Parametric RNA Partition Function Algorithms

Author: Yang Ding

Persistent link: http://hdl.handle.net/2345/1813

This work is posted on eScholarship@BC, Boston College University Libraries.

Boston College Electronic Thesis or Dissertation, 2010

Copyright is held by the author, with all rights reserved, unless otherwise noted.
Boston College

The Graduate School of Arts and Sciences

Department of Biology

Parametric RNA Partition Function Algorithms

a thesis

by

YANG DING

submitted in partial fulfillment of the requirements

for the degree of

Master of Science

August 2010
ABSTRACT
Parametric RNA Partition Function Algorithms
by
Yang Ding

Chair: Peter Clote

In addition to the well-characterized messenger RNA, transfer RNA and ribosomal RNA, many new classes of noncoding RNA (ncRNA) have been discovered in the past few years. ncRNA has been shown to play important roles in multiple regulation and development processes. The increasing needs for RNA structural analysis software provide great opportunities on computational biologists.

In this thesis I present three highly non-trivial RNA parametric structural analysis algorithms: 1) RNAhairpin and RNAmultiloop, which calculate partition functions with respect to hairpin number, multiloop number and multiloop order, 2) RNAshapeEval, which is based upon partition function calculation with respect to a fixed abstract shape, and 3) RNAProfileZ, which calculates the expected partition function and ensemble free energy given an RNA position weight matrix. I also describe the application of these software in biological problems, including evaluating purine riboswitch aptamer full alignment sequences to adopt their consensus shape, building hairpin and multiloop profiles for certain Rfam families, tRNA and pseudoknotted RNA secondary structure predictions.

These algorithms hold the promise to be useful in a broad range of biological problems such as structural motifs search, ncRNA gene finders, canonical and pseudoknotted secondary structure predictions.
ACKNOWLEDGEMENTS

I would like to express great gratitude towards my advisor, Professor Peter Clote, not only for his invaluable guidance in the theoretical RNA field, but also for his willingness to spare a lot of time to listen to my sometimes naive research ideas and discuss them with me.

Thanks to Professors Peter Clote, Jeffrey Chuang and David Mathews for being in my master thesis committee.

I want to thank Dr.W. Andy Lorenz for countless long discussions about research and life during the past three years, and helping me immensely in going through the up and downs in graduate school.

I would like to thank Prof.Jeffrey Chuang for having multiple discussions with me about current biological advances, teaching me a lot of biology, as well as giving me many helpful career advices.

I also would like to thank Prof.David Mathews for carefully reviewing my thesis almost line by line, making numerous comments and pointing out quite a few typos. Hopefully the final version of the thesis is significantly better than the first draft that you read!

Special thanks go to my parents, for always being by my side in the past 25 years, no matter what difficulty I have been going through. I will try my best to make you proud.

I would like to thank my personal friends Jiantao Wu, Chenjia Xu, Lulu Zhang,
Mike Lers and Dr. James Whittle, without them my life in Boston will be much less colorful.

I also want to thank my lab members Vinodh Mechery, Dr. Ivan Dotu, Dr. Valentin Ilyin and Dr. Chip Stewart from the Marth lab, for their help in my research and many discussions with me in the past few years.
TABLE OF CONTENTS

ACKNOWLEDGEMENTS ........................................... i
LIST OF FIGURES ............................................ v
LIST OF TABLES ............................................... vi

CHAPTER

I. Introduction .................................................. 1
  1.1 Background ............................................... 1
  1.2 Overview ............................................... 3

II. Theoretical Background ...................................... 5
  2.1 RNA Secondary Structure ................................ 5
  2.2 Energy Models .......................................... 6
  2.3 Nussinov-Jacobson MFE algorithm ....................... 7
  2.4 Zuker MFE Algorithm .................................... 8
  2.5 McCaskill Algorithm .................................... 11
  2.6 Ding and Lawrence Sampling Algorithm ............... 13

III. Parametric Partition Function of RNA with respect to Hairpin Number, Multiloop Number and Multiloop Order ................. 15
  3.1 Number of Hairpins ..................................... 15
  3.2 Number of Multiloops ................................... 17
  3.3 Multiloop Order ....................................... 18
  3.4 RNAmultiloop, RNAhairpin .............................. 20
  3.5 Computational Complexity ................................ 21

IV. RNAshapeEval: Parametric Partition Function of RNA with Respect to an Abstract Shape ............................................. 22
  4.1 Recursions for Shape Partition Function ............. 22
  4.2 RNAshapeEval, RNAshapeMFE, RNAshapeSamp ........ 26
  4.3 Computational Complexity ................................ 28

V. RNAprofileZ: Expected Partition Function of an RNA Position Specific Scoring Matrix .................................................... 29
  5.1 Overview ............................................... 29
  5.2 Derivation of Hairpin Recursions ...................... 33
  5.3 Derivation of Interior Loop Recursions ............... 34
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Feynman diagram of original recursions from McCaskill’s algorithm [15] to compute the partition function.</td>
<td>10</td>
</tr>
<tr>
<td>4.1</td>
<td>Parse tree of the shape [ ] [ [ ] [ ] ]</td>
<td>26</td>
</tr>
<tr>
<td>6.1</td>
<td>The distribution of shape probability scores for seed alignment vs. full alignment. The sequences are from Rfam purine riboswitch aptamers (RF00167).</td>
<td>38</td>
</tr>
<tr>
<td>6.2</td>
<td>The Hairpin Profiles for Rfam family RF00004, RF00005, RF00015.</td>
<td>40</td>
</tr>
<tr>
<td>6.3</td>
<td>The Multiloop Number Profiles for Rfam family RF00010, RF00023, RF00036.</td>
<td>41</td>
</tr>
<tr>
<td>6.4</td>
<td>The Multiloop Order Profiles for Rfam family RF00010, RF00023, RF00036.</td>
<td>42</td>
</tr>
</tbody>
</table>
TABLE OF TABLES

Table

6.1 Performance Comparison of MFE, shapeMFE and shapeMEA method on tRNA data 43
CHAPTER I

Introduction

1.1 Background

In the past few years, the traditional view that RNA is merely a passive carrier of genetic information has been shown to be simplistic and even misleading. Molecular biologists have started to appreciate the extensive role that noncoding RNA (ncRNA) plays in regulation and development.

One prominent example of ncRNA is micro-RNA (miRNA). Derived from stem-loop precursor structures, miRNAs are post-transcriptional regulatory elements that bind to complementary sequences in the 3′ untranslated regions (UTRs) of target messenger RNA transcripts 1 (mRNAs), resulting in gene silencing. Since its discovery in early 1990s [1], miRNA has been shown to be involved in multiple negative regulation processes such as transcript degradation and sequestering, translational suppression and is suggested to be involved in some positive regulation process such as transcriptional and translational activation.

Riboswitches are another interesting example. Bacterial riboswitches are a portion of the 5′ UTR of messenger RNA that can undergo a conformational change, ultimately regulating protein production. Riboswitches are often found upstream

---

1This statement has been experimentally validated, with a few notable exceptions, for miRNA in animals. In contrast, miRNAs in plants generally hybridize to coding regions of mRNA.
of operons, regulating groups of genes, as in purine de novo synthesis and salvage [20]. In the past two years, eukaryotic riboswitches have been found that reside in an intronic region and control alternative messenger RNA splicing by conformational change.

There are many more such examples. To date, the public RNA sequence repository Rfam (version 10.0 [5]) has documented 1446 families of ncRNAs.

Yet still much more is to be discovered. With the rapid development of cheaper and faster sequencing technologies, transcriptomes under multiple conditions from different organisms will soon be sequenced. As one of hottest areas in the current biological research, there is no doubt that there are many new classes of ncRNAs to be discovered and unexpected functions to be associated with the known ncRNA families.

Since in biology the function of a molecule is usually closely associated with its structure, it is of great interest for biologists to determine the structure of a newly acquired RNA sequence. However, current experimental structure-determination technologies such as X-ray crystallography, Nuclear Magnetic Resonance (NMR) or chemical probing are all time-consuming and labor-intensive, so there is no obvious way to determine RNA structures in a high throughput manner.

To partially meet this demand, computational biologists have been trying very hard in the past few decades to build computational energy models for RNAs, and design algorithms based on this model to do noncoding RNA detection, structural prediction, structural alignment etc. Due to the high flexibility of RNA molecules, most of computational research of RNAs are focused on their secondary structures, i.e. all the base pair interactions. The most sophisticated and widely used RNA secondary structure energy model so far is the Turner Nearest-Neighbor Energy
Model. It is an energy model that decomposes RNA secondary structures into loops, and the sum of all loop energy scores equal to the total free energy of the molecule. Most of the loop energy scores are determined experimentally.

Based on this energy model researchers have devised algorithms to predict the RNA Minimum Free Energy (MFE) secondary structure [29], calculate the total partition function [15] of an RNA sequence and sample RNA secondary structures [3] according to their Boltzmann probability, to name only a few.

The algorithms presented in this thesis are intricate extensions of these algorithms. In this set of algorithms we introduce another dimension: we calculate the MFE secondary structure, partition function and sample secondary structures according to the value of a specific parameter, such as the abstract shape, the number of hairpins, number of multiloops etc. These algorithms are in a way following the same lines of other parametric algorithms developed by the Clote Lab, such as RNAbor [4], which calculates the partition function with respect to the base pair distance from a given (initial) secondary structure, and RNAmutants[23], which calculates the total partition function for all the sequences that has a certain distance from the original RNA sequence.

1.2 Overview

In this thesis we describe three sets of dynamic programming algorithms to calculate RNA parametric partition functions, as well as their applications in biological problems. After an introduction to the subject, in Chapter 2 we give necessary background on which our algorithms are based upon: the Nussinov-Jacobson [19] and Turner Nearest-Neighbor Energy Model [13, 28], the Zuker algorithm [29] to calculate the Minimum Free Energy (MFE) secondary structure, the McCaskill algorithm [15]
to calculate the partition function of an RNA sequence and Ding-Lawrence algorithm [3] to sample RNA secondary structures according to their Boltzmann probability.

In the following three chapters, I present the parametric partition function algorithms. More specifically, in Chapter 3, I present the algorithm for the program RNAhairpin, which calculates the partition function and Boltmann probability with respect to the number of hairpins, as well as the algorithms for RNAmultiloop, which calculates the partition function and Boltzmann probability with respect to the number of multiloops and multiloop order.

In Chapter 4 I present the program RNAshapeEval, which finds the RNA subsegment that has the highest probability of adopting a fixed shape, and calculate the MFE structure for the subsegment; RNAshapeMFE, which calculates the MFE secondary structure within a fixed shape and RNAshapeSamp, which samples the secondary structure within the fixed shape, according to their Boltzmann probabilities.

In Chapter 5, I present the algorithm for the program RNAprofileZ, which calculates the expected partition function given an RNA position weight matrix.

In Chapter 6 I present the biological applications of my parametric partition function algorithms, including both experimental results that I got from running my program on certain families of Rfam sequences.

I summarize my results and point out potential improvements and future work in Chapter 7.

In conclusion, within this thesis, I present three highly non-trivial RNA parametric partition function calculation algorithms, which could prove to be promising tools in various aspects of RNA research, including structural motif search, secondary structure prediction, noncoding RNA detection and characterization etc.
CHAPTER II

Theoretical Background

In this chapter I will present the energy models and algorithms which my parametric partition algorithms are based upon.

2.1 RNA Secondary Structure

Given an RNA sequence $S = a_1, a_2, ..., a_n$, the secondary structure is a set of base pairs, which satisfies the following conditions. Letting $(i, j)$ denote the base pair between nucleotides $a_i$ and $a_j$, where $i < j$, and letting $a[i, j]$ denote the subsequence $a_i, ..., a_j$, we have the following requirements.

- Two base pairs $(i, j), (i', j')$ are either identical, or $i \neq i'$ and $j \neq j'$. This means that we don’t allow base triples in the definition of secondary structures.

- For any base pair $(i, j)$, we have $j - i > 3$. This means that sharp hairpin loop are prohibited for biophysical considerations.

- For any two base pairs $(i, j), (i', j')$, assuming $i < i'$, then either $i < i' < j' < j$ or $i < j < i' < j'$. This means that pseudoknots are not considered, mostly due to consideration of computational complexity.

The only nucleotides that are allowed in a base pair are A-U, U-A, C-G, G-C and the wobble base pair G-U, U-G.
2.2 Energy Models

There are two major RNA secondary structure models:

The **Nussinov-Jacobson Energy Model** assigns different energy scores to each type of base pairs, and the total free energy of a secondary structure equals the sum of all the base pair energy scores.

In this model:

\[
E(S) = a \cdot N(A,U) + b \cdot N(C,G) + c \cdot N(G,U)
\]

Here \(a, b, c\) are energy parameters, \(N(A,U)\) is the number of AU base pairs, \(N(C,G)\) is the number of CG base pairs, \(N(G,U)\) is the number of GU wobble base pairs.

Although it’s a useful toy model to consider certain theoretical problems, the Nussinov-Jacobson model is generally considered as not accurate enough for serious biological applications.

The **Turner Nearest-Neighbor Energy Model** is a loop-based model. Each RNA secondary structure can be uniquely decomposed into a set of non-overlapping loops such as stacked base pairs, hairpins, bulges, interior loops and multiloops. Each of the loop energy values can be obtained through table look-up or heuristic formula.

In this model the energy values for stacked base pairs, hairpins, bulges and interior loops of different nucleotide compositions are all determined experimentally, by the least-square fit of the UV absorption (optical melting) data, while the energy parameters for multiloop are assigned by a heuristic affine function, which is due to the consideration of experimental evidence as well as computational complexity. Total energy of a secondary structure equals the sum of all the loop energy scores.
2.3 Nussinov-Jacobson MFE algorithm

Given an energy model, one of the most interesting questions, is of all the valid secondary structures, which one of them has the lowest free energy. This energy is called the Minimum Free Energy (MFE), and the secondary structure that has this energy is called Minimum Free Energy structure (MFE structure).

In this section I will describe the Nussinov-Jacobson algorithm to compute the Minimum Free Energy under the Nussinov-Jacobson energy model.

To make the presentation easier, we define an energy function \( a(i, j) \) between nucleotide \( i \) and nucleotide \( j \) to be:

\[
a(i, j) = \begin{cases} 
-5, & \text{if } a_i, a_j \text{ is CG or GC}, \\
-4, & \text{if } a_i, a_j \text{ is AU or UA}, \\
-1, & \text{if } a_i, a_j \text{ is GU or UG}, \\
+\infty, & \text{all the other cases}.
\end{cases}
\]

The actually energy parameter values could be modified.

The Nussinov-Jacobson algorithm solves the MFE problem by dynamic programming technique. Define \( E(i, j) \) to be the minimum free energy of a secondary structure on the sequence \( a[i, j] = a_i, a_{i+1}, ..., a_j \). We have the base case, \( E(i, j) = 0 \) if \( j - i \leq \theta \), where \( \theta \) (usually taken to be 3) is the minimum number of unpaired bases in a hairpin loop. The inductive case is given as follows:

\[
E(i, j) = \min \begin{cases} 
E(i, j - 1) \\
E(i + 1, j - 1) + a(i, j) \\
\min_{i < k < j} E(i, k - 1) + a(k, j) + E(k + 1, j - 1)
\end{cases}
\]

The idea is to consider rightmost nucleotide, it will either be not base paired, or base pair with the leftmost nucleotide, or base pair with some nucleotide in the middle part of the sequence.
2.4 Zuker MFE Algorithm

Next I will introduce the Zuker MFE algorithm, which calculates MFE and MFE secondary structure of an RNA sequence.

To do this we will need the following notations and definitions:

Given RNA nucleotide sequence $a_1, \ldots, a_n$, we will use the standard notation $\mathcal{H}$ to denote the free energy of a hairpin, $\mathcal{I}$ to denote the free energy of an interior loop (combining the cases of stacked base pair, bulge and proper internal loop), while the free energy for a multiloop containing $N_b$ base pairs and $N_u$ unpaired bases is given by the affine approximation $a + bN_b + cN_u$.

**Definition II.1 (Minimum Free Energy).**

- $E(i, j)$: minimum free energy over all secondary structures of $a[i, j]$.

- $EB(i, j)$: minimum free energy over all secondary structures of $a[i, j]$, in which nucleotide $i$ and $j$ form a base pair.

- $EM(i, j)$: minimum free energy over all secondary structures of $a[i, j]$, subject to the constraint that $a[i, j]$ is part of a multiloop and has *at least* one component.

- $EM1(i, j)$: minimum free energy over all secondary structures of $a[i, j]$, subject to the constraint that $a[i, j]$ is part of a multiloop and has *exactly* one component. Moreover, it is *required* that $i$ base-pair in the interval $[i, j]$; i.e. $(i, r)$ is a base pair, for some $i < r \leq j$.

With this, we have the following recursions that define the minimum free energy on a subsequence of $a_1, \ldots, a_n$: 
\[
E(i, j) = \begin{cases} 
E(i, j - 1), & \text{when nucleotide } j \text{ is unpaired.} \\
EB(i, j), & \text{when nucleotide } i \text{ and nucleotide } j \text{ base-pair.} \\
\min_{i < r < j - 3} (E(i, r - 1) + EB(r, j)), & \text{when nucleotide } j \text{ base pair with } i < r < j - 3.
\end{cases}
\]

The constrained MFE closed by base pair \((i, j)\) is given by

\[
EB(i, j) = \begin{cases} 
H(i, j), & \text{hairpin} \\
\min_{i < \ell < r \leq j} I(i, \ell, r, j), & \text{interior loop or bulge} \\
\min_{i + 1 \leq r \leq j - 5} (a + b + EM(i + 1, r - 1) + EM1(r, j - 1)), & \text{multiloop}
\end{cases}
\]

The multiloop MFE function with a single component and where position \(i\) is required to base-pair in the interval \([i, j]\) is given by

\[
EM1(i, j) = \min_{i + 4 \leq r \leq j} (EB(i, r) + b + c(j - r)).
\]

Finally, the multiloop MFE function with one or more components, having no requirement that position \(i\) base-pair in the interval \([i, j]\) is given by

\[
EM(i, j) = \begin{cases} 
\min_{i \leq r \leq j - 4} (EM1(r, j) + c(r - i)), \\
\min_{i + 5 \leq r \leq j - 4} (EM(i, r - 1) + EM1(r, j)).
\end{cases}
\]

See Figure 2.1 for a pictorial representation of the recursions of the McCaskill’s partition function algorithm. The recursions just described for free energy can be graphically visualized by replacing ‘\(Z\)’ by ‘\(E\)’ and ‘+’ by ‘\(\text{min}\)’.
Having introduced the algorithm to compute the Minimum Free Energy, the next question is how to retrieve the secondary structure that has the Minimum Free Energy. To do this we will need more traceback or backtracking functions, one for each of the recursions for E, EB, EM, EM1. To give an idea of how backtracking works, I will give the pseudocode for the backtrack function for E in Algorithm 1.

Notice the above backtrack function will only return one secondary structure, so if there are multiple secondary structures with the same MFE, the algorithm will only return one of them. This could cause serious problems when predicting the secondary structures for very large RNAs. An exponential algorithm to list all the secondary structures above a certain energy threshold from MFE is described in [27].

The backtracking algorithm described here is the deterministic version of the Ding-
Algorithm 1 Backtrack Function for E, BacktrackE(i,j,paren)

\[ E_0 = E(i,j); \]
if \( j - i < 4 \) then
    return;
else if \( E(i,j-1) == E_0 \) then
    BacktrackE(i, j-1, paren); //j is unpaired.
else if \( EB(i,j) == E_0 \) then
    paren[i] = '(';
    paren[j] = ')';
    BacktrackEB(i, j, paren); //i base pair with j.
else
    for \( r = i + 1 \) to \( j - 4 \) do
        if \( E(i,r-1)+EB(r,j)==E_0 \) then
            paren[r] = '(';
            paren[j] = ')';
            BacktrackEB(r, j, paren); // j base pair with a nucleotide in the middle.
            BacktrackE(i, r-1, paren);
        end if
    end for
end if

Lawrence sampling algorithm that I will describe later.

2.5 McCaskill Algorithm

Analogous to the Zuker MFE algorithm, one can calculate the total partition function of an RNA sequence by McCaskill’s algorithm.

In addition to the notations and definitions introduced in the last section, we will need the following:

For RNA sequence \( a_1, \ldots, a_n \), for all \( 1 \leq i \leq j \leq n \), the McCaskill partition function \( Z(i,j) \) is defined by \( \sum_S e^{-E(S)/RT} \), where the sum is taken over all secondary structures \( S \) of \( a[i,j] \), \( E(S) \) is the free energy of secondary structure \( S \), \( R \) is the universal gas constant with value \( R = 1.987 \text{ cal/mol}^{-1} \text{ K}^{-1} \), and \( T \) is absolute temperature.

Definition II.2 (McCaskill’s partition function).

- \( Z(i,j) \): partition function over all secondary structures of \( a[i,j] \).
• $ZB(i, j)$: partition function over all secondary structures of $a[i, j]$, which contain the base pair $(i, j)$.

• $ZM(i, j)$: partition function over all secondary structures of $a[i, j]$, subject to the constraint that $a[i, j]$ is part of a multiloop and has at least one component.

• $ZM1(i, j)$: partition function over all secondary structures of $a[i, j]$, subject to the constraint that $a[i, j]$ is part of a multiloop and has exactly one component. Moreover, it is required that $i$ base-pair in the interval $[i, j]$; i.e. $(i, r)$ is a base pair, for some $i < r \leq j$.

With this, we have the unconstrained partition function

$$Z(i, j) = Z(i, j - 1) + ZB(i, j) + \sum_{r=i+1}^{j-\theta-1} Z(i, r - 1) \cdot ZB(r, j). \quad (2.7)$$

The constrained partition function closed by base pair $(i, j)$ is given by

$$ZB(i, j) = e^{-H(i,j)/RT} + \sum_{i \leq \ell \leq r \leq j} e^{-\mathcal{I}(i,\ell,r,j)/RT} + e^{-(a+b)/RT} \left( \sum_{r=i+1}^{j-\theta-2} ZM(i+1, r - 1) \cdot ZM1(r, j - 1) \right). \quad (2.8)$$

The multiloop partition function with a single component and where position $i$ is required to base-pair in the interval $[i, j]$ is given by

$$ZM1(i, j) = \sum_{r=i+\theta+1}^{j} ZB(i, r) \cdot e^{-\left(b+c(j-r)\right)/RT} \quad (2.9)$$

Finally, the multiloop partition function with one or more components, having no requirement that position $i$ base-pair in the interval $[i, j]$ is given by

$$ZM(i, j) = \sum_{r=i}^{j-\theta-1} ZM1(r, j) \cdot e^{-c(r-i)/RT} + \sum_{r=i+\theta+2}^{j-\theta-1} ZM(i, r - 1) \cdot ZM1(r, j). \quad (2.10)$$

See Figure 2.1 for a pictorial representation of the recursions of McCaskill’s (original) algorithm [15];

We now turn to the Ding and Lawrence algorithm to sample secondary structures of an RNA sequence according to their Boltzmann probabilities.
2.6 Ding and Lawrence Sampling Algorithm

In this section I will describe the Ding and Lawrence sampling algorithm. To do stochastic backtracking we will first need to calculate the partition function matrices according to the McCaskill algorithm. Similar to the backtracking process for MFE structure, we will need four sets of backtrack functions, called $S_{\text{BacktrackZ}}$, $S_{\text{BacktrackZB}}$, $S_{\text{BacktrackZM1}}$, $S_{\text{BacktrackZM}}$, and they represent the backtrack function for $Z$, $Z_B$, $Z_M$, $Z_{M1}$ correspondingly. The backtrack function for $Z$ will be described below:

**Algorithm 2** Stochastic Backtrack Function for $Z$, $S_{\text{BacktrackZ}}(i,j,\text{paren})$

1. $Z_0 = Z(i,j)$;
2. $\text{sum} = 0$;
3. $rd = \text{random}(\cdot) \cdot Z_0$; //Generate a random number between 0 and $Z_0$.
4. if $j - i < 4$ then
   return;
5. else
   $\text{sum} += Z(i, j - 1)$; // $j$ is unpaired.
6. if $\text{sum} > rd$ then
   $S_{\text{BacktrackZ}}(i, j-1, \text{paren})$;
   end if
7. $\text{sum} += Z_B(i, j)$; // $i$ base-pairs with $j$.
8. if $\text{sum} > rd$ then
   paren[i]='(';
   paren[j]=')';
   $S_{\text{BacktrackZB}}(i, j, \text{paren})$;
   end if
9. for $r = i + 1$ to $j - 4$ do
   $\text{sum} += Z(i, r - 1) \ast Z_B(r, j)$; // $j$ base pair with a nucleotide in the middle.
   if $\text{sum} > rd$ then
   paren[r]='(';
   paren[j]=')';
   $S_{\text{BacktrackZB}}(r,j, \text{paren})$;
   $S_{\text{BacktrackZ}}(i, r-1, \text{paren})$;
   end if
   end for
   end if

The idea is that we first generate a random number between 0 and $Z_0 = Z(i,j)$, then determine the contribution to the Boltzmann partition function for each sub-case. Depending upon which interval the random number lies, we recursively apply
the stochastic backtracking function. Each time we call the backtrack function for ZB, we add a base pair to our sampled secondary structure, since ZB(i,j) represents the case where nucleotide i and nucleotide j base-pair.
CHAPTER III

Parametric Partition Function of RNA with respect to Hairpin Number, Multiloop Number and Multiloop Order

In this chapter, I present the recursions to calculate the partition functions for number of hairpins, number of multiloops and multiloop order.

3.1 Number of Hairpins

We begin by defining some abbreviations for the partition function for hairpins

\[ Z_H(i, j) = \begin{cases} 
0 & \text{if } j - i \leq \theta \\
 e^{-H(i,j)/RT} & \text{else}
\end{cases} \]

where \( \theta \) is a threshold for the smallest allowable size of a hairpin loop, usually set as 3.

And we define the partition function for internal loops that having \( h \) hairpins as

\[ Z_I^h(i, j) = \sum_{i<k<\ell<j} e^{-I(i,j;k,\ell)/RT} \cdot Z_B^h(k, \ell) \]

where the sum is over \( k, \ell \) such that \( 1 \leq i < k < \ell < j \leq n \). This combines the treatment of both left and right bulges with proper internal loops.

For \( h \geq 0 \), if \( j < i \) then let \( Z^h(i, j) = Z_B^h(i, j) = Z_M^h(i, j) = Z_M1^h(i, j) = 0 \).

For \( h = 0 \), define \( Z^h(i, i) = 1 \), and for \( h > 0 \), define \( Z^h(i, i) = 0 \). For \( h \geq 0 \), define \( Z_B^h(i, i) = Z_M^h(i, i) = Z_M1^h(i, i) = 0 \). The following recursions define the
partition functions for \( 1 \leq i < j \leq n \). The unconstrained partition function for secondary structures restricted to the interval \([i, j]\) that contain \( h \) hairpins is given by

\[
Z^h(i, j) = \begin{cases} 
1 & \text{if } h = 0 \\
Z^h(i, j - 1) + ZB^h(i, j) + \sum_{r=i+1}^{j-\theta-1} \sum_{k=0}^{h-1} Z^k(i, r - 1) \cdot ZB^{h-k}(r, j) & \text{if } h > 0.
\end{cases}
\]

The partition function for secondary structures restricted to the interval \([i, j]\) that contain \( h \) hairpins and are closed by the base pair \((i, j)\) is given by

\[
ZB^h(i, j) = \begin{cases} 
0 & \text{if } h = 0 \\
ZH(i, j) + ZI^h(i, j) & \text{if } h = 1 \\
ZI^h(i, j) + \sum_{r=i+\theta+3}^{j-\theta-2} \sum_{k=1}^{h-1} ZM^k(i + 1, r - 1) \cdot ZM^{1-h-k}(r, j - 1) \cdot e^{-(a+b)/RT} & \text{if } h \geq 2
\end{cases}
\]

provided that \( i, j \) form a base pair, and 0 otherwise. The multiloop partition function with a single component and where position \( i \) is required to base-pair in the interval \([i, j]\) is given by

\[
ZM^1(i, j) = \sum_{r=i+\theta+1}^{j} ZB^h(i, r) \cdot e^{-(b+c(j-r))/RT}.
\]

Finally, the multiloop partition function with one or more components, having no requirement that position \( i \) base-pair in the interval \([i, j]\) is given by

\[
ZM^h(i, j) = \sum_{r=i}^{j-\theta-1} ZM^1(r, j) \cdot e^{-c(r-i)/RT} + \sum_{r=i+\theta+2}^{j-\theta-1} \sum_{k=1}^{h-1} ZM^k(i, r - 1) \cdot ZM^{1-h-k}(r, j).
\]
3.2 Number of Multiloops

As before, define the abbreviations for the partition function for hairpins

\[
ZH(i, j) = \begin{cases} 
0 & \text{if } j - i \leq \theta \\
\frac{e^{-\mathcal{H}(i,j)/RT}}{RT} & \text{else}
\end{cases}
\]

and internal loops having \( m \) multiloops

\[
ZI^m(i, j) = \sum_{i < k < \ell < j} e^{-I(i,j;k,\ell)/RT} \cdot ZB^m(k, \ell).
\]

where the sum is over \( k, \ell \) such that \( 1 \leq i < k < \ell < j \leq n \).

Define \( Z^0(i,i) = 1 \), and for \( m > 0 \), define \( Z^m(i,i) = 0 \). For the remaining base cases, define \( ZB^m(i,i) = ZM^m(i,i) = ZM1^m(i,i) = 0 \). The unconstrained partition function for secondary structures restricted to the interval \([i,j]\) that contain \( m \) multiloops is given by

\[
(3.5) \quad Z^m(i,j) = Z^m(i,j-1) + ZB^m(i,j) + \sum_{r=i+1}^{j-\theta-1} \sum_{k=0}^{m-1} Z^k(i,r-1) \cdot ZB^{m-k}(r,j)
\]

The partition function for secondary structures restricted to the interval \([i,j]\) that contain \( m \) multiloops and are closed by the base pair \((i,j)\) is given by

\[
(3.6) \quad ZB^m(i,j) = \begin{cases} 
ZH(i,j) + ZI^m(i,j) & \text{if } m = 0, \\
ZI^m(i,j) + \sum_{r=i+\theta+3}^{j-\theta-2} \sum_{k=0}^{m-1} ZM^k(i+1,r-1) \cdot ZM1^{m-k-1}(r,j-1) \cdot e^{-(a+b)/RT} & \text{if } m \geq 1.
\end{cases}
\]

provided that \( i,j \) form a base pair, and 0 otherwise. The multiloop partition function with a single component and where position \( i \) is required to base-pair in the interval \([i,j]\) is given by

\[
(3.7) \quad ZM1^m(i,j) = \sum_{r=i+\theta+1}^{j} ZB^m(i,r) \cdot e^{-(b+c(j-r))/RT}.
\]
Finally, the multiloop partition function with one or more components, having no requirement that position $i$ base-pair in the interval $[i, j]$ is given by

$$ZM^m(i, j) = \sum_{r=i}^{j-\theta-1} ZM^1(r, j) \cdot e^{-c(r-i)/RT} + \sum_{r=i+\theta+2}^{j-\theta-1} \sum_{k=0}^{m} ZM^k(i, r-1) \cdot ZM^{m-k}(r, j).$$

### 3.3 Multiloop Order

To define the multiloop order, we first define the Cantor-Bendixson derivative $D(S)$ of secondary structure $S$ on a given RNA sequence $a_1, \ldots, a_n$.

$$D(S) = \{(i, j) : (i, j) \in S, (\exists i < k < \ell < x < y < j)[(k, \ell), (x, y) \in S]\}$$

The iterates of $D$ are defined as follows: $D^{(0)}(S) = S$, $D^{(m+1)}(S) = D(D^{(m)}(S))$. The order, or multiloop depth, of secondary structure $S$ is the least $m$ such that $D^{(m)}(S) = \emptyset$. Multiloop order was first defined in a different, but equivalent manner, by Waterman [25] and extensively investigated by Nebel [17, 18].

Another manner of easily determining multiloop order of secondary structure $S$ is to determine the shape $\sigma$ of $S$ – see Giegerich [6] and Lorenz et al. [10] for the definition of shape. The derivative of a $\pi$-shape $\sigma$ can be defined as the shape resulting from $\sigma$ by the removal of all occurrences of $[\ ]$. The the order is the least number of iterations of the derivation resulting in the empty shape.

To define the partition function of multiloop order, we begin as before by defining the partition function for hairpins

$$ZH(i, j) = \begin{cases} 0 & \text{if } j - i \leq \theta \\ e^{-H(i, j)/RT} & \text{else} \end{cases}$$

and internal loops having multiloop order or depth $d$

$$ZI^d(i, j) = \sum_{i<k<\ell<j} e^{-I(i,j;k,\ell)/RT} \cdot ZB^d(k, \ell)$$
where the sum is over $k, \ell$ such that $1 \leq i < k < \ell < j \leq n$. Define $Z^0(i, i) = 1$ and for $d > 0$, define $Z^d(i, i) = 0$. For $d \geq 0$, define $ZB^d(i, i) = ZM^d(i, i) = ZM1^d(i, i) = 0$. The unconstrained partition function for secondary structures of multiloop order $d$, when restricted to the interval $[i, j]$, is given by

\begin{equation}
Z^d(i, j) = Z^d(i, j - 1) + ZB^d(i, j) + \sum_{r = i + 1}^{j - \theta - 1} \sum_{0 \leq k, \ell \leq d, \max(k, \ell) = d} Z^k(i, r - 1) \cdot ZB^\ell(r, j)
\end{equation}

The partition function for secondary structures of multiloop order $d$ when restricted to the interval $[i, j]$ and are closed by the base pair $(i, j)$ is given as follows. For $d = 0$, let

\begin{equation}
ZB^d(i, j) = ZH(i, j) + ZI^d(i, j)
\end{equation}

while for $d > 0$, let

\begin{equation}
ZB^d(i, j) = ZI^d(i, j) + \sum_{r = i + \theta + 3}^{j - \theta - 2} \sum_{0 \leq k, \ell \leq d, \max(k, \ell) = d - 1} ZM^k(i + 1, r - 1) \cdot ZM1^\ell(r, j - 1) \cdot e^{-(a+b)/RT}
\end{equation}

provided that $i, j$ form a base pair, and 0 otherwise. The multiloop partition function with a single component and where position $i$ is required to base-pair in the interval $[i, j]$ is given by

\begin{equation}
ZM1^d(i, j) = \sum_{r = i + \theta + 1}^{j} ZB^d(i, r) \cdot e^{-(b+c(j-r))/RT}.
\end{equation}

Finally, the multiloop partition function with one or more components, having no requirement that position $i$ base-pair in the interval $[i, j]$ is given by

\begin{equation}
ZM^d(i, j) = \sum_{r = i}^{j - \theta - 1} ZM1^d(r, j) \cdot e^{-c(r-i)/RT} + \sum_{r = i + \theta + 2}^{j - \theta - 1} \sum_{0 \leq k, \ell \leq d, \max(k, \ell) = d} ZM^k(i, r - 1) \cdot ZM1^\ell(r, j)
\end{equation}
3.4 RNAmultiloop, RNAhairpin

In the above three sections, I gave the recursions to calculate the partition function $Z_i$ with respect to some parameters $i$ (the number of hairpins, number of multiloops and multiloop order). On the other hand, we can calculate the total partition function $Z$ of a sequence by the McCaskill algorithm. Then we can get the Boltzmann probability with respect to this particular parameter by the formula $P_i = Z_i/Z$. In the definition of partition function, if we let the energies of all the secondary structures to be zero, we get a count of total number of secondary structures. So if we artificially define $e^x = 1$ for all $x \in \mathbb{R}$ in the above recursions, we can use exactly the same recursions to count the total number of secondary structures which has a particular parameter value. Also we can calculate the ensemble free energy with respect to this parameter by the formula: $F_i = -kT \log Z_i$.

The whole program is divided into two executables, one is called RNAhairpin, the other is RNAmultiloop.

RNAhairpin will take in an RNA sequence, and output the number of secondary structures, partition function and Boltzmann probability with respect to the number of hairpins $i$, in which $i = 0, 1, \ldots$ up to a specified threshold (default 5).

RNAmultiloop will take in a RNA sequence, and an option to calculate the multiloop number or multiloop order partition function. The output of RNAmultiloop will be the number of secondary structures, partition function and Boltzmann probability of an RNA sequence, with respect to multiloop number $i$ or multiloop order $i$, in which $i = 0, 1, \ldots$ up to a specified threshold (default 5).

I will give more explanations to the usage of the program and show sample output in chapter 6.
The intended application of these two programs is to run them on certain classes of RNAs, and output the Boltzmann probabilities with respect to the number of hairpins, number of multiloops and order of multiloops. These characteristics might be used as “fingerprints” for different classes of RNAs, and can be used as features for various RNA gene finders.

3.5 Computational Complexity

The algorithms to calculate partition functions with respect hairpin number, multiloop number and multiloop order all require time $O(n^5)$ and space $O(n^3)$. 
CHAPTER IV

RNAshapeEval: Parametric Partition Function of RNA with Respect to an Abstract Shape

In this chapter, I present the recursions to calculate the partition function of an RNA sequence with respect to an abstract shape, and the the package RNAshapeEval that is based on this algorithm.

4.1 Recursions for Shape Partition Function

In [6, 22] Giegerich and co-workers introduced the very elegant notion of RNA shape of a secondary structure. The shape $\sigma$ of secondary structure $S$ is a bracket-notation for the bifurcation structure of $S$; for instance, the shape of a typical cloverleaf tRNA structure is $\llbracket \llbracket \llbracket \rrbracket \rrbracket \rrbracket$. Five different definitions of shape were investigated in [6, 22], ranging from the coarsest grain $\pi$-shapes, illustrated in the previous tRNA example, to a fine grain notion. Although our approach could be carried out on any of the five granularities of shape, in this paper, we focus only on $\pi$-shapes, heretofore simply called shape.

From [10], the collection of all nonempty $\pi$-shapes can be formally defined as those expressions generated from the context-free grammar having start symbol $S$,
nonterminal symbols $S, T$, terminal symbols $\left[ , \right]$, and rules

\begin{equation}
S \rightarrow S \left[ T \right] \mid \left[ T \right] \tag{4.1}
\end{equation}

\begin{equation}
T \rightarrow S \left[ T \right] \mid \epsilon \tag{4.2}
\end{equation}

By induction on length, it can be shown that $G$ is a non-ambiguous grammar\footnote{A non-ambiguous grammar is one for which there do not exist two distinct parse trees for the same expression.} for the collection of all nonempty $\pi$-shapes.

In this section, we compute $Z^\sigma(i, j)$, defined as the sum of all Boltzmann factors $e^{-E(S)/RT}$, where the sum is taken over all shape $\sigma$ secondary structures $S$ of subsequence $a[i, j]$ of the input sequence $a_1, \ldots, a_n$. To the best of our knowledge, though related with the work in [6, 22], the results of this section cannot directly be obtained by any existing software.

For the base case, we define $Z^\emptyset(i, j) = 1$. And we define $Z^{[\ ]}(i, j)$ as follows. For $j - i \leq \theta, Z^{[\ ]}(i, j) = 0$, and for $j - i \geq \theta + 1$,

\begin{equation}
Z^{[\ ]}(i, j) = Z^{[\ ]}(i, j - 1) + \sum_{r=i}^{j-\theta-1} ZB^{[\ ]}(r, j). \tag{4.2}
\end{equation}

The constrained partition function is given by

\begin{equation}
ZB^{[\ ]}(i, j) = e^{-H(i, j)/RT} + \sum_{i<k<l<j} e^{-I(i;j;k,l)/RT} \cdot ZB^{[\ ]}(k, l). \tag{4.3}
\end{equation}

where $Z^{[\ ]}(i, i) = 0 = ZB^{[\ ]}(i, i)$.

The multiloop partition function with a single component and where position $i$ is required to base-pair in the interval $[i, j]$ is given by

\begin{equation}
ZM1^{[\ ]}(i, j) = \sum_{r=i+\theta+1}^{j} ZB^{[\ ]}(i, r) \cdot e^{(b+c(j-r))/RT}. \tag{4.4}
\end{equation}
The multiloop partition function with one or more components, having no requirement that position \(i\) base-pair in the interval \([i,j]\) is given by

\[
Z_M \left( i, j \right) = \sum_{r=i}^{j-\theta} Z_M \left( r, j \right) \cdot e^{-c(r-i)/RT}.
\]

Now we will show the general recursion when \(\sigma \neq [\,]\). We start by defining an indicator function for a given shape \(\sigma\), \(\text{Ind}(\sigma)\):

\[
\text{Ind}(\sigma) = \begin{cases} 
1 & \text{if } \sigma = \mu [\, \tau] \\
0 & \text{if } \sigma = [\delta] 
\end{cases}
\]

Notice in the case when \(\sigma = \mu [\, \tau]\), the expression \([\, \tau]\) can unambiguously be determined by parsing with respect to the context free grammar (4.1). Below we will use the same symbols in the recursion without further notification.

Then we can define \(Z^\sigma(i,j)\) as follows. For \(j - i \leq \theta\), \(Z^\sigma(i,j) = 0\), and for \(j - i \geq \theta + 1\),

\[
Z^\sigma(i,j) = Z^\sigma(i,j-1) + \text{Ind}(\sigma) \cdot \sum_{r=i+\theta+2}^{j-\theta-1} Z^\mu(i,r-1) \cdot Z^\tau(r,j) + (1 - \text{Ind}(\sigma)) \cdot \sum_{k=i}^{j-\theta-1} Z^\sigma(k,j)
\]

The constrained partition function is given by

\[
Z^\sigma_i(i,j) = \left[1 - \text{Ind}(\sigma)\right] \cdot \left[ \sum_{i<k<l<j} e^{-I(i,j;k,l)/RT} \cdot Z^\sigma(k,l) \right] + e^{-(a+b)/RT} \cdot \sum_{r=i+\theta+3}^{j-\theta-2} Z M^\mu(i+1+r-1) \cdot Z M^\tau(r,j-1)
\]

where \(Z^\bullet(i,i) = 0 = Z B^\bullet(i,i)\). Notice here \(\sigma = [\delta]\) and \(\delta = \mu' \tau', \tau'\) contains only one component.

The multiloop partition function with a single component and where position \(i\) is required to base-pair in the interval \([i,j]\) is given by

\[
Z M^\sigma_1(i,j) = \sum_{r=i+\theta+1}^{j} Z^\sigma(i,r) \cdot e^{-(b+c(j-r))/RT} \cdot [1 - \text{Ind}(\sigma)]
\]
The multiloop partition function with one or more components, having no requirement that position \(i\) base-pair in the interval \([i, j]\) is given by

\[
ZM^\sigma(i, j) = \text{Ind}(\sigma) \cdot \sum_{r=i+\theta+2}^{j-\theta-1} ZM^\mu(i, r - 1) \cdot ZM^1[r](r, j) \\
+ (1 - \text{Ind}(\sigma)) \cdot e^{-c(r-i)/RT} \cdot \sum_{r=i}^{j-\theta-1} ZM^1\sigma(r, j)
\]

To calculate the partition function with respect to a fixed shape, we first decompose the complex shape into its parse tree, then we do recursions from the leaves of the tree (all the leaves have the shape \([\ ]\) towards the root of the tree. We keep a list of non-redundant subshapes, so that we don’t have to do the recursion for a same subshape twice. See Figure 4.1 for the parse tree of the shape \([\ ] [\ ] [\ ] [\ ]\).
4.2 RNAshapeEval, RNAshapeMFE, RNAshapeSamp

In the last section, I presented the recursions to calculate the partition function $Z^\sigma$ of an RNA sequence with respect to an abstract shape $\sigma$. Analogous to the difference between McCaskill algorithm and Zuker MFE algorithm, it is straightforward to modify the above recursions to calculate the MFE and MFE structure with a fixed abstract shape. By adding stochastic backtrack functions analogous to the Ding-Lawrence algorithm, it is also obvious how to sample secondary structures with a
fixed shape according to their Boltzmann probabilities.

Based on the above recursions, we developed three programs: RNAshapeEval, RNAshapeMFE and RNAshapeSamp.

Given an RNA sequence $S$ and a fixed abstract shape $\sigma$, RNAshapeEval will compute the partition function with respect to this shape, and calculate the RNA subsequence and structure that have the highest probability of adopting the shape, i.e. the subsequence $S[i,j]$ such that $Z^\sigma(i,j)/Z(i,j)$ reaches highest value. This program could be used a tool to search complex structural motifs in a long RNA sequence.

RNAshapeMFE takes in an RNA sequence and an abstract shape, and find the MFE and MFE secondary structure under this shape. Since in many cases biologists have already had a rough idea about the “shape” of the folded RNA sequence, then to avoid inaccuracies due to the energy model, one can find the MFE structure under the constraint of the specified shape, so that the folding algorithm will not return a MFE secondary structure that is obviously wrong.

RNAshapeSamp takes in an RNA sequence, an abstract shape and the desired sample number $N$, it will output $N$ secondary structures that is compatible with the specified shape, and the probabilities of their occurrences are weighted by their energy. Since Sfold [2] has been a widely used tool in estimating certain structural features of a given RNA sequence, for example base pair probabilities, our tool should provide a more accurate estimate than Sfold when the basic shape of RNA is known in advance, since we have excluded the information provided by the secondary structures that are incompatible with the known RNA shape.
4.3 Computational Complexity

The algorithm to calculate partition function with respect a fixed abstract shape requires time $O(n^5)$ and space $O(n^3)$. 
CHAPTER V

RNAprofileZ: Expected Partition Function of an RNA Position Specific Scoring Matrix

In this chapter I will present the algorithm to compute the expected partition function of an RNA position specific scoring matrix (PSSM) under the Turner Nearest-Neighbor Energy Model.

5.1 Overview

Suppose that we are given a PSSM $f_1, \ldots, f_n$, where for each $1 \leq i \leq n$, $f_i$ is an arbitrary but fixed probability distribution on RNA nucleotides A,C,G,U. Thus for $x \in \{A,C,G,U\}$, $f_i(x) \in [0,1]$ and $f_i(A) + f_i(C) + f_i(G) + f_i(U) = 1$. Before continuing, we need a couple of definitions. Let $\mathbb{R}$ denote the set $\{A,C,G,U\}$ of RNA nucleotides, $\mathbb{BP}$ denote the set $\{AU, UA, CG, GC, GU, UG\}$ of RNA base pairs.

Given a PSSM, in this section, we derive recursions to efficiently compute the expected partition function

$$\sum_{a_1 \cdots a_n} Pr[a_1 \cdots a_n] \sum_{S \in S(1,n)} e^{-E(S,\vec{a})/RT}$$

Here, the sum is taken over all length $n$ nucleotide sequences $\vec{a} = a_1 \cdots a_n$, with probability $Pr[a_1 \cdots a_n] = \prod_{i=1}^{n} f_i(a_i)$, and $S(1,n)$ is the collection of all secondary structures on $[1,n]$. Note that $S(1,n)$ also includes the empty structure.
In [26], Waterman derived a formula for the asymptotic expected number of secondary structures, given a probability $p$ that any two positions can base-pair. This probability $p$, also called \textit{stickiness}, is usually taken to be $2(p_{APU} + p_{GCP} + p_{GPW})$, where $p_x$ is the compositional frequency of nucleotide $x$. If we were to additionally compute the asymptotic expected number of secondary structures having $k$ base pairs, then we could derive an asymptotic limit for the expected value of the partition function under the Nussinov-Jacobson energy model [19].

Under the Turner energy model, a naive approach would be to calculate each of the $4^n$ partition functions according to the McCaskill algorithm, and calculate the expected partition function according to the above formula. But the number of calculations will grow exponentially with the length of the PSSM, and quickly become formidable as the length grows. Next I will present a dynamic programming algorithm to solve this problem in polynomial time and space. Although there is some similarity to the method of Washietl and Hofacker [24] for \textit{consensus folding}\footnote{Consensus folding is a critical subprocedure used in the RNAz software [7] to detect noncoding RNA genes.} of aligned sequences (which involves an energy bonus for compensatory mutations displayed in columns $i$ and $j$ of a multiple alignment), to the best of our knowledge, the result described in this chapter does not appear to be known.

The idea is that instead of calculating the partition function of each sequence individually, we consider the entire space of sequence-structure pairs, then, in a similar fashion as with the McCaskill algorithm, we can calculate the expected partition function of each individual piece and multiply them together.
For a given PSSM $f_1, \ldots, f_n$, we first define the following:

$$\langle Z \rangle (i,j) = \sum_{a_i \cdots a_j} P_r[a_i \cdots a_j] \sum_{S \in S(i,j)} e^{-E(S,\vec{a})/RT}$$

$$\langle ZB \rangle (i,j,\alpha) = \sum_{a_i \cdots a_j, (i,j)=\alpha \in BP} P_r[a_i \cdots a_j] \sum_{S \in SB(i,j)} e^{-E(S,\vec{a})/RT}$$

$$\langle ZM \rangle (i,j) = \sum_{a_i \cdots a_j} P_r[a_i \cdots a_j] \sum_{S \in SM(i,j)} e^{-E(S,\vec{a})/RT}$$

$$\langle ZM1 \rangle (i,j) = \sum_{a_i \cdots a_j} P_r[a_i \cdots a_j] \sum_{S \in SM1(i,j)} e^{-E(S,\vec{a})/RT}$$

where $SB(i,j,\alpha)$ is the collection of all secondary structures $S$ on $[i,j]$ in which $(i,j) \in S$ and the base pair type of $(i,j)$ is $\alpha$, $SM(i,j)$ is the collection of all nonempty secondary structures $S$ on $[i,j]$ (i.e. having at least one component), and $SM1(i,j)$ is the collection of all secondary structures $S$ on $[i,j]$ in which $i$ base-pairs with some $k \in [i+\theta+1,j]$.

In addition, we define $\langle Z^I \rangle (i,j,k,\ell,\alpha,\beta)$ to be the expected partition function for interior loop in which the base pair type of $(i,j)$ is $\alpha$ and the base pair type of $(k,\ell)$ is $\beta$; And we define $\langle Z^H \rangle (i,j,\alpha)$ to be the expected partition function for hairpin in which $(i,j)$ is of base pair type $\alpha$. We will describe how to compute these quantities in the next section.

Now we give the full recursions for our algorithm. The unconstrained partition function is given by

(5.1) $\langle Z \rangle (i,j) = \langle Z \rangle (i,j-1) +$

$$\sum_{\alpha \in BP} P(\alpha) \cdot \langle ZB \rangle (i,j,\alpha) \cdot e^{-E_{AU}(\alpha)/RT} +$$

$$\sum_{j-\theta-1}^{\theta+1} \langle Z(r,i-1) \rangle \cdot \left[ \sum_{\alpha \in BP} P(\alpha) \cdot \langle ZB \rangle (r,j,\alpha) \cdot e^{-E_{AU}(\alpha)/RT} \right]$$
Since external loop appears in the equation, we will need to add the term for AU terminal penalty.

The constrained partition function closed by base pair \((i,j)\) is given by

\[
\langle ZB \rangle(i,j,\alpha) = \langle Z_H \rangle(i,j,\alpha) + \sum_{\beta \in \mathcal{BP}} P(\beta) \langle Z_T \rangle(i,j,k,\ell,\alpha,\beta) \cdot \langle ZB \rangle(k,\ell,\beta) + e^{-(a+b)/RT} \sum_{r=i+1}^{j-\theta-2} \langle ZM \rangle(i+1, r-1) \cdot \langle ZM1 \rangle(r, j-1).\]

Next we consider the multiloop recursions. There are two places where we must consider the AU-penalty: in the \(ZM1\) term, and later in the overall \(Z\) term. The terms \(a, b\) and \(c\) are penalty energies associated with a multiloop. The term \(E_{AU}\) is a penalty energy associated with a “wobble-pair” for a component base pair of a multiloop or external loop. A component is a base pair that is part of the multiloop. The term \(a\) is a one-time penalty for having a multiloop. The term \(b\) is a penalty for each component of a multiloop. The term \(c\) is the penalty for an unpaired nucleotide in the multiloop.

The multiloop partition function with a single component and where position \(i\) is required to base-pair in the interval \([i,j]\) is given by

\[
\langle ZM1 \rangle(i,j) = \sum_{r=i+\theta+1}^{j} \sum_{\alpha \in \mathcal{BP}} P(\alpha) \cdot \langle ZB \rangle(i, r, \alpha) \cdot e^{-(b+c(j-r))/RT} \cdot e^{-E_{AU}(\alpha)/RT}.
\]

Finally, the multiloop partition function with one or more components, having no requirement that position \(i\) base-pair in the interval \([i,j]\) is given by

\[
\langle ZM \rangle(i,j) = \sum_{r=i}^{j-\theta-1} \langle ZM1 \rangle(r,j) \cdot e^{-c(r-i)/RT} + \sum_{i+\theta+2}^{j-\theta-1} \langle ZM \rangle(i, r-1) \cdot \langle ZM1 \rangle(r,j).
\]
The next question is how to calculate the “expected” hairpin partition function and the “expected” interior loop partition function, i.e. the term $\langle Z_H(i, j, \alpha) \rangle$ and the term $\langle Z_I(i, j, k, \ell, \alpha, \beta) \rangle$ in the above recursions. We will describe this in the next two sections.

5.2 Derivation of Hairpin Recursions

In this section, we will show how to calculate value $\langle Z_H(i, j, \alpha) \rangle$. Here the hairpin loop is closed by base pair $(i, j)$ and is of type $\alpha \in \mathbb{BP}$.

Free energy and partition function calculation of a hairpin can be divided into a few cases.

Triloops and tetraloops are special cases, the energy values of which also come from direct table look-up. However to calculate the expected partition function we will need to exhaustively go through all the possible sequences of a certain hairpin type, calculate the partition functions for each loop sequence individually, and take the weighted average according to the PSSM. There are $4^3 = 64$ cases for triloops and $4^4 = 64$ cases for tetraloops.

In the general case, the energy for a hairpin loop is only dependent on the length of the hairpin, the type of the closing base pair, and the two bases in the hairpin loop that are adjacent to the closing base pair (one base on each end). In addition, the function $\langle Z_H(i, j, \alpha) \rangle$ has specified the type of the closing base pair, so to calculate the expected partition function, we only need to exhaustively go through the two dangling bases, calculate the partition function for each case and take the weighted average according to the PSSM. There are $4^2 = 16$ terms to average.
5.3 Derivation of Interior Loop Recursions

In this section, we will show how to calculate value $\langle Z_I \rangle(i,j,k,\ell,\alpha,\beta)$. Here the interior loop is closed by two base pairs $(i,j)$ and $(k,\ell)$, with $i < k < \ell < j$.

$s = s_1, \ldots, s_n$ denotes an arbitrary, but fixed RNA sequence. We denote an interior loop, given a fixed RNA sequence $s$, as $IL(i,j,k,\ell,s)$, and its energy as $E_{IL}(i,j,k,\ell,s)$. Note that $w_1 = j - i - 1$ and $w_2 = \ell - k - 1$. The symbols $\alpha$ and $\beta$ represent base pair type; i.e. $\alpha, \beta$ range over $\{0, 1, 2, 3, 4, 5\}$, where 0 represents the C-G base pair, 1 represents the G-C base pair, etc.

Free energy and partition function calculation of an internal loop can be divided into a few cases.

The simplest case concerns stacked base pairs, in which the energy values come from table look-up. Since the term $\langle Z_I \rangle(i,j,k,\ell,\alpha,\beta)$ already specifies the types of the two base pairs $(i,j), (k,\ell)$, in this case the “expected” partition function simply evolves taking the exponential of a simple energy value.

Subsequently, there are 1x1, 1x2 and 2x2 interior loops, the energy values of which also come from direct table look-up. However, to calculate the expected partition function we will need to exhaustively go through all the possible sequences of a certain loop type, calculate the partition functions for each loop sequence individually, and take the weighted average according to the PSSM. This case is computationally intensive, fortunately we only need to take a finite number of calculations for any large sequence, in which 2x2 loop is the worst case ($4^{2+2} = 256$ terms, reflecting all the possible sequence choice for a 2x2 loop).

A more general case is bulges of length $k$. According to the current Turner energy model, the energy value of a bulge loop only depends on its length $k$. This property
saves us a lot of computation, since we now only need to get the energy value through a single table look-up and take the exponential.

Now we consider the most general case, $w_1$ by $w_2$ internal loops, where both $w_1$ and $w_2$ are nonzero and one is greater than or equal to 3.

The energy is given by the formula $E_{IL}(i, j, k, l, s)$ equals

$$E_{\text{loop}}(w_1 + w_2) + E_{\text{asym}}(|w_1 - w_2|) + E_{\text{mis}}(s_i, s_j, s_{i+1}, s_{j-1}) + E_{\text{mis}}(s_k, s_{k+1}, s_{k-1}).$$

$E_{\text{loop}}$ is an energy term that is a function of the total loop size. $E_{\text{asym}}$ is an energy term that is a function of how different are the lengths of the two sides, $w_1$ and $w_2$. $E_{\text{mis}}$ is a mismatch energy term, basically a stacking or dangle term. This is the energy corresponding to how the base pairs adjacent to the bases of the internal loop stack on them.

In our recurrence relations, the sequence is not fixed. However, in order to get the recursions to work, at each step in the recurrence we will fix the nucleotides that contribute to the base pairs. All other positions will have nucleotides distributed according to the underlying weight matrix. These ends will be denoted by the terms $\alpha$ and $\beta$.

We wish to find $\langle Z_T \rangle(i, j, k, l, \alpha, \beta)$, where $\alpha$ denotes the kind of base pair at $(i, j)$ and $\beta$ denotes the kind of base pair at $(k, l)$. We define

$$\langle Z_T \rangle(i, j, k, l, \alpha, \beta) = \mathbb{E}[Z_T(i, j, k, l, \alpha, \beta, s)]$$

That is, the partition function is defined as the expected value, or the average partition function, over all sequences compatible with the base pairs $\alpha$ and $\beta$ at $(i, j)$. 
and \((k, l)\) respectively. We now describe how to compute this.

\[
\langle Z^I \rangle(i, j, k, l, \alpha, \beta) = \mathbb{E}[Z^I(i, j, k, l, \alpha, \beta, s)] = \mathbb{E}[\exp(- (E_{\text{loop}}(i, j, k, l) + E_{\text{asym}}(i, j, k, l) + E_{\text{mis}}(i, j, \alpha, s) \\
+ E_{\text{mis}}(l, k, \beta, s))/RT)]
\]

\[
= e^{-E_{\text{loop}}(i,j,k,l)/RT} \cdot e^{-E_{\text{asym}}(i,j,k,l)/RT} \cdot \mathbb{E}[\exp(-E_{\text{mis}}(i, j, \alpha, s)/RT)] \cdot \mathbb{E}[\exp(-E_{\text{mis}}(l, k, \beta, s)/RT)].
\]

Here, we have used the independence of nucleotide probability distributions at different positions, in order to write the expectation of a product as the product of expectations. Moreover, we have used the fact that the terms \(E_{\text{loop}}\) and \(E_{\text{asym}}\) are constant (and precomputed), since \(i, j, k,\) and \(l\) are fixed, hence their expectations can be replaced by the terms \(E_{\text{loop}}\) and \(E_{\text{asym}}\).

We then note that \(\mathbb{E}[\exp(-E_{\text{mis}}(i,j,\alpha,s)/RT)]\) is equal to

\[
\sum_{x,y \in \{A,C,G,U\}} P(s_{i+1} = x) \cdot P(s_{j-1} = y) \cdot e^{-E_{\text{mis}}(i,j,\alpha,x,y)/RT}.
\]

That is, the mismatch energy is determined completely by the base pair type and the neighboring nucleotides.

By now we have showed how to calculate \(\langle Z^H \rangle(i, j, \alpha)\) and \(\langle Z^I \rangle(i, j, k, \ell, \alpha, \beta)\), plugging in these terms into the recursion given in the first section of this chapter, we have completed the presentation of the algorithm for RNAprofileZ.

### 5.4 Computational Complexity

The algorithm to calculate the expected partition given a PSSM requires time \(O(n^3)\) and space \(O(n^2)\), which is on the same order as McCaskill algorithm.

---

\(^2\)While this observation holds for PSSM’s, a much more delicate approach would be necessary in the case of a first-order Markov chain.
CHAPTER VI

Biological Applications and Experimental Results

The algorithms described in the last three chapters have all been implemented in the C Programming Language, I also built web servers for each of the software packages. In this chapter, I will describe the applications of these software to biological problems.

6.1 Shape Probabilities of Rfam Full Alignment

Rfam (version 10.0 [5]) is a public data repository containing annotated ncRNA families and other structured RNA elements. For each family, it usually includes two kinds of RNA sequences: The “seed” alignment is a hand-curated alignment that contains representative members of the ncRNA family. This seed alignment is then used to build a Stochastic Context-Free Grammar (SCFG) model by the software Infernal [16], and search against known genome databases for structural homologs. The resulting sequences are included in the “full” alignment.

Then the natural question arises: How reliable is the full alignment? Can we use other methods to cross-check the results obtained by the software Infernal? To investigate this question, we took the Rfam family for purine riboswitch aptamer portions (RF00167) and use our software RNASHAPEsAMP to calculate the Boltzmann probability that the sequences will adopt the consensus shape \([ [ ] [ ] ]\) or
The seed alignment for this family has 133 sequences, and full alignment has 1241 sequences (3 other sequences from the full alignment are excluded from this experiment due to ambiguous nucleotides). We plotted the histograms of the probability score distributions for the seed alignment and full alignment of this family.

In general, the sequences from the seed alignment have mean shape probability 0.775 ± 0.280, while the sequences from the full alignment have a lower mean probability 0.758 ± 0.277. And indeed, some of the sequences from the full alignment have surprisingly low Boltzmann probabilities. For example, the RNA element with the EMBL accession number AM180355.1/2436189-2436127 (sequence CAUAUAUUUU-UGACAAUAUGGGUCAUAAAGUUUCUACGGGAUAACCGUAAAUAUUCUGACUAG) only has a probability 0.062 of adopting the shape [ [ ] [ ] ] or [ [ ] [ ] [ ] ], which casts doubts to the identity of this particular RNA element.
Although in principle the software RNAshapes[6, 22] from the Giegerich group could also be used for the same purpose, our software RNAshapeSamp is more appropriate in this context for two important reasons: First, our program does exact computation of the shape probabilities, whereas RNAshapes uses sampling to give estimates for long RNA sequences. Also RNAshapes estimates the probabilities of all the possible shapes for a single sequence at the same time, whereas our program will only calculate the Boltzmann probability for a specified shape, so our software is undoubtedly faster.

This result suggests that thermodynamic information maybe be used to help eliminate false positive alignments that is caused by only considering sequence covariation, and our software package RNAshapeEval could serve as an important supplement to the covariance-based software such as Infernal.

6.2 Hairpin and Multiloop Profiles

Each ncRNA family has its distinct sequence and structural characteristics, and ncRNA gene finders for specific RNA families usually employ many different features to help them decide whether a particular sequence element belongs to the family.

Our software RNAhairpin and RNAmultiloop can be used to generate “fingerprints” for each RNA family (especially for long and complex RNA sequences). This provides much more information than the consensus structure of the family alone, and can be built in future ncRNA gene finders as valuable features.

To illustrate the use of our program, we pick several Rfam families with different structures and compute their hairpin and multiloop “profiles”, that is, we randomly take 10 sequences from each family, compute the Boltzmann probability of having 0 hairpin, 1 hairpin, 2 hairpin, etc. for each sequence, we call this a hairpin profile of
the sequence. We take the average probabilities of having $k$ hairpins for each family, and call it the hairpin profile of this family. We plotted the hairpin profiles for RNA families U2(RF00004), tRNA(RF00005) and U4(RF00015) and the result is shown in Figure 6.2. The exact sequences used in each family could be found in the appendix.

![Figure 6.2: The Hairpin Profiles for Rfam family RF00004, RF00005, RF00015.](image)

Similarly we can make multiloop number profiles and multiloop order profiles. In Figure 6.3 and Figure 6.4 we show the multiloop number profiles and multiloop order profiles obtained from RNA families RNaseP bact a(RF00010), tmRNA(RF00023) and RRF(RF00036).

Notice for multiloop number/order profiles we deliberately choose to use RNA families with long sequence, because multiloops are “large” structural features and are energetically not so favorable, so we need a fairly long sequence to have a fair chance to see more than one multiloops appear in one sequence.

As we can see clearly from the above figures, each ncRNA family indeed has
Figure 6.3: The Multiloop Number Profiles for Rfam family RF00010, RF00023, RF00036.

distinct hairpin number, multiloop number and multiloop order profiles. Since our computation considers all possible secondary structures for each RNA sequence, our tool provides richer information about the structural properties for each RNA family.

6.3 Prediction of tRNA Secondary Structures

In many of the biological problems, we already know which family the given RNA sequence is from and have a rough idea about the “shape” of the folded RNA sequence. This prior information should certainly be taken into consideration when we are making secondary structure predictions. For example, tRNA has a well-known clover-leaf shape of [[[]][[]]], can we use this knowledge to help us improve the accuracy of secondary structure prediction?

To answer this question, we took all of the 482 tRNA sequences from Sprinzl’s data collection[21] (since our algorithms currently does not handle chemically modified nucleotides, to make a fair comparison, we choose to ignore the information about
chemically modified nucleotides and treat them as regular nucleotides), and compare the performance of three different secondary structure prediction strategies.

1) Zuker’s MFE algorithm. We used the RNAfold [8] program from the Vienna RNA package.¹

2) shapeMFE. We use our own software RNAshapeMFE, and it calculates the MFE structure with the specified shape \([ [ ] [ ] [ ] ]\).

3) shapeMEA. We first use our program RNAshapeSamp to sample 1000 secondary structures with the shape \([ [ ] [ ] [ ] ]\), and estimate base pair probabilities from these sample secondary structures. Then we calculate the Maximum Expected Accuracy (MEA) secondary structure using these base pair probabilities.

The notion Expected Accuracy is first introduced in [9], and the formal definition is the following:

¹Coaxial stacking has significant energy contributions in tRNAs, so in these cases, RNAstructure [14] is expected to perform much better than RNAfold. However, since we haven’t implemented coaxial stacking in the RNAshapeEval package, for fair comparison we don’t consider coaxial stacking when using Zuker’s MFE algorithm.
Table 6.1: Performance Comparison of MFE, shapeMFE and shapeMEA method on tRNA data

<table>
<thead>
<tr>
<th></th>
<th>MFE</th>
<th>shapeMFE</th>
<th>shapeMEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>sensitivity</td>
<td>0.558 ± 0.251</td>
<td>0.727 ± 0.212</td>
<td>0.765 ± 0.200</td>
</tr>
<tr>
<td>PPV</td>
<td>0.600 ± 0.237</td>
<td>0.788 ± 0.194</td>
<td>0.765 ± 0.186</td>
</tr>
</tbody>
</table>

Given a secondary structure $S = s_1 s_2 ... s_n$, the Expected Accuracy of a secondary structure equals

$$\sum_{(i,j) \in S, 1 \leq i < j \leq n} P(i, j) + \alpha \sum_{1 \leq k \leq n} Q(k)$$

Here $P(i, j)$ is probability of $i$ and $j$ base pair, $\alpha$ is a parameter to tune the weights between base pair probability and unpaired probability (in this experiment we let $\alpha = 1$), and

$$Q(k) = 1 - \sum_{j < k} P(j, k) - \sum_{k < j} P(i, k), 1 \leq k \leq n$$

i.e. $Q(k)$ is the probability that position $k$ is unpaired.

The Maximum Expected Accuracy(MEA) is a method that takes the the highest probable (defined by maximizing the Expected Accuracy) secondary structure as the best guess. This measure has been shown to outperform the tradition Minimum Free Energy(MEA) secondary structure prediction method in a number of settings, an example of this is [11].

We calculated the sensitivity and Positive Predicted Value(PPV) of tRNA structure predictions using these three strategies, and the result is shown in the table 6.1.

Sensitivity and PPV are defined as the following:

$$sensitivity = \frac{\text{Number of Correctly Predicted Base Pairs}}{\text{Number of Base Pairs in the Native Secondary Structure}}$$

$$PPV = \frac{\text{Number of Correctly Predicted Base Pairs}}{\text{Total Number of Predicted Base Pairs}}$$
As we can see clearly from the table, compared with MFE method, both shapeMFE and shapeMEA algorithms greatly improve the prediction accuracy of tRNAs, this is not surprising given the fact that we have used more information than Zuker’s MFE algorithm. Compared with shapeMFE, shapeMEA reaches higher sensitivity, though slightly lower PPV.

To consider the implication of this experiment in a broader perspective, the knowledge of “shape” of the RNAs could greatly reduce the search space for the RNA secondary structure prediction algorithms, thus resulting an improvement in the prediction performance. So the shape information should be used whenever available, and our software RNAshapeSamp and RNAshapeMFE could be used for this purpose.

Our sampling algorithm could be thought as an extension of the Ding-Lawrence sampling algorithm under a fixed shape. Since the Ding and Lawrence algorithm [2] has been widely used to biological community to estimate certain structural features, such as base pair probabilities, unpaired regions etc., our algorithm should be equally useful when the shape of the RNA sequence is already known in advance.

Also although we only tested one RNA family in this experiment, the Maximum Expected Accuracy(MEA) prediction method has been shown to have higher PPV than MFE method in general[11], it is reasonable to expect shapeMEA method will achieve similar success when the shape information about the RNA sequence is available.

### 6.4 Prediction of Pseudoknotted Structures

Due to considerations of computational complexity[12], as well as the fact that there is little data on experimentally measured pseudoknot energy values, most of the
current RNA structure prediction software cannot handle pseudoknotted structures. In this section, we illustrate an interesting use of our program RNAshapeSamp in providing valuable information on pseudoknotted RNA secondary structures.

We take the RNA element with EMBL accession number X85253.1:682-769 from the HDV ribozyme family (RF00094). The sequence and the secondary structure of this RNA element is:

```
AUGGCCGGCAUGGUGCAGCCUCUUGGCGGCUGGCAACAUUCGGACGACCUGCACUGGUAAUUGCGAAUGGACCCA

..(((((((..AAAAAAA(((.BB.....))))))))))bb....(((((((((((.(...)))))))).)))).....aaaaaaa..
```

To make sense of this structure, we can think of this pseudoknotted structure to be the following two valid secondary structures on two different pages, and get superimposed:

```
..(((((((.........(((........))))))))))......(((((((((((.(...)))))))).))))..............
```

```
...........(((((((....((...............))......................................)))))))..
```

Notice the first secondary structure has a shape [ ] [ ], while the second one has the shape [ ].

Under each shape, we use RNAshapeSamp to 1000 secondary structures, and calculate the MEA secondary structures, and we obtain the following results:

```
..(((((((.........(((........))))))))))(((...((((((((((((....)))))))).))))..........))).
```

```
...........(((((((.............(((((.............((((((((....))))))))...)))))..))))))).
```

As we can see clearly, the first MEA structure picks up most of the base pairs on page 1 of the pseudoknot, where the second MEA structure captures most of the base pairs on page 2 of the pseudoknot.

Notice the energetic difference of these two MEA structures are 3.67 kcal/mol, which means one structure will happen 500 fold more often in the sampled secondary structures if we use the traditional Ding-Lawrence algorithm, not to mention that
there is no way for them to correctly identify this particular structure as the page 2 secondary structure.

This experiment suggests that our tool might be useful in predicting certain type of RNA pseudoknots. But of course, what we are showing here is only an anecdotal result, and we still need to do more systematic benchmark with other algorithms in pseudoknotted secondary structure prediction.
CHAPTER VII

Summary

7.1 Future Work

There are multiple directions that the work presented in the thesis could be extended.

For the RNA parametric partition function calculation, we might be able to extend our algorithms to other structural features such as the number of bulges, number of multiloop junctions, etc.

An important extension to the RNAshapeEval package would be to implement the exact calculation of base pair probabilities with a fixed shape. In the current version of the software, to calculate the Maximum Expected Accuracy (MEA) secondary structure with a certain shape, we first sample a large number of secondary structures within the shape, and then estimate the base pair probabilities. The exact computation of the base pair probabilities will promise to be more accurate and faster than the sampling method.

In Chapter 5 we presented the algorithm to compute the expected partition function of an RNA position specific scoring matrix (PSSM) (or profile) under the Turner Nearest-Neighbor Energy model. This is an interesting theoretical result, however by assuming the RNA sequences come from a PSSM we ignored the interactions between nucleotides, which could be very important in considering RNA-related problems.
Hopefully the algorithm presented in this thesis could serve as a useful stepping stone to inspire the algorithms to calculate the expected partition functions for a Markov chain or even Stochastic Context-Free Grammar.

On the technical side, it would be very useful to implement extra features to the current software such as considering energy contributions from dangling ends, excluding isolated base pairs (helices of length 1), and handling variable temperatures, etc.

### 7.2 Discussion and Conclusion

To sum up, in this thesis I present three highly non-trivial RNA parametric partition function calculation algorithms. They include 1) RNAhairpin and RNAmultiloop, which calculate partition functions and Boltzmann probabilities with respect to hairpin number, multiloop number and multiloop order, 2) RNAshapeEval, which finds the RNA subsequence that has the highest probability of adopting a certain abstract shape and fold it within the shape; RNAshapeSamp, which calculates the probability of a sequence adopting a fixed shape, as well as samples secondary structures under the shape according to their Boltzmann probabilities; RNAshapeMFE, which folds the RNA sequence under the specified shape, and 3) RNAprofileZ, which calculates the expected partition function and ensemble free energy given an RNA position weight matrix.

I also describe the application of these software in biological problems, including evaluating purine riboswitch aptamer full alignment sequences to adopt their consensus shape, building hairpin and multiloop profiles for certain Rfam families, tRNA and pseudoknotted RNA secondary structure predictions. These three sets of algorithms might prove to be useful in a wide range of biological applications such as
searching genomic databases for complex structural motifs, ncRNA gene finders for specific RNA families, canonical and pseudoknotted secondary structure predictions for a known RNA family etc.


.1 EMBL Accession Number of sequences that are used to generate hairpin profiles

RF00004
AAPU01010615.1:193731-193537
AANS01001054.1:6664-6467
AF053589.1:90-279
X56454.1:125-277
EF140768.1:2-192
Z36100.1:1808-1619
AF325695.1:199-9
AAKD03000004.1:610702-610510
AACS01000161.1:56642-56830
AY661656.1:2159-2358
RF00005
K01389.1:345-433
M17309.1:99-171
X61674.1:1095-1008
X13888.1:63-151
D31785.1:714-785
AE0099773.1:7700-7629
X02444.1:95-15
L13782.1:442-515
AB042240.3:36390-36319
J01390.1:13362-13432
RF00015
EMBL Accession Number of sequences that are used to generate multiloop number and multiloop order profiles

RF00010
AACK01000018.1:26603-26989
BA000012.4:1306844-1306445
CP000362.1:1906740-1906341
CP000555.1:489173-489511
AM167904.1:3141205-3140801
AAVS01000005.1:163696-164059
AP008229.1:4089958-4089609
X73135.1:43-490
CP000089.1:575353-575735
CP000781.1:1989320-1989711
RF00023
CP000653.1:3374760-3375121
AP006618.1:4642707-4642341
BA000008.3:138204-137780
AAWA01000016.1:115195-115569
CP000478.1:4622994-4623346
AAWN01000028.1:25192-25539
CP000127.1:1307217-1307576
AJ965256.1:1165488-1165838
AE017198.1:812877-813242
AY129337.1:100059-100494
RF00036
AF217162.1:1537-1873
AF217173.1:1552-1888
AF321145.1:1327-1663
K02011.1:1705-2041
AJ286340.1:1537-1876
M38430.1:1673-2009
U36876.1:1534-1870
AJ418518.1:1549-1885
AJ418528.1:1477-1813
U36859.1:1501-1837