RNA folding pathways and kinetics using 2D energy landscapes

Evan Senter · Ivan Dotu · Peter Clote

Received: date / Accepted: date

Abstract RNA folding pathways play an important role in various biological processes, such as (i) the hok/sok (host-killing/suppression of killing) system in E. coli to check for sufficient plasmid copy number, (ii) the conformational switch in spliced leader (SL) RNA from Leptomonas collosoma, which controls trans splicing of a portion of the 5′ exon, and (iii) riboswitches – portions of the 5′ untranslated region of messenger RNA that regulate genes by allostery.

Since RNA folding pathways are determined by the energy landscape, we describe a novel algorithm, FFTbor2D, which computes the 2D projection of the energy landscape for a given RNA sequence. Given two metastable secondary structures A, B for a given RNA sequence, FFTbor2D computes the Boltzmann probability $p(x, y) = Z_{x,y}/Z$ that a secondary structure has base pair distance $x$ from A and distance $y$ from B. Using polynomial interpolation with the fast Fourier transform, we compute $p(x, y)$ in $O(n^5)$ time and $O(n^2)$ space, which is an improvement over an earlier method, which runs in $O(n^7)$ time and $O(n^4)$ space.

FFTbor2D has potential applications in synthetic biology, where one might wish to design bistable switches having target metastable structures A, B with favorable pathway kinetics. By inverting the transition probability matrix determined from FFTbor2D output, we show that L. collosoma spliced leader RNA has larger mean first passage time from A to B on the 2D energy landscape, than 97.145% of 20,000 sequences, each having metastable structures A, B.

Source code and binaries are freely available for download at http://bioinformatics.bc.edu/clotelab/FFTbor2D. The program FFTbor2D is implemented in C++, with optional OpenMP parallelization primitives.

Keywords RNA · Boltzmann partition function · FFT · polynomial interpolation · dynamic programming · kinetics

PACS 87.14.gn · 05.10.-a · 82.39.-k · 82.39.Pj

Research supported by National Science Foundation grants DMS-1016618 and DBI-1262439.

E. Senter, I. Dotu and P. Clote
Department of Biology, Boston College, Chestnut Hill, MA 02467, USA. Tel.: +1-617-552-1332. Fax: +1-617-552-2011. E-mail: clote@bc.edu.
1 Introduction

RNA folding pathways play an important role in biological processes. For instance, in the hok/sok (host-killing/suppression of killing) system [11], the transition between two metastable RNA structures determines the fate of a cell as follows. The hok gene of E. coli and other bacteria codes a small (52 amino acid) toxin causing irreversible damage to the cell membrane. It has been shown that hok-mRNA is constitutively expressed from a weak promoter, while the rapidly degraded sok-RNA is constitutively expressed from a strong promoter. The hok-mRNA is initially inactive, since a foldback sequesters the Shine-Dalgarno sequence; however, slow exonucleolytic processing digests the last $\approx 40$ nt of the 3’ end of hok-mRNA, transforming the molecule into its active form in which the Shine-Dalgarno sequence is no longer sequestered. If R1 plasmids of E. coli are present in sufficient copy number, then a portion of the 64 nt sok-RNA, which is complementary to hok-mRNA leader region, binds to the active conformation of hok-mRNA, thus causing degradation of the complex by RNase III [11]. If plasmids are not present in sufficient copy number, then the cell is killed by hok toxin, thus ensuring fitness of the daughter cells.

In the case of spliced leader (SL) RNA from certain trypanosomes and nematodes, a portion of the 5’ exon is donated to another mRNA by trans splicing. Intermediate structures appear to be important in the process of splicing, as shown by LeCuyer and Crothers [12], who performed stopped-flow rapid-mixing and temperature-jump measurements of the kinetics for the structural transition between two low energy structures of SL RNA from Leptomonas collosoma. Conformational switches are thought not only to play a role in such trans splicing, but as well in transcriptional and translational regulation, protein synthesis, and mRNA splicing.

For these reasons, substantial experimental and computational work has been done on folding pathways and the kinetics of RNA folding; below, we cite only a small sample of the work in this area. On the experimental side, Neupane et al. [19] applied single-molecule force spectroscopy of the add adenine riboswitch, to show how folding relates to gene regulation; see also [25,2,17]. On the computational side, Morgan and Higgs [18] appear to have been among the first to have considered the computational problem of determining optimal and near-optimal folding pathways between two metastable secondary structures $A, B$ of a given RNA sequence. In [5], Flamm developed an event-driven Monte Carlo simulation, Kinfold, to determine expected folding time between two reference structures, including the mean first passage time to fold into the minimum free energy structure. Using RNAsubopt [26], Flamm et al. [6] designed the exponential time exact algorithm, barriers, that computes the optimal folding pathway between metastable structures $A$ and $B$. Since barriers may not converge, due to its reliance on the enumeration of possibly exponentially many structures, in [4] we developed a local search (tabu) method that rapidly returns near-optimal folding pathways; in benchmarking results [4], local search was shown to outperform various methods, including direct and indirect path methods of Morgan and Higgs [18], breadth-first search.
with bounded lookahead, Findpath [7], and the exponential time, exact method barriers [6], in producing near-optimal pathways within reasonable time. There are far too many contributions to kinetics and folding pathways to adequately survey here; the references [27,22,10,29,13] give some idea of the variety of methods.

1.1 Preliminaries

A secondary structure for a given RNA nucleotide sequence $s = s_1, \ldots, s_n$ is a set $S$ of Watson-Crick or wobble base pairs $(i,j)$, containing neither base triples nor pseudoknots. The number of base pairs in $S$ is denoted by $|S|$. The secondary structure $S$ is compatible with $s$ if for every base pair $(i,j)$ in $S$, the pair $(s_i, s_j)$ is contained in the set $B$ of six canonical (Watson-Crick and wobble) base pairs. Throughout this paper, by structure, we always mean a secondary structure which is compatible with an arbitrary, but fixed RNA sequence $s$.

If $A, B$ are secondary structures of $s$, then the base pair distance, $d_{BP}(A, B)$, is defined to be the $|A - B| + |B - A|$, i.e. the number of base pairs belonging to one structure but not the other. Structures $S, T$ are said to be $k$-neighbors if $d_{BP}(S, T) = k$. In [8], we described recursions for an $O(n^3)$ time and $O(n^3)$ algorithm, RNAbor, that computes the Boltzmann probability $p_k$ of structures to have base pair distance $k$ from a given reference structure $A$. In [21], we described an $O(n^3)$ time and $O(n^2)$ space algorithm, by using polynomial interpolation with the fast Fourier transform (FFT). In [14], Lorenz et al. generalized the recursions of RNAbor [8] to yield recursions for an $O(n^7)$ time and $O(n^4)$ space algorithm, RNA2Dfold, that computes the Boltzmann probability $p(x, y)$ that a structure has base pair distance $x$ from reference structure $A$, and distance $y$ from another reference structure $B$. The goal of this paper is to describe a new algorithm, FFTbor2D, using polynomial interpolation with the FFT, to reduce the worst case complexity of RNA2Dfold to $O(n^5)$ time and $O(n^2)$ space. As well, we provide an illuminating application by computing the mean first passage time between metastable structures $A, B$ of spliced leader RNA from L. collosoma on the 2D energy landscape computed by FFTbor2D.

The general idea of using interpolation to compute partition function values was first suggested by Waldispühl and Ponty in the context of the RNAmutants program [23]. Subsequently, we used the Fast Fourier Transform (FFT) in our algorithm FFTbor [21] to interpolate the probabilities $p_k$ that structures from the Boltzmann ensemble have base pair distance $k$ from a target structure $S^*$. This paper extends the result from [21] to two dimensions; i.e. in the algorithm FFTbor2D, we interpolate the probabilities $p(x, y)$ that structures from the Boltzmann ensemble have base pair distance $x$ [resp. $y$] from target structure $A$ [resp. $B$].

1.2 Plan of paper

The plan for the rest of this paper is as follows. In Section 2, we develop the quintic $O(n^5)$ time and quadratic $O(n^2)$ space algorithm FFTbor2D, which uses dynamic programming to evaluate a complex polynomial $Z(x)$ at quadratically many complex roots of unity, and then use the fast Fourier transform (FFT) to compute the coefficients of $Z(x)$ by polynomial interpolation. The coefficients of
Z(x) yield the 2D energy landscape for a given RNA sequence. Moreover, by exploiting parity and complex conjugates, we obtain an additional reduction of time by a factor of 4. Although this section is rather technical, the main result entails a significant improvement in the algorithm run time. In Section 3, we present benchmarking results, comparing the run times of FFTbor2D and RNA2Dfold (developed by Lorenz et al. [14]). In Section 4, we apply FFTbor2D to determine the mean first passage time (MFPT) along the 2D energy grid in folding between two metastable structures of L. collosoma spliced leader RNA. Finally, in Section 5, we describe differences in the algorithms RNA2Dfold and and FFTbor2D, and mention relative strengths of each software depending on the envisioned application.

2 Polynomial interpolation using the FFT

For expository clarity, we describe FFTbor2D and all recursions in terms of the Nussinov energy model [20], where the energy \( E_0(i, j) \) of a base pair \((i, j)\) is defined to be \(-1\), and the energy \( E(S) \) of a secondary structure \( S \) is \(-1\) times the number \(|S|\) of base pairs in structure \( S \). Nevertheless, the implementation of FFTbor2D involves the full Turner energy model [28], where free energy \( E(S) \) depends on negative, stabilizing energy contributions from base stacking, and positive, destabilizing energy contributions due to loss of entropy in loops.

Given reference secondary structures \( A, B \) of a given RNA sequence \( s = s_1, \ldots, s_n \), our goal is to compute

\[
Z_{1,n}^{x,y} = \sum_{S \text{ such that } d_{BP}(S, A) = x, d_{BP}(S, B) = y} e^{-\frac{E(S)}{R T}}
\]

for all \( 0 \leq x, y < n \), where \( R \) is the universal gas constant, \( T \) absolute temperature, \( E(S) \) denotes the free energy of \( S \), and \( S \) ranges over all secondary structures that are compatible with \( s \). As mentioned, we emphasize that for expository reasons alone, the Nussinov energy model is used in the recursions in this paper, although full recursions and the implementation of FFTbor2D involve the Turner energy model.

For any secondary structure \( S \) of \( s \), and any values \( 1 \leq i \leq j \leq n \), the restriction \( S_{[i, j]} \) is defined to be the collection of base pairs of \( S \), lying within interval \([i, j]\); i.e. \( S_{[i, j]} = \{(k, \ell) : i \leq k < \ell \leq j\} \). In [14], Lorenz et al. generalized the dynamic programming recursions of our earlier work [8], to yield recursions for the partition function \( Z_{i,j}^{x,y} \) in equation (1). In the context of the Nussinov model, \( Z_{i,j}^{x,y} \) is equal to

\[
Z_{i,j-1}^{(x-\alpha_0), (y-\beta_0)} + \sum_{s_k, s_j \in B, i \leq k < j} e^{-\frac{E_0(k, j)}{R T}} \sum_{u+v=x-\alpha(k)} \sum_{u'+v'=y-\beta(k)} Z_{k,k-1}^{u,v} Z_{k+1,j-1}^{u',v'}
\]

where \( \alpha_0 = 1 \) if \( j \) is base paired in \( A_{[i,j]} \) and \( 0 \) otherwise, \( \beta_0 = 1 \) if \( j \) is base paired in \( B_{[i,j]} \) and \( 0 \) otherwise, \( E_0(k, j) = +\infty \), and \( \alpha(k) = d_{BP}(A_{[i,j]} \cup A_{[k+1,j-1]} \cup \{(k, j)\}) \), and \( \beta(k) = d_{BP}(B_{[i,j]} \cup B_{[k+1,j-1]} \cup \{(k, j)\}) \).
2.1 Recursions to compute the polynomial $Z_{i,j}(x)$

Given RNA sequence $s = s_1, \ldots, s_n$ and two arbitrary, but fixed reference structures $A, B$, we define the polynomial

$$Z(x) = \sum_{r=0}^{n-1} \sum_{s=0}^{n-1} z_{rn+s} x^{rn+s}$$

where (constant) coefficients

$$z_{rn+s} = Z_{r,n} = \sum_{S \text{ such that } d_{BP}(S,A) = r, d_{BP}(S,B) = s} e^{-\frac{E(S)}{RT}}$$

where $E(S)$ denotes the free energy of $S$. If we evaluate the polynomial $Z(x)$ at $n^2$ distinct pairs of values $a_0, \ldots, a_{n^2-1}$ in

$$Z(a_0) = z_0, \ldots, Z(a_{n^2-1}) = z_{n^2-1},$$

then Lagrange polynomial interpolation guarantees that we can determine the coefficients $c_{rn+s}$ of $Z(x)$, for $0 \leq r, s < n$. Due to technical difficulties concerning numerical robustness, we will perform polynomial interpolation by using Vandermonde matrices and the fast Fourier transform (FFT).

The following theorem shows that a recursion, analogous to equation (2), can be used to compute the polynomial $Z_{i,j}(x)$ defined by

$$Z_{i,j}(x) = \sum_{r=0}^{n-1} \sum_{s=0}^{n-1} z_{rn+s(i,j)} x^{rn+s} = \sum_{k=0}^{n^2-1} z_k(i,j)x^k$$

where

$$z_{rn+s}(i,j) = Z_{i,j}^{r,s} = \sum_{S \text{ such that } d_{BP}(S,A_{i,j}) = r, d_{BP}(S,B_{i,j}) = s} e^{-\frac{E(S)}{RT}}.$$

Here, in the summation, $S$ runs over structures on $s_i, \ldots, s_j$, which are $r$-neighbors of the restriction $A_{i,j}$ of reference structure $A$ to interval $[i,j]$, and simultaneously $s$-neighbors of the restriction $B_{i,j}$ of reference structure $B$ to interval $[i,j]$.

**Theorem 1:** Let $s_1, \ldots, s_n$ be a given RNA sequence. For any integers $1 \leq i < j \leq n$, let

$$Z_{i,j}(x) = \sum_{r=0}^{n-1} \sum_{s=0}^{n-1} z_{rn+s} x^{rn+s}$$

where

$$z_{rn+s}(i,j) = Z_{i,j}^{r,s}.$$

Inductively we define $Z_{i,j}(x)$ to equal

$$Z_{i,j}(x) = Z_{i,j-1}(x) \cdot x^{\alpha_{n+\beta_0} + \sum_{i \leq k < j} \left( e^{-\frac{E(k,i)}{RT}} \cdot Z_{i,k-1}(x) \cdot Z_{k+1,j-1}(x) \cdot x^{\alpha(k)n+\beta(k)} \right)}$$
where \( a_0 = 1 \) if \( j \) is base-paired in \( A_{i,j} \) and 0 otherwise, \( \beta_0 = 1 \) if \( j \) is base-paired in \( B_{i,j} \) and 0 otherwise, and \( a(k) = d_{BP} (A_{i,j}, A_{i,k-1} \cup A_{[k+1,j-1]} \cup \{(k,j)\}) \), \( \beta(k) = d_{BP} (B_{i,j}, B_{i,k-1} \cup B_{[k+1,j-1]} \cup \{(k,j)\}) \). The proof is given in supplementary information.

Note that if one were to compute all terms of the polynomial \( Z_{1,n}(x) \) by explicitly performing polynomial multiplications, then the computation would require \( O(n^3) \) time and \( O(n^3) \) space, the same time complexity of [14]. Instead of explicitly performing polynomial expansion in \( x \), we instantiate \( x \) to a complex number \( \rho \in \mathbb{C} \), and apply the following recursion, by setting \( Z_{i,j}(\rho) \) equal to

\[
Z_{i,j-1}(\rho) \cdot \rho^{a_n + \beta_0} + \sum_{\{s_k, s_j\} \in \mathbb{R}, \ i \leq k < j} \left( e^{-\frac{\pi i (s_k, s_j)}{n}} \cdot Z_{i,k-1}(\rho) \cdot Z_{k+1,j-1}(\rho) \cdot \rho^{\alpha(k) n + \beta(k)} \right)
\]

In this fashion, we can compute \( Z(\rho) = Z_{1,n}(\rho) \) in \( O(n^3) \) time and \( O(n^2) \) space. For \( n^2 \) distinct complex numbers \( \rho_i \) where \( 0 \leq i \leq n^2 - 1 \), we can compute and save only the values \( Z(\rho_0), \ldots, Z(\rho_{n^2-1}) \), each time re-using the \( O(n^2) \) space for the next computation of \( Z(\rho_i) \). It follows that the computation resources used to determine the (column) vector

\[
Y = (y_0, \ldots, y_{n^2-1})^T = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n^2-1} \end{pmatrix}
\]

where \( y_0 = Z(\alpha_0), \ldots, y_{n^2-1} = Z(\alpha_{n^2-1}) \) are thus quintic time \( O(n^5) \) and quadratic space \( O(n^2) \).

### 2.2 Polynomial interpolation

Our plan is to determine the coefficients of the polynomial \( Z(x) \) in equation (3) by polynomial interpolation. For reasons of numerical stability, we instead determine the coefficients of the polynomial \( p(x) \), defined by

\[
p(x) = \sum_{r=0}^{n-1} \sum_{s=0}^{n-1} p_{r+n+s} x^r n^s = \sum_{r=0}^{n-1} \sum_{s=0}^{n-1} \frac{z_{r+n+s}}{z} x^r n^s,
\]

where the fast Fourier transform (FFT) is used to implement the interpolation of the coefficients using the inverse discrete Fourier transform (DFT), as described in Section 2.6. The following pseudocode describes how to compute the \( m \) most significant digits for probabilities \( p_{r+n+s} = \frac{Z_{r+n+s}}{Z} \). It is well-known that the FFT requires \( O(N \log N) \) time to solve the inverse discrete Fourier transform for a polynomial of degree \( N \). In our case, \( N = n^2 \), and so line 6 involving the FFT requires time \( O(n^2 \log n) \).

The pseudocode for the algorithm to compute \( p(x) \) is given in Figure 1. In the next section, we explain a highly non-trivial improvement of this algorithm to reduce time by a factor of 4.
2D energy landscapes of RNA

Algorithm for \texttt{FFTbor2D}

\textbf{Input:} RNA sequence \( s = s_1, \ldots, s_n \), and distinct secondary structures \( A, B \) of \( s \), and integer \( m \).

\textbf{Output:} Probabilities \( p_{r,s} = p(r,s) = Z_{r,s}^{1,n} / Z \) to \( m \) significant digits for \( x, y = 0, \ldots, n - 1 \).

Let \( i \) denote \( \sqrt{-1} \), \( \alpha = \exp(\frac{2\pi i}{n}) \) and \( \alpha^k = \exp(\frac{2\pi i k}{n}) \).

1. for \( k = 0, \ldots, n^2 - 1 \)
2. compute the \( k \)th roots of unity \( \alpha^k \)
3. for \( k = 0, \ldots, n^2 - 1 \)
4. compute \( y_k = 10^{m\cdot\frac{\pi k}{n}} \) //normalize \( y_k \)
5. compute \( p_0, \ldots, p_{n^2-1} \) by FFT
6. for \( k = 0 \) to \( n^2 - 1 \)
7. \( p_k = \lfloor 10^m \cdot pk \rfloor \cdot \frac{1}{10^m} \)
8. //truncate to \( m \) most significant digits

\textbf{Fig. 1} Pseudocode to compute the \( m \) most significant digits for probabilities \( p_{r,s} = Z_{r,s}^{1,n} / Z \). In our implementation, due to numerical stability issues in the FFT engine, precision parameter \( m \) has an upper bound of 8—only the \( m = 8 \) most significant digits are computed with \texttt{FFTbor2D}. (Note that the software actually uses base 2 precision parameter, with maximum of 27, where \( 2^{27} \approx 10^{8} \).) It is well-known that the FFT requires \( O(N \log N) \) time to solve the inverse discrete Fourier transform for a polynomial of degree \( N \). In our case, \( N = n^2 \), and so the FFT requires time \( O(n^2 \log n) \).

2.3 Speed-up by factor of 4

Recall that if \( a + bi \) is a complex number, where \( a, b \) are real values and \( i \) denotes \( \sqrt{-1} \), then the complex conjugate of \( a + bi \), denoted by \( \overline{a + bi} \) is defined to be \( a - bi \). Recall that a complex \( n \)th root of unity is a number whose \( n \)th power equals one. Moreover, \( e^{2\pi i/n} \) is the principal complex \( n \)th root of unity; i.e. \( \{e^{2\pi ik/n} : k = 0, \ldots, n - 1 \} \) is a set of pairwise distinct \( n \)th roots of unity. For notational reasons below, we will write ‘\( n \)th root of unity’ instead of ‘\( n \)th root of unity’. We have the following.

\textbf{Lemma 1:} Let \( A, B \) denote two distinct, arbitrary but fixed, secondary structures of RNA sequence \( s \), let \( S \) range over all secondary structures of \( s \), and let \( d_0 \) denote \( d_{BP}(A,B) \). If \( x = d_{BP}(A,S) \) and \( y = d_{BP}(S,B) \), then \( y \in \{d_0 - x + 2k : k = 0, \ldots, x \} \).

It follows that if \( x = d_{BP}(A,S) \) and \( y = d_{BP}(S,B) \), then the only possible values for \((x,y)\) are \((0, d_0), (1, d_0 - 1), (1, d_0 + 1), (2, d_0 - 2), (2, d_0), (2, d_0 + 2), (3, d_0 - 3), (3, d_0 - 1), (3, d_0 + 1), (3, d_0 + 3), \ldots \). As a corollary, we have the \textit{parity condition}, that

\begin{equation}
\text{d}_{BP}(A,S) + \text{d}_{BP}(S,B) \equiv \text{d}_{BP}(A,B) \mod 2
\end{equation}

\(10\)

first noticed in [14], as well as the triangle inequality \( \text{d}_{BP}(A,S) + \text{d}_{BP}(S,B) \geq \text{d}_{BP}(A,B) \) for base pair distance, probably folklore. Lorenz et al. [14] exploited the parity condition and the triangle inequality by using sparse matrix methods to improve on the efficiency of the naive implementation of the \( O(n^2) \) time and \( O(n^3) \) space algorithm to compute the partition function, \( Z_{1,n}^{r,s} \), and minimum free energy structure, \( MFE_{1,n}^{r,s} \), over all structures having base pair distance \( r \) to \( A \) and \( s \) to \( B \). The following lemma is not difficult to establish.
Lemma 2: If $Z(x)$ is the complex polynomial defined in equation (3), then for any complex $n$th root of unity $\alpha$, it is the case that $Z(\alpha) = Z(\overline{\alpha})$.

Lemma 3: Let $Z(x)$ be defined by equation (3), and let $\alpha \in \mathbb{C}$ be any complex number. If the base pair distance between reference structures $A, B$ is even, then $Z(-\alpha) = Z(\alpha)$, while if the distance is odd, then $Z(-\alpha) = -Z(\alpha)$.

Lemma 4: Suppose that $M$ is evenly divisible by 4, $\nu = \exp\left(\frac{2\pi i}{M}\right)$ is the principal $M$-root of unity, and $\frac{M}{4} < k \leq \frac{2M}{4}$. Then
$$\nu^k = -\left(\nu^{-(M/2-k)}\right) = -\nu^{M/2-k}.$$

Lemma 1 is proved by simple induction; Lemma 2 is proved by a computation involving binomial coefficients; Lemma 3 is immediate by the parity observation above, resulting from Lemma 1; Lemma 4 is elementary, relying on Euler’s formula and trigonometric addition formulas. Details proofs of Lemmas 2, 3, 4 can be found in supplementary information.

Lemma 1 entails that either all even coefficients, or all odd coefficients of $Z(x)$ are zero, and so by a variable change described in detail below, we require only half the number of evaluations of $Z(x)$, in order to perform polynomial interpolation. Lemma 2 entails that we require only half again the number of evaluations of $Z(x)$, since the remainder can be inferred by taking the complex conjugate. Lemma 1 and Lemma 2, along with a precomputation of powers of the complex roots of unity, lead to a large performance speed-up in our implementation of FFTbor2D – indeed, by a factor of 4 or more. Though the intuitive idea of how to obtain this speedup by a factor of four may be apparent, the technical details leading to the pseudocode of FFTbor2D, presented in Figure 2, are rather tricky. These details are presented in the next two subsections, which can be skipped by the reader wishing to move on to the algorithm itself.

2.4 Time reduction due to Lemma 1

Let $n$ denote the length of RNA sequence $s$, and let $N$ denote the least even integer greater than or equal to $n$. Since $N$ is even, we have $(r+s) \equiv (r \cdot (N+1) + s) \mod 2$. For distinct fixed structures $A, B$, let $\pi_1(k) = \lfloor \frac{k}{N+1} \rfloor$, and $\pi_2(k) = k \mod (N+1)$, and define the polynomial

$$Z(x) = \sum_{r=0}^{N} \sum_{s=0}^{N} z_{r,N+s} x^r N+s = \sum_{k=0}^{(N+1)^2-1} z_{\pi_1(k) \cdot (N+1) + \pi_2(k)} x^\pi_1(k) (N+1) + \pi_2(k) = \sum_{k=0}^{(N+1)^2-1} z_k x^k$$

where for the last equality, we have used the fact that $k = \pi_1(k) \cdot (N+1) + \pi_2(k)$, well-known from row major order of a 2-dimensional array.
Consider the coefficients of the polynomial

\[ Z(x) = \sum_{r=0}^{N} \sum_{s=0}^{N} z_{r,s} x^{r+s} = \sum_{k=0}^{(N+1)^2-1} z_k x^k. \]  

(11)

Since \( N \) is even, the parity of \( r+s \) equals the parity of \( r(N+1)+s \), hence it follows from the parity condition that either (i) all coefficients \( z_1, z_3, z_5, \ldots \) of odd parity are zero, or (ii) all coefficients \( z_0, z_2, z_4, \ldots \) of even parity are zero. To simplify notation, in the remainder of this subsection, let \( M \) be the least integer greater than or equal to \( (N+1)^2 \) that is evenly divisible by 4, and let \( M_0 = M/2 \). We will assume that \( Z(x) = \sum_{k=0}^{M_0-1} z_k x^k \), whereupon coefficients \( z_k = 0 \) for \( k > (N+1)^2/2 \).

**Case 1:** All coefficients \( z_k \) of odd parity in equation (11) are zero.

In this case, we have \( Z(x) = \sum_{k=0}^{M_0-1} z_k x^{2k} \), where we have made a variable change \( u = x^2 \), and coefficient changes \( b_k = a_{2k} \).

By evaluating \( M_0 = M/2 \) many complex \( M_0 \)-roots of unity, we can use polynomial interpolation to determine all coefficients \( b_k \) of the polynomial

\[ Y(u) = \sum_{k=0}^{M_0-1} b_k u^k = \sum_{k=0}^{M_0-1} z_k x^{2k}. \]

Since \( Y(x^2) = Z(x) \), we have \( Y(\exp(2\pi k i/2M)) = Y(\exp(4\pi k i/2M)) = Z(\exp(2\pi k i/2M)) \), hence we use the previous recursions (6) to evaluate \( Z(\exp(2\pi k i/2M)) \).

Instead of performing \( M \) evaluations of \( Z(x) \) at \( M \)-roots of unity, this requires only \( M_0 = M/2 \) evaluations of \( Y(u) \) at \( M_0 \)-roots of unity; i.e. only half the number of evaluations of \( Z(x) \) are necessary to obtain the coefficients of \( Y(x) \). But then, we immediately obtain the full polynomial \( Z(x) \), since its coefficients of odd parity are zero.

**Case 2:** All coefficients \( z_k \) of even parity in equation (11) are zero.

In this case, \( z_0, z_2, z_4, \ldots \) are zero, so \( Z(x) = \sum_{k=0}^{M_0-1} z_{2k+1} x^{2k+1} \). But then \( Z(x) = x \cdot Y(u) \), where \( Y(u) = \sum_{k=0}^{M_0-1} b_k u^k \), where we have made a variable change \( u = x^2 \), and coefficient changes \( b_k = z_{2k+1} \). Similarly to Case 1, we can interpolate the \( M_0 \) coefficients of the polynomial \( Y(u) = \sum_{k=0}^{M_0-1} b_k u^k \) by evaluating \( M_0 \) many complex \( M_0 \)-roots of unity. Since \( Z(x) = x \cdot Y(x^2) \), \( Y(x^2) = x^{-1} \cdot Z(x) \), so \( Y(\exp(2\pi k i/2M)) = Y(\exp(4\pi k i/2M)) = \exp(-2\pi k i/2M) \cdot Z(\exp(2\pi k i/2M)) \), employing the previous recursions (6) to evaluate \( Z(\exp(2\pi k i/2M)) \). Note, that unlike the Case 1, since \( Z(x) = x \cdot Y(x^2) \), we have \( Y(x^2) = \frac{Z(x)}{x} \), which explains the presence of additional factor \( \exp(-2\pi k i/2M) \) in Case 2. Thus, instead of performing \( M \) evaluations of \( Z(x) \) at \( M \)-roots of unity, we perform only \( M_0 = M/2 \) evaluations of \( Y(u) \) at \( M_0 \)-roots of unity; i.e. only half the number of evaluations of \( Z(x) \) are necessary to obtain the coefficients of \( Y(x) \). But then, we immediately obtain the full polynomial \( Z(x) \), since \( Z(x) = x \cdot Y(x^2) \), and the coefficients of \( Z(x) \) of even parity are zero.

In the following, we will need the observation, that if the parity of base pair distance \( d_{BP}(A,B) \) between \( A,B \) is even, then

\[ Y(x^2) = Z(x) \]  

(12)
while if the parity is odd, then

\[ Y(x^2) = \frac{1}{x} \cdot Z(x). \quad (13) \]

2.5 Time reduction due to Lemma 2

As before, let \( M \) be the least number evenly divisible by 4, which is greater than or equal to \((N+1)^2\), let \( \nu = \exp(\frac{2\pi i}{M}) \) and \( \omega = \nu^2 = \exp(\frac{2\pi i}{2M}) = \exp(\frac{2\pi i}{M^2}) \).

Clearly, \( \nu \) is a principal complex \( M \)-root of unity, while \( \omega \) is a principal complex \( M^2 \)-root of unity. Evaluate \( Z(\alpha) \) for each \( M^2 \)-root of unity that belongs to the first quadrant, and apply Lemma 2 to infer the values of \( Z(\alpha) \) for each \( M^2 \)-root of unity that belongs to the fourth quadrant. More precisely, we compute

\[ Y(\omega^k) = \begin{cases} 
Z(\nu^k) & \text{if } d_{BP}(A, B) = 0 \mod 2 \\
\frac{1}{\nu^k} \cdot Z(\nu^k) & \text{if } d_{BP}(A, B) = 1 \mod 2
\end{cases} \]

\[ Y(\omega^k) = \begin{cases} 
Z(-\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 0 \mod 2 \\
\nu^{-k} \cdot Z(-\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 1 \mod 2
\end{cases} \]

\[ Y(\omega^k) = \begin{cases} 
Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 0 \mod 2 \\
\nu^{-k} \cdot Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 1 \mod 2
\end{cases} \]

\[ Y(\omega^k) = \begin{cases} 
Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 0 \mod 2 \\
\nu^{-k} \cdot -Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 1 \mod 2
\end{cases} \]

\[ Y(\omega^k) = \begin{cases} 
Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 0 \mod 2 \\
\nu^{-k} \cdot \nu \cdot Z(\nu^{\frac{M^2}{2}-k}) & \text{if } d_{BP}(A, B) = 1 \mod 2
\end{cases} \]

Line 1 follows by definition, since \( \omega = \nu^2 \); line 2 follows by equations (12) and (13); line 3 follows by Lemma 4; line 4 follows by Lemma 3. Thus if \( d_{BP}(A, B) \) is even, then

\[ y_k = Y(\omega^k) = \begin{cases} 
Z(\nu^k) & \text{for } k = 0, \ldots, \frac{M}{4} \\
Z(\nu^{\frac{M^2}{2}-k}) & \text{for } k = \frac{M}{4} + 1, \ldots, \frac{M}{2} - 1.
\end{cases} \quad (14) \]

while if \( d_{BP}(A, B) \) is odd, then

\[ y_k = Y(\omega^k) = \begin{cases} 
\nu^{-k} \cdot Z(\nu^k) & \text{for } k = 0, \ldots, \frac{M}{4} \\
-\nu^{-k} \cdot -Z(\nu^{\frac{M^2}{2}-k}) & \text{for } k = \frac{M}{4} + 1, \ldots, \frac{M}{2} - 1.
\end{cases} \quad (15) \]

It follows that values \( y_0, \ldots, y_{M/2-1} \) can be obtained by only \( \frac{M}{4} \) evaluations of \( Z(x) \).
Improved Algorithm for FFTbor2D

Input: RNA sequence $s = s_1, \ldots, s_n$, and distinct secondary structures $A, B$ of $s$, and integer $m$.

Output: Probabilities $p(x, y) = Z(x, y)^2/|Z(x, y)|^2$ to $m$ significant digits for $x, y = 0, \ldots, n - 1$. Let $N$ be the least even number greater than or equal to $n$, $M$ be the least number evenly divisible by 4, which is greater than or equal to $(N + 1)^2$, $M_0 = M/2$, $\nu = \exp(\frac{2\pi i}{M_0})$, $\omega = \nu^2 = \exp(\frac{2\pi i}{M_0})$. For $0 \leq k < M^2$, let $\pi_1(k) = \lfloor k/M \rfloor$, $\pi_2(k) = k - M \cdot \pi_1(k) = k \mod M$, and note that $k = \pi_1(k) \cdot M + \pi_2(k)$.

1. for $k = 0, \ldots, M^2$
2. compute the $M$ roots of unity $\nu^k, \nu^{-k}$
3. for $k = 0, \ldots, M^2 - 1$
4. if $d_{BP}(A, B)$ even
5. if $k \leq M$
6. $y_k = Y(\omega^k) = Z(\nu^k)$ by (14)
7. else // $\frac{M}{2} < k < \frac{M^2}{2}$
8. $y_k = Y(\omega^k) = Z(\nu^{M/2-k})$ by (14)
9. else // $d_{BP}(A, B)$ is odd
10. if $k \leq M$
11. $y_k = Y(\omega^k) = \nu^{-k} \cdot Z(\nu^k)$ by (15)
12. else // $\frac{M}{2} < k < \frac{M^2}{2}$
13. $y_k = Y(\omega^k) = -1 \cdot \nu^{-k} \cdot Z(\nu^{M/2-k})$ by (15)
14. //note that $Z = \sum_{r,n}Z_{r,n} = y_0 = Z(1)$
15. for $k = 0, \ldots, M^2 - 1$
16. $y_k = 10^{m} \cdot \frac{y_k}{Z}$ //normalize $y_k$
17. //compute coefficients of $\frac{Y(x)}{Z}$ using (16)
18. if $d_{BP}(A, B)$ even then
19. for $k = 0$ to $M - 1$
20. $r = \pi_1(k), s = \pi_2(k)$
21. if $k$ even
22. $Z_i^r = a_k/2$ from (16)
23. else // $k$ odd
24. $Z_i^r = 0$
25. else // $d_{BP}(A, B)$ odd
26. for $k = 0$ to $M - 1$
27. $r = \pi_1(k), s = \pi_2(k)$
28. if $k$ even
29. $Z_i^r = 0$
30. else // $k$ odd
31. $Z_i^r = a(k-1)/2$ from (16)
32. for $k = 0$ to $(N + 1)^2$
33. $z_k = \lfloor 10^m \cdot z_k \rfloor \cdot 10^{m}$
34. //truncate to $m$ significant digits

Fig. 2 Pseudocode to compute the $m$ most significant digits for probabilities $p_k = \frac{Y(x)}{Z} = Z_{r,s}^{\pi_1(k), \pi_2(k)}$. Our program, FFTbor2D, supports values of $m = 1, \ldots, 8$ for the precision parameter $m$. (Note that the software actually uses base 2 precision parameter, with maximum of 27, where $2^{27} \approx 10^8$.)

2.6 Using the fast Fourier transform

Now let $M_0 = M/2$, let $\nu = \exp(\frac{2\pi i}{M_0})$ be the principal $M$ root of unity, and $\omega = \nu^2 = \exp(\frac{2\pi i}{M_0})$ be the principal $M_0$ root of unity. Recall that the
Vandermonde matrix $V_{M_0}$ is defined to be the $M_0 \times M_0$ matrix, whose $i, j$ entry is $\omega^{i-j}$, i.e.

$$
V_{M_0} = \begin{pmatrix}
1 & 1 & 1 & \cdots & 1 \\
1 & \omega & \omega^2 & \cdots & \omega^{M_0-1} \\
1 & \omega^2 & \omega^4 & \cdots & \omega^{2(M_0-1)} \\
1 & \omega^3 & \omega^6 & \cdots & \omega^{3(M_0-1)} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \omega^{M_0-1} & \omega^{2(M_0-1)} & \cdots & \omega^{(M_0-1)(M_0-1)}
\end{pmatrix}
$$

The Fast Fourier Transform (FFT) is the $O(n \log n)$ algorithm, which computes the Discrete Fourier Transform (DFT), defined as the matrix product $Y = V_{M_0} A$:

$$
\begin{pmatrix}
y_0 \\
y_1 \\
y_2 \\
\vdots \\
y_{M_0-1}
\end{pmatrix}
= V_{M_0} \begin{pmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_{M_0-1}
\end{pmatrix}
$$

The $(i,j)$ entry of $V_{M_0}^{-1}$ is $\frac{\omega^{-ji}}{M_0}$ and that

$$
a_j = \frac{1}{M_0} \sum_{k=0}^{M_0-1} y_k \omega^{-kj} = \frac{1}{M_0} \sum_{k=0}^{M_0-1} y_k \nu^{-2kj}
$$

for $j = 0, \ldots, M_0 - 1$ (for more on FFT, see [3]).

Since we defined $Y$ in (8) by $Y = (y_0, \ldots, y_{M_0-1})^T$, where $y_0 = Z(\alpha_0), \ldots, y_{M_0-1} = Z(\alpha_{M_0-1})$ and $\alpha_k = \omega^k \exp(\frac{L_{\omega^2} Z_{\omega^2}(k)}{M_0})$, it follows that the coefficients $z_k = Z_{\omega^2}^{T(k), \pi_2(k)}$ in the polynomial $Z(x) = z_0 + z_1 x + \cdots + z_M x^M$ defined in (3) can be computed, at least in principle, by using the FFT. However, since the values of $z_k$ are astronomically large, numerical instability makes even this approach infeasible for moderate values of $n$. Nevertheless, we apply this approach to compute the $m$ most significant digits of $\frac{Z_{\omega^2}^{T(k), \pi_2(k)}}{Z_{\omega^2}}$, where the partition function $Z = \sum_S \exp(-E(S)/RT)$ satisfies $Z = \sum_{x,y} Z_{x,y}^{T(k), \pi_2(k)}$. This leads to numerical stability, allowing FFTbor2D to compute the $m$ most significant digits of $p(x,y) = \frac{Z_{x,y}^{T(k), \pi_2(k)}}{Z_{\omega^2}}$. Pseudocode for the complete algorithm, FFTbor, is given in Figure 2.

3 Benchmarking

To perform comparative benchmarking between RNA2Dfold and FFTbor2D, we took precision parameter $m = 8$, and proceeded as follows. For each sequence length $n = 20, 25, 30, \ldots, 300$, we generated 100 random sequences using probability 0.25 for each nucleotide A, C, G, U. For a given RNA sequence $s$, the metastable structure $A$ was taken to be the MFE structure of $s$. Using RNAbor, we determined that value $k_0 \geq 10$, for which partition function $Z_{k_0}$ constitutes a visible peak in the graphical output – see Figure 2 and 3 of [8] for an example. Subsequently, metastable
structure $B$ was taken to be that structure having minimum free energy over all structures, whose base pair distance from $A$ was $k_0$.

For all $0 \leq x, y \leq n$, RNA2Dfold and FFTbor2D were benchmarked in the computation of all Boltzmann probabilities $p(x, y) = 2^{(x+y)/2}$, where $x$ [resp. $y$] represents base pair distance to metastable structure $A$ [resp. $B$]. Care was taken for both software to employ the same energy model (Turner99 energy model, no dangling, suppression of minimum free energy structure computations for RNA2Dfold) and the same number of parallel threads (8 threads using OpenMP). Nonetheless, there are slight differences in the energy models – namely, RNA2Dfold includes mismatch penalties for multiloop stems and for exterior loops, while FFTbor2D does not. Even in the computation of the partition function $Z$, for spliced leader RNA from *L. collosoma* of length 56 nt, RNA2Dfold -d0 obtains a value of $-9.660419 \text{ kcal/mol}$, while FFTbor2D obtains $-9.660543 \text{ kcal/mol}$; similarly, for attenuator RNA of length 73 nt, RNA2Dfold -d0 obtains a value of $-22.171785 \text{ kcal/mol}$, while FFTbor2D obtains $-22.173213 \text{ kcal/mol}$. Note that the straightforward calculation of the partition function, following McCaskill’s algorithm [15] makes no use of the FFT engine, and thus the differences cannot be due to floating point or precision issues.

For benchmarking purposes, to allow for a fair comparison of FFTbor2D with RNA2Dfold, we restricted the range of $x, y$ in the same manner as done in the source code of RNA2Dfold. In that code, parameters $K$ [resp. $L$] are defined respectively to be the sum of the number of base pairs in reference structure $A$ [resp. reference structure $B$] plus the number of base pairs in the maximum matching (Nussinov) structure which contains no base pair of $A$ [resp. $B$]. For $x \geq K, y \geq L$, both RNA2Dfold and FFTbor2D set $p(x, y) = 0$. For the benchmarking results displayed in Figures 3, 4, 5, the values $x, y$ are restricted in FFTbor2D to $0 \leq x, y \leq \max(K, L)$, while $0 \leq x \leq K$ and $0 \leq y \leq L$ in RNA2Dfold.

Figure 3 depicts average run time of RNA2Dfold and FFTbor2D as a function of RNA sequence length, for random RNA sequences of lengths 20 – 200 and their metastable structures $A, B$, as previously explained. We see that both programs have roughly comparable run times for sequences of length up to approximately 80 nt, while FFTbor2D is demonstrably faster for longer sequences. Figure 4 presents log run time as a function of sequence length, in order to more clearly determine the crossover point in performance. RNA2Dfold is marginally faster for sequences of length up to roughly 80 nt, though the difference is in the millisecond range. Figure 5 shows that the standard deviation of run times on random sequences is tiny for FFTbor2D compared with RNA2Dfold, where standard deviation increases rapidly as a function of sequence length. This figure shows that run time of RNA2Dfold depends on sequence details, as well as sequence length, while the run time of FFTbor2D depends only on sequence length.

4 Kinetics

In this section, we describe folding kinetics along the 2D energy grid, as depicted in Figure 6. Consider the 56 nt *L. collosoma* spliced leader RNA [12], described in the Introduction, having sequence AACUAAAACA AUUUUUGAAG AACAGUUUCU GUACUUCAUU GGUAAUGUAGA GACUUC. Let $A$ denote the minimum free energy structure of spliced leader, using Turner 1999 energies as implemented in Vienna RNA Package 1.8.5:
Fig. 3 Run time in seconds for RNA2Dfold and FFTbor2D on random RNA sequences of length 20 – 200 nt, where sequence generation and choice of metastable structures A, B is described in the text. Beyond a length of approximately 80 nt, FFTbor2D is demonstrably faster.

\[
\text{\ldots ((...(((...(((((...)))))))))...))\ldots,}
\]
and let B denote the low energy alternate structure for spliced leader:

\[
\text{\ldots ((((((((....)))))))..))....}.
\]

Using the program Switch Design (switch.pl) described in [7], we generated 20,000 sequences, for which structures A, B are metastable. For spliced leader RNA as well as for each of these 20,000 sequences, we computed the corresponding probability profile \( p(x, y) \) using FFTbor2D, and subsequently defined the Markov chain \( M(s) = (Q, M) \), where \( Q = \{ (x, y) : 0 \leq x, y \leq n, \ p(x, y) > 0 \} \) is the set of states, and the transition probability matrix \( M = (M_{i,j}) \) is defined by

\[
M_{(x,y),(u,v)} = \begin{cases} \frac{1}{|Q|-1} \cdot \min(1, \frac{p(u,v)}{p(x,y)}) & \text{if } (u, v) \neq (x, y) \\ 1 - \sum_{(u,v) \neq (x,y)} M_{(x,y),(u,v)} & \text{if } (u, v) = (x, y). \end{cases}
\]

Let \( d_0 = d_{BP}(A, B) \) denote the base pair distance between the metastable structures A, B, and let \( M_{(d_0,0)} \) denote the matrix obtained from \( M \) by removing both the row and column corresponding to \((0, d_0)\). For spliced leader and each of the 20,000 sequences obtained from Switch Design, we determined the mean first passage
time (MFPT) from state \((0, d_0)\), corresponding to metastable structure \(A\), to state \((d_0, 0)\), corresponding to metastable structure \(B\), by computing \((I - M^{-1}_{(d_0, 0)}))^{-1} e\), where \(I\) denotes the identity matrix, and \(e\) denotes the column vector composed entirely of ones [16]. Using LAPACK [1] for matrix inversion, we found that *L. collosoma* spliced leader RNA has a MFPT on the 2D energy grid, which is smaller than only 2.855% of the 20,000 sequences generated by Switch Design, thus constituting a Z-score of 1.989 for the kinetics of folding from \(A\) to \(B\) (see left panel of Figure 7). This result seems to suggest that spliced leader could be under evolutionary pressure for slow folding between these metastable structures, if we take MFPT from \((0, d_0)\) to \((d_0, 0)\) as a surrogate for *Kinfold* [6] folding time from \(A\) to \(B\) – an interpretation which seems to be consistent with the functional role of spliced leader as described in the Introduction. Since accurate *Kinfold* kinetics requires many simulations, each requiring enormous time [24], our method may prove useful in synthetic biology, in prioritizing computationally designed RNA sequences for subsequent experimental validation.

Finally, it should be mentioned that the GC-content of spliced leader RNA is 30.357%, which constitutes a Z-score of +2.50; i.e. the overwhelming majority of

---

**Fig. 4** Logarithm of run time in seconds for *RNA2Dfold* and *FFTbor2D* on random RNA sequences of length less than 200 nt, for same data as that in Figure 3. By taking logarithm of run times, the crossover points are apparent, where *FFTbor2D* is faster than *RNA2Dfold*. For very small sequences, *RNA2Dfold* is faster, though since both programs converge in a fraction of a second, this difference is of no practical consequence.
Fig. 5 (Left) Standard deviation of run times of RNA2Dfold and FFTbor2D as a function of sequence length \( n \). (Right) Minimum and maximum run times for RNA2Dfold and FFTbor2D. For each collection of 100 random sequences of length \( n \), the minimum and maximum run time for a sequence of that length was computed. Taken together, these figures clearly show the run time dependence of RNA2Dfold on particular sequences, while the run time of FFTbor2D depends only on sequence length, rather than sequence details.

Fig. 6 2D projection of energy landscape for Spliced Leader (SL) RNA from *Leptomonas collosoma*, having sequence AACUUAAAACAUUUUGAAGAACAGUUUCGUACUUCAUUGGUAAUGUAGAGACUUC and metastable secondary structures \( A = \ldots ((((((.....)))))))... \ldots \ldots \ldots 

The \( x \)-axis (resp. \( y \)-axis) represents base pair distance between metastable structure \( A \) (resp. \( B \)), while the \( z \)-axis represents the ensemble free energy \(-RT \log \sum_{x,y} Z_{x,y}\), where \( Z_{x,y} \) is computed in FFTbor2D by \( Z_{x,y} = p(x,y) \cdot Z \). Low energy positions \((x,y)\) correspond to high Boltzmann probability positions. The left panel depicts a heat map of the ensemble free energy, while the right panel depicts a contour map with level curves. In analogy with mountain climbing, one expects an optimal path to follow along the valley regions in traversing the landscape from \( A \) to \( B \). Data produced with FFTbor2D; graphics produced using Mathematica.

the 20,000 sequences generated by *Switch Design* have higher GC-content than that of spliced leader RNA from *L. collosoma*, as shown in the right panel of Figure 7.
5 Discussion

Given an RNA sequence \( s \) and two reference secondary structures \( A, B \), the algorithm, \textit{FFTbor2D}, computes the partition function \( Z(x, y) \), defined to be the sum of Boltzmann factors \( \exp(-E(S)/RT) \) over all secondary structures \( S \), having base pair distance \( x \) to \( A \) and distance \( y \) to \( B \), where \( 0 \leq x, y \leq n \) and \( n \) denotes the length \( s \). Using polynomial interpolation with the FFT and exploiting the observations of Lemmas 1, 2, \textit{FFTbor2D} has worst case complexity \( O(n^5) \) time and \( O(n^2) \) space. This worst case algorithmic complexity is two orders of magnitude faster and requires two orders of magnitude less space than the worst case complexity of the algorithm \textit{RNA2Dfold} of Lorenz et al. [14]. This run time complexity bound is not only theoretical, but entails a significant practical speedup, as depicted in Figures 3, 4 and 5.

An important advantage of \textit{RNA2Dfold} over \textit{FFTbor2D} is that the former can additionally compute the structures \( M_{x,y} \) having minimum free energy over all structures that are \( x \)-neighbors of metastable \( A \) and simultaneously \( y \)-neighbors of metastable \( B \). (There is a similar advantage of \textit{RNAbor} [8] over the faster \textit{FFTbor} [21] .) As well, \textit{RNA2Dfold} directly computes the partition function values \( Z_{x,y} \), while \textit{FFTbor2D} estimates \( Z_{x,y} \) by computing \( p(x, y) \cdot Z \). This difference entails a significant loss of precision, when depicting the energy landscape.

The right panel of Figure 6 depicts a contour heat map of the 2D energy landscape for spliced leader RNA from \textit{L. collosoma}, as computed by \textit{FFTbor2D}. This figure should be compared with the left panel of Figure 8, which depicts a
Fig. 8 2D projection of energy landscape for Spliced Leader (SL) RNA from *Leptomonas collosoma*, as in Figure 6, except that in the left panel, ensemble free energy $-RT \log Z_{x,y}$ is computed from the values of $Z_{x,y}$ output by RNA2Dfold, while in the right panel, ensemble free energy is computed from the values $Z_{x,y} = p(x,y) \cdot Z$, where values $p(x,y)$ are output by RNA2Dfold. The loss of detail in the 2D energy landscape is caused uniquely by working with probabilities $p(x,y)$, rather than partition function values $Z_{x,y}$. Data produced with RNA2Dfold; graphics produced using Mathematica.

contour heat map of the 2D energy landscape for the same RNA, as computed by RNA2Dfold. Notice the additional detail in this figure, due to the fact that RNA2Dfold directly computes $Z_{x,y}$, while FFTbor2D computes Boltzmann probabilities $p(x,y)$ by interpolation, allowing one to subsequently compute $Z_{x,y} = p(x,y) \cdot Z$. The additional detail of the energy landscape is lost in the right panel of Figure 8, obtained by computing ensemble free energy by $-RT \log p(x,y) + RT \log Z$, where $p(x,y)$ is parsed from RNA2Dfold output. It follows that the loss of detail in 2D energy landscape is due solely to the fact that probabilities $p(x,y)$ are computed by FFTbor2D, rather than partition function values $Z_{x,y}$. Given numerical stability issues involving the FFT engine, FFTbor2D can only estimate the probabilities $p(x,y)$ to within $m = 8$ significant places. Nevertheless, our algorithm FFTbor2D was developed with the intended application in synthetic biology, where one wishes to prioritize RNA candidate sequences with respect to kinetics. For such applications, the speedup of FFTbor2D is an important asset.

Acknowledgments

We would like to thank anonymous reviewers for their critique and suggestions. FFTbor2D uses the implementation FFTW of the Fast Fourier Transform, developed by Frigo and Johnson [9] at http://www.fftw.org/. The research was funded by National Science Foundation grants DMS-1016618 and DBI-1262439. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

2D energy landscapes of RNA


Appendix

Here, we provide proofs of Theorem 1 and of Lemmas 2, 3, 4.

**Theorem 1:** Let \( s_1, \ldots, s_n \) be a given RNA sequence. For any integers \( 1 \leq i \leq j \leq n \), let \( Z_{i,j}(x) = \sum_{r=0}^{n-1} \sum_{s=i}^{j} x^{r,n+s} \) where \( x^{r,n+s} = Z_{i,j}(x) = \sum_{r=0}^{n-1} \sum_{s=i}^{j} x^{r,n+s} \). Then for \( i \leq j \leq i + \theta \), \( Z_{i,j}(x) = 1 \) and for \( j > i + \theta \) we have the recurrence relation

\[
Z_{i,j}(x) = Z_{i,j-1}(x) + \sum_{r=0}^{n-1} \sum_{s=j+1}^{n} x^{r,n+s} \cdot Z_{i,k-1}(x) \cdot Z_{k,n+1,j-1}(x) \cdot x^{n(k)n+\beta(k)}
\]

(17)

where \( \alpha_0 = 1 \) if \( j \) is base-paired in \( A_{i,j} \) and 0 otherwise, \( \beta_0 = 1 \) if \( j \) is base-paired in \( B_{i,j} \) and 0 otherwise, and \( \alpha(k) = d_{BP}(A_{i,j}) \cup A_{i,j} \cup \{ (k,k) \} \), \( \beta(k) = d_{BP}(B_{i,j}) \cup B_{i,j} \cup \{ (k,k) \} \).

**Proof:** First, some notation is necessary. Recall that if \( F \) is an arbitrary polynomial [resp. analytic] function, then \( [x^{r,n+s}]F(x) \) denotes the coefficient of monomial \( x^{r,n+s} \) in the Taylor expansion of \( F(x) \) – for instance, in equation (3) of the main text, \( [x^{r,n+s}]Z(x) = z_{r,n,s} \).

By definition, it is clear that \( Z_{i,j}(x) = 1 \) if \( i \leq j \leq i + \theta \), where we recall that \( \theta = 3 \) is the minimum number of unpaired bases in a hairpin loop. For \( j > i + \theta \), we have

\[
[x^{r,n+s}]Z_{i,j}(x) = z_{r+n,s}(i,j) = Z_{n+s}
\]

\[
= Z_{i,j-1}(x) + \sum_{k=0}^{n-1} \sum_{s=0}^{n} e^{-E_{k}(i,j)} \cdot Z_{k,n+1,j-1}(x) \cdot x^{n(k)n+\beta(k)}
\]

\[
= \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{i,j-1}(x) + \sum_{k=0}^{n-1} \sum_{s=0}^{n} e^{-E_{k}(i,j)} \cdot \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{k,n+1,j-1}(x)
\]

\[
= \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{i,j-1}(x) + \sum_{k=0}^{n-1} \sum_{s=0}^{n} e^{-E_{k}(i,j)} \cdot \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{k,n+1,j-1}(x)
\]

\[
= \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{i,j-1}(x) + \sum_{k=0}^{n-1} \sum_{s=0}^{n} e^{-E_{k}(i,j)} \cdot \left[ x^{(r-\alpha)+s(\beta)} \right] Z_{k,n+1,j-1}(x)
\]

By induction, the proof of the theorem now follows. \( \square \)

The following lemma was proved in [21], and is reproduced for the convenience of the reader.

**Lemma 2:** If \( Z(x) \) is the complex polynomial defined in equation (9) of the main text, then for any complex nth root of unity \( \alpha \), it is the case that \( Z(\pi) = \overline{Z(\alpha)} \). In other words, if \( \alpha \) is a complex nth root of unity of the form \( a+bi \), where \( a, b \in \mathbb{R} \) and \( b > 0 \), and if \( Z(a+bi) = A+Bi \) where \( A, B \in \mathbb{R} \), then it is the case that

\[
Z(a-bi) = A-Bi.
\]

**Proof:** Letting \( i = \sqrt{-1} \), if \( \theta = \frac{2\pi}{n} \), then \( \omega = e^{i\theta} = \cos(\theta) + i\sin(\theta) \) is the principal nth complex root of unity, and \( 1, \omega, \ldots, e^{(n-1)i\theta} = \omega^{n-1} \) together constitute the complete
collection of all nth complex roots of unity – i.e., the n unique solutions of of the equation $x^n - 1 = 0$ over the field $\mathbb{C}$ of complex numbers. Clearly, for any $1 \leq r < n$, $e^{-ir\theta} = e^{2\pi ir/n}$, $e^{i\theta} = e^{(2\pi - r)\theta} = e^{i(n-r)\theta}$, and $e^{i\theta} = e^{i(n-r)\theta} = e^{i(n-r)\theta}$. Moreover, if $\omega = e^{i\theta} = a + bi$ where $b > 0$, then we have $e^{-ir\theta} = a - bi$. It follows that for any nth root of unity of the form $a + bi$, where $b > 0$, the number $a - bi$ is also an nth root of unity.

Recall that $Z(x) = \sum_{k=0}^{n} c_k x^k$, where $c_k \in \mathbb{R}$ are real numbers representing the partition function $Z_{s_1, s_2, \ldots, s_n}$ over all secondary structures of a given RNA sequence $s_1, \ldots, s_n$, whose base pair distance from initial structure $S^*$ is $k$. Thus, in order to prove the lemma, it suffices to show that for all values $k = 0, \ldots, n - 1$, if $a + bi$ is a complex nth root of unity, where $a, b \in \mathbb{R}$ and $b > 0$, and if $(a + bi)^k = C + D\nu$ where $C, D \in \mathbb{R}$, then $(a - bi)^k = C - D\nu$. Indeed, we have the following.

$$(a + bi)^m = \sum_{k=0}^{m} \left( \begin{array}{c} m \\ k \end{array} \right) a^{m-k} \cdot (bi)^k$$

$$(bi)^k = \begin{cases} b^k & \text{if } k \equiv 0 \mod 4 \\ ib^k & \text{if } k \equiv 1 \mod 4 \\ -b^k & \text{if } k \equiv 2 \mod 4 \\ -ib^k & \text{if } k \equiv 3 \mod 4 \end{cases}$$

$$(a - bi)^m = \sum_{k=0}^{m} \left( \begin{array}{c} m \\ k \end{array} \right) a^{m-k} \cdot (-bi)^k$$

$$(-bi)^k = \begin{cases} b^k & \text{if } k \equiv 0 \mod 4 \\ -ib^k & \text{if } k \equiv 1 \mod 4 \\ -b^k & \text{if } k \equiv 2 \mod 4 \\ ib^k & \text{if } k \equiv 3 \mod 4 \end{cases}$$

It follows that each term of the form $a^{m-k} \cdot (bi)^k$, for $k = 0, \ldots, m$, is the complex conjugate of $a^{m-k} \cdot (-bi)^k$, and thus $(a + bi)^m$ is the complex conjugate of $(a - bi)^m$. Since $Z(a + bi)$ is a sum of terms of the form $c_k (a + bi)^k$, it follows that $Z(a - bi)$ is the complex conjugate of $Z(a + bi)$. This completes the proof of the lemma.

**Lemma 3:** Let $d_0 = d_{HF}(A, B)$. Then for any complex number $\alpha \in \mathbb{C}$, $Z(-\alpha) = -1^{d_0} \cdot Z(\alpha)$.

**Proof:** The lemma states that if the base pair distance between reference structures $A, B$ is even, then $Z(-\alpha) = Z(\alpha)$, while if the distance is odd, then $Z(-\alpha) = -Z(\alpha)$. Suppose first that $d_0$ is even. By Lemma 1, $Z(x) = Z(x) = z_0 + z_1 x + z_2 x^2 + \cdots + z_{M-1} x^{M-2}$, and so $Z(-\alpha) = Z(\alpha)$. Suppose now that $d_0$ is odd. By Lemma 1, $Z(x) = z_1 x^1 + z_2 x^3 + z_3 x^5 + \cdots + z_{M-1} x^{M-1}$, and so $Z(-\alpha) = -Z(\alpha)$.

**Lemma 4:** Suppose that $\nu = \exp(\frac{2\pi i}{M})$ is the principal $M$-root of unity, and that $\frac{M}{4} < k \leq \frac{M}{2}$. Then

$$\nu^k = -\left(\nu \cdot (M/2 - k)\right) = -\nu^{M/2 - k}.$$  

**Proof:** Recall Euler’s formula in complex analysis: $\exp(iz) = \cos(z) + i \sin(z)$. As well, recall that $\sin(\pi) = 0$, $\cos(\pi) = -1$, and the trigonometric addition formulas:

$$\cos(a - b) = \cos(a) \cos(b) + \sin(a) \sin(b)$$

$$\sin(a - b) = \sin(a) \cos(b) - \sin(b) \cos(a).$$
Then
\[
\nu^{M/2 - k} = \exp \left( \frac{2\pi i (M/2 - k)}{M} \right)
\]
\[
= \cos \left( \frac{2\pi (M/2 - k)}{M} \right) + i \sin \left( \frac{2\pi (M/2 - k)}{M} \right)
\]
\[
= \cos \left( \pi - \frac{2\pi k}{M} \right) + i \sin \left( \pi - \frac{2\pi k}{M} \right)
\]
\[
= \left[ \cos(\pi) \cos \left( \frac{2\pi k}{M} \right) + \sin(\pi) \sin \left( \frac{2\pi k}{M} \right) \right] + \left[ \sin(\pi) \cos \left( \frac{2\pi k}{M} \right) - \sin \left( \frac{2\pi k}{M} \right) \cos(\pi) \right]
\]
\[
= - \cos \left( \frac{2\pi k}{M} \right) + i \sin \left( \frac{2\pi k}{M} \right)
\]
\[
= -1 \left[ \cos \left( \frac{2\pi k}{M} \right) - i \sin \left( \frac{2\pi k}{M} \right) \right]
\]
\[
= -1 \cdot \cos \left( \frac{2\pi k}{M} \right) + i \sin \left( \frac{2\pi k}{M} \right)
\]
\[
= -1 \cdot \exp \left( \frac{2\pi ik}{M} \right) = -\nu^k.
\]

It follows that \( \nu^{M/2 - k} = -\nu^k \), so \( \nu^k = -\nu^{(M/2 - k)} = -\nu^{(M/2 - k)} \). This completes the proof of the lemma.