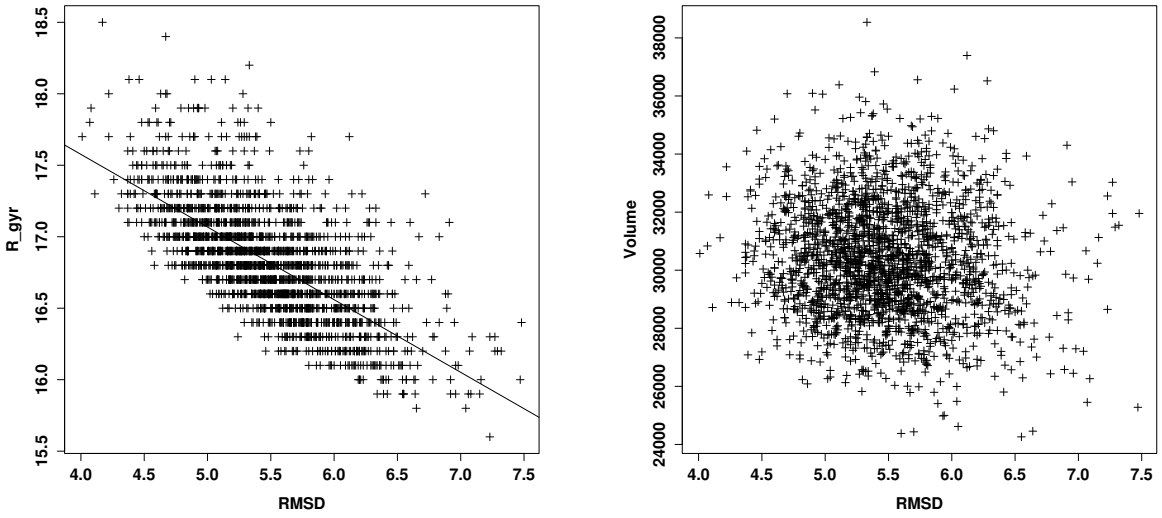
Once the radius of gyration are computed:

1. In the **SQL Data Query** section, copy-and-paste the following SQL command. Make sure to adapt your 10-digit directory key:
   SELECT `RMSD, R_gyr` **FROM** `qQHxGzuIff` **WHERE** `(RMSD<8);`
2. Check the **Show results in 2D graphic** option.
3. Check the **Show also the correlation** option.
4. Click the **Submit** button.

Here, the radius of gyration anti-correlates ($P \approx 0.7$) with the RMSD (see Figure 5.2a). The values obtained are slightly lower than the 20Å expected, this is in no way critical.

5.2.3 Volume

The volume of an RNA, approximated by an ellipsoid, may also be of interest. In our case, there is no relation between the volume and the RMSD (see Figure 5.2b).



(a) (b)

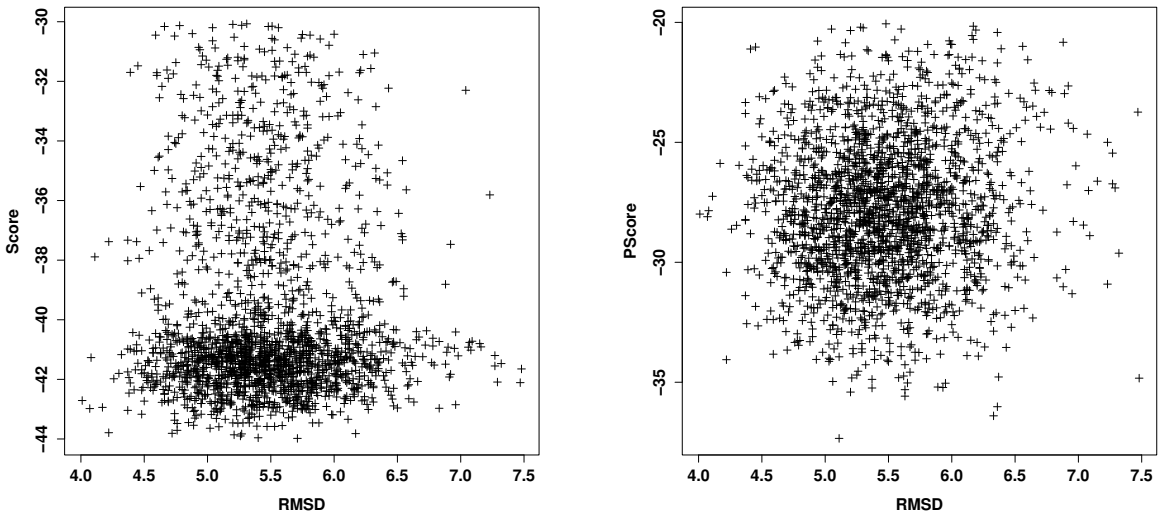
Figure 5.2: Various geometrical data against RMSD for human H/ACA MC-Sym models **(a)** Radius of gyration against RMSD. The oblique line is a linear best-fit curve of the points. **(b)** Volume against RMSD.

5.2.4 Internal energy

Finally, one could resort to the internal energy of a model to assess it's quality. In the pipeline, we offer two measures: Score and P-Score:

1. Open the `commands.html` file in your working directory.
2. In the **Analysis** section, check the **Score** and the **P-Score** options.
3. Click the **Analyze** button.

Score makes use of the non-bonded energy evaluation of the Amber '99 force-field [51]. It comprises mainly of the van der Waals and a distance-dependant Coulomb terms, to simulate implicitely the solvent. It applies only between the nucleobase atoms, such that Score can be used even on un-refined MC-Sym models. P-Score is entirely knowledge-based and has been derived from a non-redundant subset of the PDB. It measures the valence and torsion angles between three and four consecutive phosphate atoms, respectively. It is similar in spirit to those derived elsewhere (see [36]). The torsion term is novel, and it's goal is to palliate to the locality of the measurements. There is no trend between internal energy and RMSD, as shown in Figure 5.3.



(a) (b)

Figure 5.3: Internal energy data against RMSD for human H/ACA MC-Sym models **(a)** Score against RMSD. **(b)** P-Score against RMSD.

5.2.5 A final round

All the geometrical parameters and internal energy can be blended into a single structure selection statement:

1. In the **SQL Data Query** section, copy-and-paste the following SQL command. Make sure to adapt your 10-digit directory key:
   SELECT * **FROM** qQHxGzuIff **WHERE** (R_gyr > 17.0) **AND** (Bipolar > 0.8) **AND** (Score < −40) **AND** (PScore < −30);
2. Click the **Submit** button.

This SQL query will select models which conforms to all of the **WHERE** clauses. The number of structures selected is still high enough to forbid a manual inspection. By clustering these models by structural similarity, we can bring down the number of representatives:

1. Re-open the `commands.html` file.
2. In the **K-Means Clustering** section, choose *SQL select* as the **Number of models**.
3. Click the **Submit** button.

We have now partitioned all the structures into five buckets; then, we could ask for the best Score model per cluster:

1. In the **SQL Data Query** section, copy-and-paste the following SQL command. Make sure to adapt your 10-digit directory key:
   SELECT * **FROM** qQHxGzuIff Table_1
   INNER JOIN
   (**SELECT MIN**(Score) **AS** MinScore, Cluster **FROM** qQHxGzuIff
   GROUP BY Cluster) Table_2
   ON (Table_1.Cluster = Table_2.Cluster)
   AND (Table_1.Score = Table_2.MinScore);
2. Click the **Submit** button.

Figure 5.4 shows the selected model of the most populous cluster (cluster 1) optimally superimposed on the NMR solution structure. The RMSD is 4.8Å, all heavy atoms. The Interaction Network Fidelity value is 84.9% (TP=44, FP=17, FN=0). A closer inspection of the differences between our model and the one published reveals that our model has 17 more inter-nucleobase stacking interactions than the reference. This also explains why our RMSD is that high.

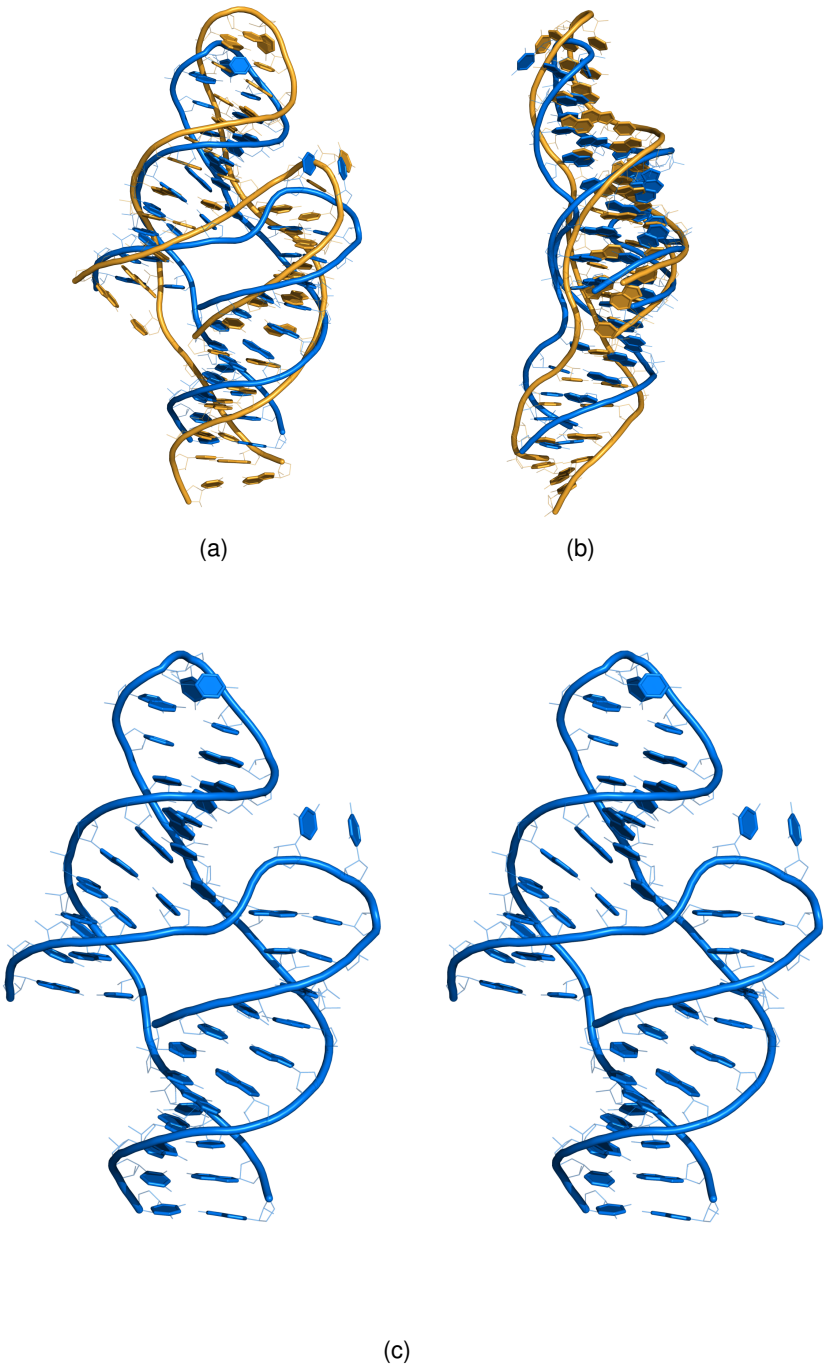


Figure 5.4: A model of human U65 H/ACA. **(a)** and **(b)** The MC-Sym model (blue) is optimally superimposed on the NMR solution model (yellow; PDB code 2P89). **(c)** Stereo-view of the MC-Sym model.

5.2.6 Distance-based restraints

The most useful restraint types are distance-based, because they only necessitate cartesian distances between pairs of atoms, and is easily computed. These can be used in the MC-Sym script, as hard constraints (i.e. cannot be violated), or as soft constraints in a post-processing step. There are multiple experimental and computational methods to obtain distance restraints. Notably:

- **Covariation**, which provide long-range distance constraints, and possibly base pair type inference (see, for instance [73, 74, 75, 76]).
- **Nuclear Magnetic Resonance**, which provide tight distance constraints between pairs of atoms [77, 78].
- **EDTA-FE(II)**, using a modified uridine nucleotide. Provides distance restraints within 25Å [79].
- **MPE-FE(II)**, using an helical motif and an intercalation agent. Provides distance restraints within 25Å [80].
- **MOHCA**, a parralel version of EDTA-based cleavage. Also provides distance restraints within 25Å [81].

Models can be evaluated for their fit to distance restraints in the **Distance Restraints** section of the `commands.html` file. Distance restraints file formats are also given for each type of experiment.

5.2.7 Inside and outside

Hydroxyl radical (OH) footprinting can be used to define the inside and the ouside of an RNA molecule, by unveiling the backbone positions which are protected from solvent [82, 83]. The computation is rather intensive, because of the need of the solvent-accessible surface area of backbone proton atoms. Furthermore, the correlation between the OH profile and solved structures can be low (about 0.1 to 0.6 for the P4-P6 fragment of group I Intron [84, 85, 81, 36]). Models can be compared to an OH profile in the **Hydroxyl Radical Footprinting** section of the `commands.html` file.

5.2.8 Rigid-body docking

From small-angle X-ray scattering (SAXS) or from cryo-electron microscopy (cryo-EM), one can obtain a beads model or a molecular envelope, which approximates the 3D fold. Then, 3D models can be fitted into the beads model using a rigid-body docking procedure, as implemented in the Situs suite of programs [86]. Models can be fitted into beads models in the **Molecular Envelope** section of the `commands.html` file.

Chapter 6

Other tools

In this chapter we will describe how to use other tools that are part of the MC-Fold and MC-Sym pipeline.

6.1 Rendering secondary structures

Secondary structures are best grasped by humans in the form of a figure, instead of the dot-bracket notation. If you have a sequence and it's secondary structure, then it's possible to generate the figure:

1. Browse to the MC-Pipeline web page:
   http://www.major.iric.ca/MC-Pipeline/
2. Locate the section **DOT-BRACKET RENDERING**.
3. Fill the **dot-bracket** text area.
4. Adjust the index of the first nucleotide.
5. Click the **Submit** button.

The secondary structure renderer is based on the one distributed with the CONTRAfold computer program [6]. Note that the renderer doesn't make proper layouts for pseudoknotted structures. For that, one can use the PseudoViewer online web server [87]. For rendering that makes use of the Leontis-Westhof nomenclature [53], please visit:
http://www.major.iric.ca/~mokdada/mcsketch/.

6.2 Mutating nucleobases

If you already have a 3D structure, and you'd like to alter it's sequence or replace non-canonical nucleobases by standard ones, then use our mutation tool, found here:

1. Browse to the MC-Pipeline web page:
   http://www.major.iric.ca/MC-Pipeline/
2. Locate the section **RNA SEQUENCE MUTATION**.
3. Fill the **Structure** and **Sequence** input fields.
4. Click the **Submit** button.

This tool mutates the nucleobase atoms only. Hence, it doesn't guarantee to form proper base pairs: for instance, if a G=C base pair is mutated to an A=A following the sequence mutation, then the A=A base pair will most likely be crooked, because the base pair has not been replaced by another one, but instead by two point mutations. To mutate a base pair, consider section 6.3.

6.3 Mutating a base pair

To mutate a single base pair, make an MC-Sym script which will:

1. Load all nucleotides between the two ones of the pair into `fragment_A`.
2. Load all nucleotides 5' of the 5'-paired and 3' of the 3'-paired into `fragment_B`.
3. Load two NCMs; one that bridges from `fragment_A` to the base pair, the other one from the base pair to `fragment_B`.
4. Assemble the fragments and dump them all.

Figure 6.1 shows an MC-Sym script which will mutate the G3=C27 base pair of the Rat 28S sarcin-ricin loop (PDB file 430D; Figure 4.2) to an A=U base pair. Here are the steps to proceed with the example:

1. Submit the script shown in Figure 6.1, along with PDB file 430D (see sections 4.4 and 5.1.1).
2. When MC-Sym has finished producing the models, open the `commands.html` file in your working directory.
3. Choose the **Refine** option in the **Minimization** section, then click on the **Minimize** button.
4. When the minimization phase is finished, return to the `commands.html` file in your working directory.
5. Check the **Score** option in the **Analysis** section, then click on the **Analyze** button.
6. Choose your model among the top best scores.

Here are some key points to note in the script file (Figure 6.1):

- The sequence is the one which includes the mutated base pair.
- We modeled the new base pair using NCMs.
- We allowed the MC-Sym script to run a very short time: 2 minutes only.
- We allowed MC-Sym to produce solutions which are as close to one another as 0.1Å.

```
// ==================== Sequence ====================
sequence( r A1  GGAUGCUCAGUACGAGAGGAACCGCAUCC )
//              ((((((((((.((((..))))))))))))))
//              AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//              123456789012345678901234567890
//                       1         2

// ==================== Library ====================
fragment_A = library(
    pdb( "430D.pdb" )
    A4:A26 <- A4:A26 )

fragment_B = library(
    pdb( "430D.pdb" )
    A1:A2, A28:A29 <- A1:A2, A28:A29 )

ncm_01 = library(
    pdb( "MCSYM-DB/2_2/AUAU/*R20*.pdb.gz" )
    #1:#2, #3:#4 <- A3:A4, A26:A27 )

ncm_02 = library(
    pdb( "MCSYM-DB/2_2/GAUC/*R20*.pdb.gz" )
    #1:#2, #3:#4 <- A2:A3, A27:A28 )

// ==================== Backtrack ====================
structure = backtrack
(
    fragment_A
    merge( ncm_01      1.5 )
    merge( ncm_02      1.5 )
    merge( fragment_B  1.5 )
)

// ==================== Backtrack Restraints ====================
clash
(
    structure
    1.5 !( pse || lp || hydrogen )
)
backtrack_rst
(
    structure
    width_limit  = 25%,
    height_limit = 33%,
    method       = probabilistic
)

// ==================== Ribose Restraints ====================
ribose_rst
(
    structure
    method    = estimate,
    pucker    = C3p_endo,
    glycosyl  = anti,
    threshold = 3.0
)

// ==================== Exploration Initialization =========
explore
(
    structure
    option(
        model_limit = 9999,
        time_limit  = 2m,
        seed        = 3210 )
    rmsd( 0.1 sidechain && !( pse || lp || hydrogen ) )
    pdb( "SRLau" zipped )
)
```

Figure 6.1: MC-Sym script for a base pair mutation. The mutation will replace the G3=C27 base pair to an A=U base pair, in the PDB file 430D.

6.4 Using MC-Search

MC-Search is a computer program that takes for input a motif descriptor and searches for it in PDB files [88]. As RNAs are made up of modular blocks [89, 90, 91], we can seek and use them when modeling in 3D. Consider the following *tricky* sequence and the top two MC-Fold solutions:

```
>tricky
AAUGUUACUCUGGAAUGACUAUAAACAUU
(((((((((((..)))))))...)))))))
((((((((((((..))))))))...))))))
```

We can see here that MC-Fold is fooled by the GGAA sequence and folds it into a tetraloop. The presence of two Uracils flanking the GGAA makes it not a good candidate for a GNRA tetraloop. Instead, the hairpin head is, most likely, a lonepair triloop (LPT [43, 44]), because the sequence has already been observed to fold in a LPT. Using a structural mask to force the lonepair triloop motif, we can now refold the sequence with MC-Fold:

```
>tricky
AAUGUUACUCUGGAAUGACUAUAAACAUU
*********((...).)*********** LPT mask
((((((((((...).)))))...))))))
((((((((((...).))))))...))))))
|    |     |     |     |     |
1    5    10    15    20    25
```

Still here, there's an ambiguity about the 3'-bulge. If we allow to put two consecutive Adenosines, A24 and A25, into an A-platform motif [92], then the two other Adenosines, A21 and A23, can make base pairs with U6 and U5, respectively. Here, contrary to the LPT, we do not have any sequence or structural evidence for our modeling hypothesis. Only further structure probing will (in)validate this hypothesis. The final secondary structure is:

```
>tricky
AAUGUUACUCUGGAAUGACUAUAAACAUU
((((((((((...).))))).)..))))
```

We will see how to use MC-Search to fetch from the PDB actual 3D instances for our LPT and for our A-platform.

6.4.1 MC-Search; lonepair triloop

Devising the motif

Before using MC-Search, one needs to fabricate the motif descriptor to look for. For our lonepair triloop, the descriptor will be:

```
sequence( RNA A1 NUNNNANN )
relation(
A1 A8 { pairing }
A2 A6 { pairing }
)
```

LPT are characterized by 1. the UNNNA sequence, 2. the lonepair UA and 3. a 3'-bulge just after the A, allowing for a UA_handle [93]. Save the descriptor in your computer in a file called `lpt.mcs`.

Submitting the motif

Once the motif has been described in the MC-Search language, we can now submit the motif to the MC-Search web server:

1. Browse to the MC-Search web page:
   http://www.major.iric.ca/MC-Search/
2. Click the **Browse...** button and locate the file `lpt.mcs`.
3. Click the **Submit** button.

This will launch MC-Search on your descriptor; it will redirect your web browser to a working directory. Please take note of your 10-digits working directory key, as it will be needed afterwards. Our key is TVB24yoMdS. The file `results.pdb` will contain all the 3D instances that correspond to the searched motif.

Adapting the sequence

We now need to adapt the sequences in our 3D motifs to the one we are modeling. Nucleobases 3 to 5 and 7 need to be mutated, as well as the first base pair (the second base pair, the UA_handle, is already tailored for our needs because it has been explicitely looked for in the MC-Search script). To do so, perform these steps:

1. Locate and click on the `commands.html` link. This will open a web page which will allow us to make sequence mutations on our motifs.
2. Copy-and-paste the following lines in the **Enter your sequence mutation** input box:

```
mutate( nucleobase A3 G )
mutate( nucleobase A4 G )
mutate( nucleobase A5 A )
mutate( nucleobase A7 U )
mutate( basepair A1 A8 C G )
```

3. Click the **Submit** button.

Each sequence mutation command is performed sequentially. At the bottom of the results web page there's a **HERE** web link; click on it to return to the working directory. The PDB file `results.pdb` now has our LPT motif dressed with our modeled sequence!

6.4.2 MC-Search; adenosine platform

Devising the motif

For our adenosine platform, we will look for two consecutive, base-paired, adenosines. The motif is now composed of two strands: A and B. We will further impose that the A-platform be flanked by base pairs. Save the descriptor in a file called `aa.mcs`.

```
sequence( RNA A1 NN )
sequence( RNA B1 NAAN )
relation(
A1 B4 { pairing }
B2 B3 { pairing }
A2 B1 { pairing }
)
```

Submitting the motif

Submit the motif `aa.mcs` to the MC-Search web server. Don't forget to note the 10-digits working directory key. Our key is VQBgSPCDDR.

Adapting the sequence

Modify the sequence in the 3D instances, such that the 5' base pair is a G=C, and the 3' base pair is a U=A:

```
mutate( basepair A1 B4 G C )
mutate( basepair A2 B1 U A )
```

6.4.3 Using MC-Search's results

We have prepared two building blocks using MC-Search: a lonepair triloop and an adenosine platform. Now, we will see how to use them when modeling with MC-Sym. Figure 6.2 shows the MC-Sym script that loads and uses the MC-Search results. Notice the use of the *ccm* algorithm instead of *estimate* in the **ribose_rst** section. Because *ccm* optimizes the construction of the backbone, it's able to build the S-loop caused by the adenosine platform.

```
// ==================== Sequence ====================
sequence( r A1  AAUGUUACUCUGGAAUGACUAUAAACAUU )
//              ((((((((((...).))))).)..))))
//              AAAAAAAAAAAAAAAAAAAAAAAAAAAA
//              123456789012345678901234567890
//                       1         2

// ==================== Library ====================

motif_AAP = library(
    pdb( "../VQBgSPCDDR/results.pdb" )
    #1:#2, #3:#6 <- A4:A5, A23:A26 )

ncm_05a = library(
    pdb( "MCSYM-DB/ss2/UU/*.pdb.gz" )
    #1:#2 <- A5:A6
    rmsd( 0.1 sidechain && !( pse || lp || hydrogen ) ) )

ncm_05b = library(
    pdb( "MCSYM-DB/ss2/UA/*.pdb.gz" )
    #1:#2 <- A22:A23
    rmsd( 0.1 sidechain && !( pse || lp || hydrogen ) ) )

motif_LPT = library(
    pdb( "../TVB24yoMdS/results.pdb" )
    #1:#8 <- A10:A17 )

// ==================== Backtrack ====================
structure = backtrack
(
    ncm_01
    merge( ncm_02     1.5 )
    merge( ncm_03     1.5 )
    merge( motif_AAP  1.5 )
    merge( ncm_05a    1.5 )
    merge( ncm_06     1.5 )
    merge( ncm_05b    1.5 )
    merge( ncm_07     1.5 )
    merge( ncm_08     1.5 )
    merge( ncm_09     1.5 )
    merge( motif_LPT  1.5 )
)

// ================== Backtrack Restraints ==================
distance(  A5:PSY   A6:PSY  0.0   6.0 )
distance( A21:PSY  A23:PSY  0.0  13.6 )

// ================== Ribose Restraints ==================
ribose_rst
(
    structure
    method    = ccm,
    pucker    = C3p_endo,
    glycosyl  = anti,
    threshold = 2.0
)
```

Figure 6.2: MC-Sym script that uses MC-Search's results. The script is not complete; it only indicates the sections that are to be edited from the original script generated by the MC-Sym script generator.

Acknowledgments

We would like to thank the following people for helping us with the web site and the user's guide:

- Patrick Gendron, at the Institute for Research in Immunology and Cancer.
- Ali Mokdad, at the Institute for Research in Immunology and Cancer.
- Adelene Sim, at Stanford University.
- Jeremy West, at University of Chicago.
- Cédric Reymond, at University of Sherbrooke.

Notes

Bibliography

[1] Y Wang, S Juranek, H Li, G Sheng, T Tuschl, and DJ Patel. Structure of an argonaute silencing complex with a seed-containing guide DNA and target RNA duplex. *Nature*, 456:921–926, 2008.

[2] M Parisien and F Major. The MC-Fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature*, 452:51–55, 2008.

[3] DH Mathews. Revolutions in RNA secondary structure prediction. *J Mol Biol*, 359:526–532, 2006.

[4] T Xia, J SantaLucia Jr, ME Burkard, R Kierzek, SJ Schroeder, X Jiao, C Cox, and DH Turner. Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs. *Biochemistry*, 37:14719–14735, 1998.

[5] HM Berman, J Westbrook, Z Feng, G Gilliland, TN Bhat, H Weissig, IN Shindyalov, and PE Bourne. The Protein Data Bank. *Nucleic Acids Res*, 28:235–242, 2000.

[6] CB Do, DA Woods, and S Batzoglou. CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics*, 22:e90–e98, 2006.

[7] M Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*, 31:3406–3415, 2003.

[8] Y Ding, CY Chan, and CE Lawrence. Sfold web server for statistical folding and rational design of nucleic acids. *Nucleic Acids Res*, 32:135–141, 2004.

[9] IL Hofacker. Vienna RNA secondary structure server. *Nucleic Acids Res*, 31:3429–3431, 2003.

[10] DH Mathews, MD Disney, JL Childs, SJ Schroeder, M Zuker, and DH Turner. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proc Natl Acad Sci (USA)*, 101:7287–7292, 2004.

[11] J Ruan, GD Stormo, and W Zhang. An iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics*, 20:58–66, 2004.

[12] B Knudsen and J Hein. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res*, 31:3423–3428, 2003.

[13] WK Dawson, K Fujiwara, and G Kawai. Prediction of RNA pseudoknots using heuristic modeling with mapping and sequential folding. *PLoS One*, 2:e905, 2007.

[14] S Siebert and R Backofen. MARNA: multiple alignment and consensus structure prediction of RNAs based on sequence structure comparisons. *Bioinformatics*, 21:3352–3359, 2005.

[15] DH Mathews. Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization. *RNA*, 10:1178–1190, 2004.

[16] JD Puglisi, JR Wyatt, and I Tinoco Jr. A pseudoknotted RNA oligonucleotide. *Nature*, 15:283–286, 1998.

[17] DP Aalberts and NO Hodas. Asymmetry in RNA pseudoknots: observation and theory. *Nucleic Acids Res*, 33:2210–2214, 2005.

[18] MS Waterman. Sequence alignments in the neighborhood of the optimum with general application to dynamic programming. *Proc Natl Acad Sci (USA)*, 80:3123–3124, 1983.

[19] MS Waterman and TH Byers. A dynamic programming algorithm to find all solutions in the neighborhood of the optimum. *Math Biosci*, 77:179–188, 1985.

[20] C Ehresmann, F Baudin, M Mougel, P Romby, JP Ebel, and B Ehresmann. Probing the structure of RNAs in solution. *Nucleic Acids Res*, 15:9109–9128, 1987.

[21] EJ Merino, KA Wilkinson, JL Coughlan, and KM Weeks. RNA structure analysis at single nucleotide resolution by selective 2'-hydroxyl acylation and primer extension (SHAPE). *J Am Chem Soc*, 127:4223–4231, 2005.

[22] DH Mathews and DH Turner. Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. *J Mol Biol*, 317:191–203, 2002.

[23] J Reeder and R Giegerich. Consensus shapes: an alternative to the Sankoff algorithm for RNA consensus structure prediction. *Bioinformatics*, 21:3516–3523, 2005.

[24] IL Hofacker, M Fekete, and PF Stadler. secondary structure prediction for aligned RNA sequences. *J Mol Biol*, 319:1059–1066, 2002.

[25] AO Harmanci, G Sharma, and DH Mathews. PARTS: probabilistic alignment for RNA joinT secondary structure prediction. *Nucleic Acids Res*, 36:2406–2417, 2008.

[26] G dos Santos, AJ Simmonds, and HM Krause. A stem-loop structure in the wingless transcript defines a consensus motif for apical RNA transport. *Development*, 135:133–143, 2008.

[27] IL Hofacker, W Fontana, PF Stadler, S Bonhoeffer, M Tacker, and P Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie*, 125:167–188, 1994.

[28] R development core team. *R: a language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. http://www.R-project.org.

[29] F Major, M Turcotte, D Gautheret, G Lapalme, E Fillion, and R Cedergren. The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science*, 253:1255–1260, 1991.

[30] F Major. Building three-dimensional ribonucleic acid structures. *Comp Sci Eng*, 5:44–53, 2003.

[31] P Kapranov, J Cheng, S Dike, DA Nix, R Duttagupta, AT Willingham, PF Stadler, J Hertel, J Hackermüller, IL Hofacker, I Bell, E Cheung, J Drenkow, E Dumais, S Patel, G Helt, M Ganesh, S Ghosh, A Piccolboni, V Sementchenko, H Tammana, and TR Gingeras. RNA maps reveal new RNA classes and a possible function for pervasive transcription. *Science*, 316:1484–1488, 2007.

[32] DP Bartel. MicroRNAs: target recognition and regulatory functions. *Cell*, 136:215–233, 2009.

[33] CB Anfinsen, E Haber, M Sela, and FH White Jr. The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain. *Proc Natl Acad Sci (USA)*, 47:1309–1314, 1961.

[34] BA Shapiro, YG Yingling, W Kasprzak, and E Bindewald. Bridging the gap in RNA structure prediction. *Curr Opin Struct Biol*, 17:157–165, 2007.

[35] R Das and D Baker. Automated de novo prediction of native-like RNA tertiary structures. *Proc Natl Acad Sci (USA)*, 104:14664–14669, 2007.

[36] MA Jonikas, RJ Radmer, A Laederach, R Das, S Pearlman, D Herschlag, and RB Altman. Coarse-grained modeling of large RNA molecules with knowledge-based potentials and structural filters. *RNA*, 15:189–199, 2009.

[37] HM Martinez, JV Maizel Jr, and BA Shapiro. RNA2D3D: a program for generating, viewing, and comparing 3-dimensional models of RNA. *J Biomol Struct Dyn*, 25:669–684, 2008.

[38] F Ding, S Sharma, P Chalasani, VV Demidov, NE Broude, and NV Dokholyan. Ab initio RNA folding by discrete molecular dynamics: from structure prediction to folding mechanisms. *RNA*, 14:1164–1173, 2008.

[39] J Frellsen, I Moltke, M Thiim, KV Mardia, J Ferkinghoff-Borg, and T Hamelryck. A probabilistic model of RNA conformational space. *PLoS Comput Biol*, 5:e1000406, 2009.

[40] E Westhof and F Jossinet. *The Assemble project*. Architecture et réactivité de l'ARN, Strasbourg, France, 2008.

[41] CC Correll, A Munishkin, YL Chan, Z Ren, IG Wool, and TA Steitz. Crystal structure of the ribosomal RNA domain essential for binding elongation factors. *Proc Natl Acad Sci (USA)*, 95:13436–13441, 1998.

[42] WL DeLano. *The PyMOL molecular graphics system*. DeLano Scientific, Palo Alto, CA, USA, 2002.

[43] AS Krasilnikov and A Mondragón. On the occurrence of the T-loop RNA folding motif in large RNA molecules. *RNA*, 9:640–643, 2003.

[44] JC Lee, JJ Cannone, and RR Gutell. The lonepair triloop: a new motif in RNA structure. *J Mol Biol*, 325:65–83, 2003.

[45] H Sierzputowska-Gracz, RA McKenzie, and EC Theil. The importance of a single G in the hairpin loop of the iron responsive element (IRE) in ferritin mRNA for structure: an NMR spectroscopy study. *Nucleic Acids Res*, 23:146–153, 1995.

[46] T Kulinski, M Olejniczak, H Huthoff, L Bielecki, K Pachulska-Wieczorek, AT Das, B Berkhout, and RW Adamiak. The apical loop of the HIV-1 TAR RNA hairpin is stabilized by a cross-loop base pair. *J Biol Chem*, 278:38892–38901, 2003.

[47] A Lescoute and E Westhof. Topology of three-way junctions in folded RNAs. *RNA*, 12:83–93, 2006.

[48] M Popenda, M Blazewicz, M Szachniuk, and RW Adamiak. RNA FRABASE version 1.0: an engine with a database to search for the three-dimensional fragments within RNA structures. *Nucleic Acids Res*, 36:D386–D391, 2008.

[49] JW Ponder and FM Richards. An efficient Newton-like method for molecular mechanics energy minimization of large molecules. *J Comput Chem*, 8:1016–1024, 1987.

[50] P Ren and JW Ponder. Polarizable atomic multipole water model for molecular mechanics simulation. *J Phys Chem B*, 107:5933–5947, 2003.

[51] J Wang, P Cieplak, and PA Kollman. How well does a restrained electrostatic potential (RESP) model perform in calculating conformational energies of organic and biological molecules? *J Comput Chem*, 21:1049–1074, 2000.

[52] W Saenger. *Principles of Nucleic Acid Structure*. Springer-Verlag, New-York, 1984.

[53] NB Leontis and E Westhof. Geometric nomenclature and classification of RNA base pairs. *RNA*, 7:499–512, 2001.

[54] S Lemieux and F Major. RNA canonical and non-canonical base pairing types: a recognition method and complete repertoire. *Nucleic Acids Res*, 30:4250–4263, 2002.

[55] H Wu and J Feigon. H/ACA small nucleolar RNA pseudouridylation pockets bind substrate RNA to form three-way junctions that position the target U for modification. *Proc Natl Acad Sci (USA)*, 104:6655–6660, 2007.

[56] R Tyagi and DH Mathews. Predicting helical coaxial stacking in RNA multibranch loops. *RNA*, 13:939–951, 2007.

[57] PV Hentenryck. *Constraint Satisfaction in Logic Programming*. The MIT Press, Cambridge, USA, 1989.

[58] R Dechter and D Frost. Backjump-based backtracking for constraint satisfaction. *Artif Intell*, 136:147–188, 2008.

[59] CA Hunter. Sequence-dependent DNA structure. the role of base stacking interactions. *J Mol Biol*, 230:1025–1054, 1993.

[60] CA Hunter and XJ Lu. DNA base-stacking interactions: a comparison of theoretical calculations with oligonucleotide x-ray crystal structures. *J Mol Biol*, 265:603–619, 1997.

[61] CA Hunter. Sequence-dependent DNA structure. the role of the sugar-phosphate backbone. *J Mol Biol*, 280:407–420, 1998.

[62] WK Olson, M Esguerra, Y Xin, and XJ Lu. New information content in RNA base pairing deduced from quantitative analysis of high-resolution structures. *Methods*, 47:177–186, 2009.

[63] DJ Klein, TM Schmeing, PB Moore, and TA Steitz. The kink-turn: a new RNA secondary structure motif. *EMBO J*, 20:4214–4221, 2001.

[64] A Lescoute, NB Leontis, C Massire, and E Westhof. Recurrent structural RNA motifs, isostericity matrices and sequence alignments. *Nucleic Acids Res*, 33:2395–2409, 2005.

[65] W Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Cryst*, A34:827–828, 1978.

[66] A Zemla, C Venclovas, J Moult, and K Fidelis. Processing and analysis of CASP3 protein structure predictions. *Proteins*, Suppl 3:22–29, 1999.

[67] K Ginalski, NV Grishin, A Godzik, and L Rychlewski. Practical lessons from protein structure prediction. *Nucleic Acids Res*, 33:1874–1891, 2005.

[68] M Parisien, JA Cruz, E Westhof, and F Major. New metrics for comparing and assessing discrepancies between RNA 3D structures and models. *RNA*, 2009. accepted for publication.

[69] BW Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim Biophys Acta*, 405:442–451, 1975.

[70] J Gorodkin, SL Stricklin, and GD Stormo. Discovering common stem-loop motifs in unaligned RNA sequences. *Nucleic Acids Res*, 29:2135–2144, 2001.

[71] A Laederach, JM Chan, A Schwartzman, E Willgohs, and RB Altman. Coplanar and coaxial orientations of RNA bases and helices. *RNA*, 13:643–650, 2007.

[72] C Hyeon, RI Dima, and D Thirumalai. Size, shape, and flexibility of RNA structures. *J Chem Phys*, 125:194905, 2006.

[73] M Levitt. Detailed molecular model for transfer ribonucleic acid. *Nature*, 172:759–763, 1969.

[74] TM Klingler and DL Brutlag. Detection of correlations in tRNA sequences with structural implications. *Proc Int Conf Intell Syst Mol Biol*, 1:225–233, 1993.

[75] F Michel and E Westhof. Modelling of the three-dimensional architecture of group I catalytic introns based on comparative sequence analysis. *J Mol Biol*, 216:585–610, 1990.

[76] D Gautheret and RR Gutell. Inferring the conformation of RNA base pairs and triples from patterns of sequence variation. *Nucleic Acids Res*, 25:1559–1564, 1997.

[77] PJ Lukavsky and JD Puglisi. Structure determination of large biological RNAs. *Methods Enzymol*, 394:399–416, 2005.

[78] MP Latham and A Pardi. Measurement of imino 1H-1H residual dipolar couplings in RNA. *J Biomol NMR*, 43:121–129, 2009.

[79] H Han and PB Dervan. Visualization of RNA tertiary structure by RNA-EDTA.Fe(II) autocleavage: analysis of tRNA(Phe) with uridine-EDTA.Fe(II) at position 47. *Proc Natl Acad Sci (USA)*, 91:4955–4959, 1994.

[80] CM Gherghe, CW Leonard, F Ding, NV Dokholyan, and KM Weeks. Native-like RNA tertiary structures using a sequence-encoded cleavage agent and refinement by discrete molecular dynamics. *J Am Chem Soc*, 131:2541–2546, 2009.

[81] R Das, M Kudaravalli, M Jonikas, A Laederach, R Fong, JP Schwans, D Baker, JA Piccirilli, RB Altman, and D Herschlag. Structural inference of native and partially folded RNA by high-throughput contact mapping. *Proc Natl Acad Sci (USA)*, 105:4144–4149, 2008.

[82] JA Latham and TR Cech. Defining the inside and outside of a catalytic RNA molecule. *Science*, 245:276–282, 1989.

[83] TD Tullius and JA Greenbaum. Mapping nucleic acid structure by hydroxyl radical cleavage. *Curr Opin Chem Biol*, 9:127–134, 2005.

[84] JH Cate, AR Gooding, E Podell, K Zhou, BL Golden, CE Kundrot, TR Cech, and JA Doudna. Crystal structure of a group I ribozyme domain: principles of RNA packing. *Science*, 273:99–109, 1996.

[85] K Takamoto, R Das, Q He, S Doniach, M Brenowitz, D Herschlag, and MR Chance. Principles of RNA compaction: insights from the equilibrium folding pathway of the P4-P6 RNA domain in monovalent cations. *J Mol Biol*, 343:1195–1206, 2004.

[86] W Wriggers, RA Milligan, and JA McCammon. Situs: A package for docking crystal structures into low-resolution maps from electron microscopy. *J Struct Biol*, 125:185–195, 1999.

[87] Y Byun and K Han. PseudoViewer3: generating planar drawings of large-scale RNA structures with pseudoknots. *Bioinformatics*, 25:1435–1437, 2009.

[88] C Olivier, G Poirier, P Gendron, A Boisgontier, F Major, and P Chartrand. Identification of a conserved RNA motif essential for she2p recognition and mRNA localization to the yeast bud. *Mol Cell Biol*, 25:4752–4766, 2005.

[89] PB Moore. Structural motifs in RNA. *Annu Rev Biochem*, 68:287–300, 1999.

[90] DK Hendrix, SE Brenner, and SR Holbrook. RNA structural motifs: building blocks of a modular biomolecule. *Q Rev Biophys*, 38:221–243, 2005.

[91] NB Leontis, A Lescoute, and E Westhof. The building blocks and motifs of RNA architecture. *Curr Opin Struct Biol*, 16:279–287, 2006.

[92] JH Cate, AR Gooding, R Podell, Z Zhou, BL Golden, AA Szewczak, CE Kundrot, TR Cech, and JA Doudna. RNA tertiary structure mediation by adenosine platforms. *Science*, 273:1696–1699, 1996.

[93] L Jaeger, EJ Verzemnieks, and C Geary. The UA_handle: a versatile submotif in stable RNA architectures. *Nucleic Acids Res*, 37:215–230, 2009.