Algorithms for Mapping Nucleic Acid Free Energy Landscapes

Thesis by

Jonathan A. Othmer

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

2009
(Defended June 9, 2008)

Acknowledgements

First I wish to acknowledge my advisor, Niles Pierce, for his support, direction, and the opportunity to work in a stimulating multidisciplinary environment. Thanks also to my committee, Houman Owhadi, Jerry Marsden, Erik Winfree, and Dan Meiron, for their advice, input, and topic suggestions throughout my tenure here. I would not be here if it were not for the attention, support and upbringing of my parents. They—and maybe my non-traditional elementary school—are largely responsible for my academic success. Finally, I wish to thank my fiancée for her companionship, her support, and for making it all worthwhile.

Abstract

To complement the utility of thermodynamic calculations in the design and analysis of nucleic acid secondary structures, we seek to develop efficient and scalable algorithms for the analysis of secondary structure kinetics. Secondary structure kinetics are modeled by a first-order master equation, but the number of secondary structures for a sequence grows exponentially with the length of the sequence, meaning that for systems of interest, we cannot write down the rate matrix, much less solve the master equation. To address these difficulties, we develop a method to construct macrostate maps of nucleic acid free energy landscapes based on simulating the continuous-time Markov chain associated with the microstate master equation. The method relies on the careful combination of several elements: a novel procedure to explicitly identify transitions between macrostates in the simulation, a goodness-of-clustering test specific to secondary structures, an algorithm to find the centroid secondary structure for each macrostate, a method to compute macrostate partition functions from short simulations, and a framework for computing transition rates with confidence intervals. We use this method to study several experimental systems from our laboratory with system sizes in the hundreds of nucleotides, and develop a model problem, the *d*-cube, for which we can control all of the relevant parameters and analyze our method's error behavior. Our results and analysis suggest that this method will be useful not only in the analysis and design of nucleic acid mechanical devices, but also in wider applications of molecular simulation and simulation-based model reduction.

Contents

Acknowledgements **iii**

Abstract **v**

List of Figures **ix**

List of Algorithms **xii**

1 Introduction **1**

1.1 Nucleic acid secondary structures 2

1.2 Thermodynamics of nucleic acid secondary structures 4

1.3 Nucleic acid secondary structure kinetics 5

1.4 Goals and outline 6

2 An Enumerative Approach to Constructing Macrostates Rate Matrices **9**

2.1 Barrier trees and basin graphs: Partitioning the secondary structure landscape 9

2.2 Phenomenological rate constants for a particle in a two-basin landscape . . 11

2.2.1 Distinguishing rate constants 12

2.3 Dominant local relaxation 13

2.3.1 Examples 15

2.3.2 Saddle assignments 15

2.4 Summary and outlook 19

3 Simulation-Based Coarse-Graining of Nucleic Acid Free Energy Landscapes **21**

3.1 Method 21

3.1.1 Locating transitions 22

3.1.2 Clustering macrostates and finding macrostate centroids 24

3.1.3 Computing partition functions 26

3.1.4 Computing transition rates 27

3.1.5 Computing initial conditions 31

3.1.6 Algorithmic implementation and complexity 31

3.2 Examples 33

3.2.1 RNA hairpin 33

3.2.2 Hybridization chain reaction 37

3.2.3 Three-arm junction 39

4 Model problem 43

4.1 Computational and analytic results 45

4.1.1 Transition identification 45

4.1.1.1 Type I errors 45

4.1.1.2 Type II errors 49

4.1.2 Clusters and partition functions 50

4.1.3 Rate constants 52

4.2 Hierarchical macrostates 53

4.3 Outlook and extensions 53

5 Centroids in Ensembles of Nucleic Acid Structures 55

5.1 Comparing representative structures 56

5.1.1 How the MFE and centroid characterize the ensemble 57

5.1.2 Robustness to uncertainty in the energy parameters 59

5.1.3 Comparing two centroids 62

5.2 Algorithms 64

5.2.1 A single strand 64

5.2.2 A complex of interacting strands 68

5.2.2.1 A box containing a finite number of strands 70

5.2.2.2 Distinguishability issues 70

5.3 Summary 74

6 Conclusions and Outlook 75

Bibliography **76**

List of Figures

1.1 A non-pseudoknotted secondary structure 2

2.1 Example barrier tree and basin graph 10

2.2 A discrete two-basin potential 11

2.3 Basin graph and master equation solution for a 21 nucleotide RNA 16

2.4 Master equation solution for a 39 nucleotide RNA 17

2.5 Comparison of saddle assignment methods for a 27 nucleotide RNA 18

3.1 Computation of d_V for a two-well free energy landscape 23

3.2 Hierarchical clustering of two-dimensional data 25

3.3 Trajectory-based method applied to a 23 nucleotide RNA 35

3.4 Speedup under parallelization 36

3.5 Speedup under revised parallelization scheme 37

3.6 The trajectory-based method applied to HCR and a variant 38

3.7 Two steps of the HCR system 40

3.8 Catalytic three-arm junction 41

3.9 Alternative design of a catalytic three-arm junction 42

4.1 Numerical success probabilities for model problem 45

4.2 Type I error estimates for model problem 49

4.3 Type II error estimates for the model problem 51

4.4 Estimated partition functions for the model problem 51

4.5 Hierarchical macrostates for the model problem 54

5.1 Example with different MFE and centroid structures 58

5.2 Comparison between the MFE and centroid structures for designed sequences 60

5.3 Comparison between the MFE and centroid structures for random sequences 61

5.4 Well-clusteredness of MFE and centroid structures 62

5.5 Comparison between the centroid and the Ding centroid for designed sequences 65

5.6 Comparison between the centroid and the Ding centroid for random sequences 66

List of Algorithms

3.1 Pseudocode for the trajectory-based method 32

5.1 Pseudocode to find the minimum free energy (MFE) structure 69

5.2 Pseudocode to find the centroid structure, argmin $n(s)$ 69

5.3 Pseudocode to find the centroid structure, argmin $n(s)$, that operates in time $O(N^3)$ 71

5.4 Pseudocode to find the centroid structure of a multistranded complex in time $O(N^4)$ 71

5.5 Pseudocode to find the centroid structure of a multistranded complex in time $O(N^3)$ 72

Chapter 1

Introduction

Nucleic acids play varied roles in the cell, DNA as a storage medium and RNA as a messenger and regulatory element [3]. Recently, nucleic acids have been used as a versatile nanotechnological building material. This is due in part to the relative simplicity of the material (compared to, for example, proteins): The specificity of Watson-Crick base pairing (A pairs to T and G to C) and the fact that understanding structural features at the level of secondary structures is sufficient for nanotechnological applications, makes nucleic acids relatively easy to analyze and design. A wide range of work has been done designing both structures and dynamic devices [45, 46]. Recently, researchers have built devices whose autonomous function obeys prescribed dynamics [57].

The thermodynamic properties of nucleic acid secondary structures are well studied and can be computed efficiently by dynamic programming algorithms [37]. The inverse problem of design, choosing a sequence that adopts a particular structure with high probability, though probably not solvable in polynomial time, is also well understood [20]. Secondary structure kinetics are less well studied. The difference may partially be a result of the fact that there is an experimentally parameterized model for folding energies [35, 43], but relatively little known about kinetics at the secondary structure level. Authors have studied kinetics via the master equation formalism [58] and Monte Carlo simulations [25], but neither of these approaches alone is sufficient for large problems—the master equation because of the exponentially large number of secondary structures that must be enumerated and the Monte Carlo simulations because of the difficulty in interpreting the simulated trajectories. This thesis fills this gap by developing a simulation-based method to characterize the folding kinetics of nucleic acids that scales to systems of experimental interest.

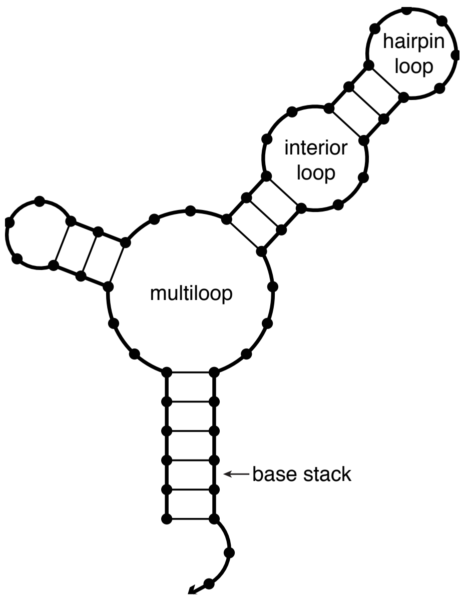


Figure 1.1: A non-pseudoknotted secondary structure. To compute the energy of a structure, it is decomposed into hairpin loops, interior loops, multi-loops, and base stacks; the energies of each component are then added together.

1.1 Nucleic acid secondary structures

The primary structure of a nucleic acid (NA) is the sequence of bases, taken from $\{A, C, G, U\}$ for RNA or $\{A, C, G, T\}$ for DNA, that comprise the strand. The strand may fold and base pair with itself. The pairs form only between the bases $\{A{\cdot}U,\ C{\cdot}G,\ A{\cdot}T,\ G{\cdot}T,\ G{\cdot}U\}$. Nucleic acid *secondary structures* ignore the full three-dimensional conformation by considering only which bases are paired with each other. No base is allowed to pair with more than one other base, and all base pairs must be nested. That is, for a strand with bases labeled $1, \ldots, N$, if base m is paired to n and r to s then either $m < r < s < n$ or $r < m < n < s$. This prohibits *pseudoknots*.[1] Figure 1.1 shows a non-pseudoknotted secondary structure with the different types of loops in the energy model labeled. The secondary structures for a particular sequence form a discrete space, Ω, and the energy of each structure is computed via a loop-decomposition model that has been experimentally parameterized [35, 43].

[1] The prohibition of pseudoknots is an algorithmic, not a physical, constraint. Pseudoknots are, in fact, integral to the formation of many biologically and nanotechnologically important structures [49, 55]; however, until recently they have been excluded from most NA folding algorithms because pseudoknotted structures cannot easily be included in the dynamic programming framework that underlies most algorithms. Indeed, pseudoknot minimum free energy (MFE) determination is NP-hard [2, 33]. Recent work has included restricted classes of pseudoknots in MFE determination, partition function, and kinetic simulation algorithms [21, 28, 41].

Though the space is discrete, the number of secondary structures for a sequence has been empirically found to scale exponentially with its length, N [60];

$$|\Omega| \approx 1.8^N. \tag{1.1}$$

Given a particular NA sequence we are typically not interested in a single secondary structures but rather the *ensemble* of structures. For this work we consider an ensemble to be a probability space, that is, the triple $(\Omega, \mathcal{F}, p)$ of states, a σ-algebra of measurable sets on Ω, and a probability measure on $\mathcal{F}$. The σ-algebra on $\mathcal{F}$ will always be all subsets of Ω, so we will leave it out of the notation. If a measure is unspecified it is assumed to be the Boltzmann equilibrium distribution

$$\pi(s) = \frac{e^{-\Delta G(s)/k_B T}}{Q}, \tag{1.2}$$

where $\Delta G(s)$ is the free energy of structure s, k_B is Boltzmann's constant, and T the temperature, though we will sometimes explicitly refer to the "equilibrium ensemble."[2] The normalizing constant, Q, is the partition function

$$Q = \sum_{s \in \Omega} e^{-\Delta G(s)/k_B T}. \tag{1.3}$$

A useful construct for manipulating ensembles of secondary structures is the pair probability matrix. For a sequence of length N, the pair probability matrix is an $N \times (N+1)$ matrix whose $(i,j)^{th}$ entry is the probability in (Ω, p) that bases i and j are paired; the $(i,(N+1))^{st}$ entry is the probability that base i is unpaired. More formally, we define a map, ρ, from the space of probability measures on Ω to the space of $N \times (N+1)$ matrices with entries in $[0,1]$ by

$$\rho : \mathcal{M}(\Omega) \longrightarrow [0,1]^{N\times(N+1)}, \quad P_{i,j} = \rho_{i,j}(p) = \sum_{s\in\Omega} p(s)S(s)_{i,j}, \tag{1.4}$$

where $S(s)_{i,j} = 1$ if i and j are paired in s and $S(s)_{i,N+1} = 1$ if i is unpaired. The norm

[2] There are two common definitions of ensemble in physics [15]. One is a set of constraints on the system, for example, constant temperature, volume, and particle number in the Canonical ensemble. The second is simply the set of configurations of the system. The probability distribution on those configurations is left unspecified, but for our purposes it is important to keep track of whether the system is at equilibrium, or if we are considering a non-equilibrium probability measure, such as one resulting from a kinetic simulation.

that we use to compare pair probability distributions is the ℓ^1 norm, which is defined by

$$\|P - R\|_1 = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N+1} |P_{i,j} - R_{i,j}|. \tag{1.5}$$

When comparing individual secondary structures, a natural metric is the nucleotide distance, which counts the number of nucleotides paired differently in the two structures. This is equivalent to (1.5) with both matrices representing ensembles of just one structure, and when comparing two single structures, we may denote this by $d(s^1, s^2)$. An alternative metric is the base-pair distance, which is the cardinality of the symmetric difference of the base pairs in the two structures. This can also be computed by leaving out the $(N+1)^{st}$ column from the sum in (1.5), that is,

$$\|P - R\|_{BP} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} |P_{i,j} - R_{i,j}|. \tag{1.6}$$

1.2 Thermodynamics of nucleic acid secondary structures

The loop-based energy model by which secondary structures are evaluated lends itself to the efficient calculation of thermodynamic quantities of the equilibrium ensemble. This is because the energy of a secondary structure is merely the sum of the energies of its constituent loops. Further, the nesting property for non-pseudoknotted structures means that when a pair is formed between indices d and e, the substructures on $(i, d-1)$, $(d+1, e-1)$, and $(e+1, N)$ are independent. Thus, a dynamic programming approach can be used to calculate on short subsequences and build up to longer subsequences. In this way, quantities that at first glance are sums over an exponentially large number of structures can be calculated in time that is polynomial in the sequence length.

The application of dynamic programming to the thermodynamics of nucleic acid secondary structures has a long history, beginning in the 1970s with the work of Waterman and Smith [52] and Nussinov et al. [40], and followed by Zuker and Stiegler [61]. In 1990 McCaskill [37] introduced an algorithm to compute the partition function and base pair probabilities for the single-stranded equilibrium ensemble of non-pseudoknotted structures, and this algorithm is the basis of subsequent work in this area. Lyngsø et al. [34] suggested improvements that reduce the computational complexity from $O(N^4)$ to $O(N^3)$ for a sequence of length N without approximation. Rivas and Eddy [41] describe an algorithm that extends the prediction of minimum free energy structures to a class of pseudoknots. Dirks and Pierce [21, 22] extended partition function and pair probability calculations to a smaller class of pseudoknots. Dirks et al. [19] extended the partition function calculations to complexes of interacting strands, and computed the equilibrium concentrations for a dilute solution of interacting strands. Other researchers have sought to extend dynamic programming approaches to calculating additional properties of ensembles. For example, Miklós et al. [38] calculated the mean and variance of the free energy over the ensemble. Ding and Lawrence [18] developed a method to generate samples from the equilibrium ensemble using the recursions that make up the partition function calculation. Though invaluable for analysis and design of nucleic acids, thermodynamic calculations do not give insight into the folding (or mis-folding) of a strand.

1.3 Nucleic acid secondary structure kinetics

Secondary structure kinetics are modeled by a first-order master equation that follows the time-varying probabilities of all states in the system [25, 58]. For each state in the system

$$\frac{d}{dt}p_i(t) = \sum_{j=1}^{|\Omega|} \left[k_{ij}p_j(t) - k_{ji}p_i(t)\right].$$

Aggregating the transition rates, k_{ij}, into the rate matrix, K, the equation becomes

$$\frac{d}{dt}p = Kp, \tag{1.7}$$

and with initial condition p_0, has solution

$$p(t) = e^{Kt}p_0. \tag{1.8}$$

For secondary structures, a pair of states is considered to be connected if one can be reached from the other by an elementary move, which we define as the addition or removal of a single base pair.[3] Since each state is connected to the open conformation (with no base pairs) by

[3] Other moves between secondary structures, such as shifting one end of a base pair, are possible, but they are not implemented in the simulation software we use [44], so we will not include them.

a sequence of elementary moves, the system as a whole is irreducible.

Though the free energies of secondary structures are well established, transition rates are not. For this study we use the Kawasaki [30] rule to construct approximate rates, as has been done in previous work [25]. With this rule, the rate matrix, has entries,

$$\begin{aligned} k_{ij} &= \begin{cases} e^{-(\Delta G_i - \Delta G_j)/2k_BT} & i \text{ is connected to } j \\ 0 & \text{otherwise} \end{cases} \\ k_{ii} &= -\sum_{j \neq i} k_{ji}. \end{aligned} \tag{1.9}$$

These rates obey detailed balance, that is, they are reversible with respect to π, so $k_{ij}\pi(j) = \pi(i)k_{ji}$. This fact and irreducibility are sufficient to ensure that $p(t) \to \pi$ as $t \to \infty$. The rate matrix is not symmetric, but reversibility implies that it is self-adjoint with the inner product

$$\langle x, y \rangle_\pi = x^T \text{diag}(\pi)^{-1} y, \quad x, y \in \mathbb{R}^{|\Omega|}.$$

Thus, it has real eigenvalues and a complete set of eigenvectors. The eigenvalues can be ordered $0 = \lambda_0 > -\lambda_1 \geq \cdots \geq -\lambda_{|\Omega|-1}$, and the eigenvector corresponding to λ_0 is π. Then (1.8) can be rewritten

$$p(t) = \pi + c_1 v_1 e^{-\lambda_1 t} + \cdots + c_{|\Omega|-1} v_{|\Omega|-1} e^{-\lambda_{|\Omega|-1} t}, \tag{1.10}$$

where v_i are right eigenvectors and c_i are constants depending on the initial conditions. (See Brémaud [7] or van Kampen [50] for additional background.) The master equation can be solved directly via a numerical method. Alternatively, one can construct stochastic trajectories through state space by simulating the continuous time Markov chain generated by K [25].

1.4 Goals and outline

Our goal is to develop a macrostate analog to the microstate master equation (1.7) with physically meaningful macrostates and transition rates that are consistent with the underlying microstate dynamics. We present two coarse-graining methods. The first (Chapter 2) is a top-down approach where we partition the free energy landscape into basins surrounding local minima and their connecting saddles and compute transition rates by solving eigenvalue problems on small sub-matrices. The second (Chapter 3) is an approach based on simulating the continuous-time Markov chain generated by K, that is the combination of several elements: a procedure to explicitly identify transitions between macrostates in the simulation, a problem-specific goodness-of-clustering test, an algorithm to find the centroid secondary structure for each macrostate, a method to compute macrostate partition functions from many short simulations, and a procedure for computing transition rates via first-passage time simulations with confidence intervals. We apply this method to systems of experimental interest in our laboratory. In Chapter 4 we develop a model problem that allows detailed analysis of the simulation-based method and suggests its wider applicability. Chapter 5 describes the algorithm to compute the centroid in detail and examines the relationship between the centroid and the minimum free energy structure in how they characterize the equilibrium ensemble.

Chapter 2

An Enumerative Approach to Constructing Macrostates Rate Matrices

This chapter presents a top-down approach to finding macrostates and computing transition rates with the goal of developing a macrostate analog to the secondary structure master equation (1.7). The method is the combination of two new ideas: an algorithm to partition the free energy landscape into basins while retaining information about the connectivity of the landscape and a method to compute phenomenological transition rates by solving local eigenvalue problems. The computation of rates follows from the analysis of Widom who analyzed transition rates for a two-basin case in a series of papers in the 1960s [53, 54]. We compare our method to related work by Wolfinger et al. [56], who use a different partitioning approach, and we gain insight into when their method of computing transition rates works.

2.1 Barrier trees and basin graphs: Partitioning the secondary structure landscape

A top-down decomposition of a nucleic acid free energy landscape begins with a procedure for grouping secondary structures into macrostates. General methods exist that are based on, for example, the eigenstructure of the rate matrix [14] or graph partitioning [10]. Alternatively, one could seek to use physical insight into the system to construct macrostates before looking at the rate matrix: Flamm et al. [26] construct a "barrier tree" consisting of all basins (local minima) and saddle points in the free energy landscape (Figure 2.1 (a)). The leaves of the tree correspond to the local minima, while interior nodes correspond to

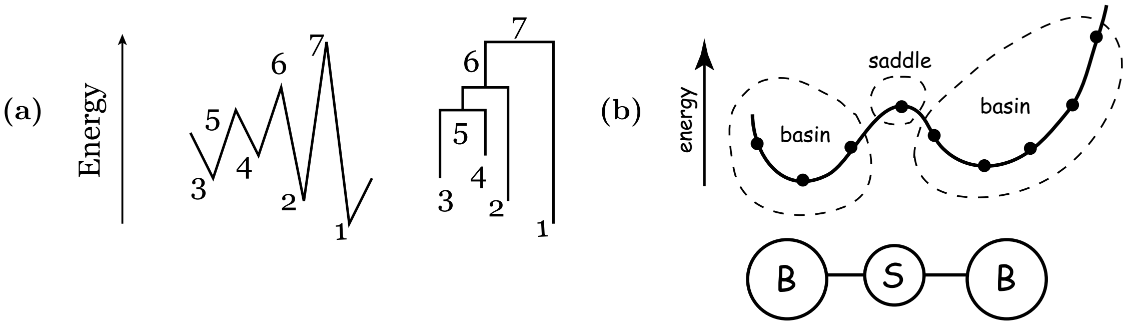


Figure 2.1: (a) A simple free energy landscape and its barrier tree. (b) A free energy landscape and its basin graph.

saddles. The height of a saddle between two leaves represents the highest energy that must be reached to travel between the two minima.

Though the barrier tree provides important information about the height of saddles separating states, it ignores the connectivity of the landscape. For example, in Figure 2.1 it appears from the tree description that to travel from 3 to 1, one must follow the path $3 \to 5 \to 6 \to 7 \to 1$, while in reality 4 and 2 must also be visited. In order to have a representation that is more true to the connectivity of the free energy landscape, we generalize the notion of the barrier tree to a *basin graph*, where the connectivity of the basins and saddle points in the landscape is explicit. This is shown in Figure 2.1 (b).

In order to group secondary structures into basins and saddles, we enumerate all secondary structures, and we assume that they are sorted by increasing energy. A secondary structure satisfies *one* of the following characteristics ("downhill" is with respect to the free energy):

1. The structure has no downhill neighbors: It is a new basin.
2. The structure's downhill neighbors are all part of the same basin: It is added to that basin.
3. The structure's neighbors reside in a number of basins or saddles: It is part of the saddle joining all basins reachable by downhill steps.

The algorithm considers the structures in order of increasing energy and adds them to the appropriate basin or saddle. It requires only a single pass, and since the number of neighbors of a secondary structure for a sequence of length N is at most N^2—there are at most N^2 possible base pairs and neighbors only differ by one base pair—the running

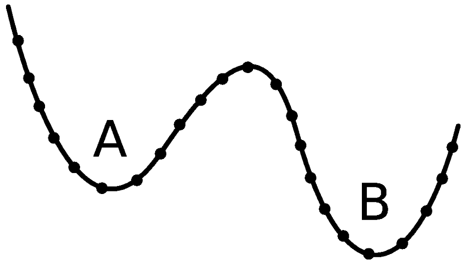


Figure 2.2: A discrete two-basin potential

time is bounded by $N^2|\Omega|$, where $|\Omega|$ is the number of secondary structures. If a group of neighboring structures has the same energy, then they are treated as a single state for the purposes of determining which of (1–3) hold.

Figure 2.1 shows a simple free energy landscape and its basin graph. Figure 2.3 shows a basin graph for the landscape of a 21 nucleotide RNA sequence. The blue, numbered circles represent basins and the purple circles saddles. Basin graphs are the foundation of our macrostate approach, as each basin in the free energy landscape will comprise a macrostate.

2.2 Phenomenological rate constants for a particle in a two-basin landscape

In an insightful analysis, Widom [54] elucidated the relation between the phenomenological reaction rates between two macroscopic substances, A and B, and the underlying transition probabilities between all of the microstates in the system. We will present his whole derivation, since the insight gained will lead us to a coarsening method for complete NA landscapes.

Consider the two-basin landscape with discrete states depicted in Figure 2.2. We assume that the transitions only occur between adjacent states and the rates between states obey detailed balance with respect to the Boltzmann equilibrium distribution. Intuitively, a particle started in any state within one basin will quickly reach “equilibrium” within that basin. On longer time scales it will make transitions between the basins, and it is this transition rate that we wish to derive.

We assume that the dynamics are governed by a master equation with rate matrix K

(Section 1.3). The solution for a particular microstate, i, is

$$p_i(t) = \pi(i) + c^1 v_i^1 e^{-\lambda_1 t} + \cdots + c^{|\Omega|-1} v_i^{|\Omega|-1} e^{-\lambda_{|\Omega|-1} t}, \tag{2.1}$$

where the v^j are the eigenvectors of K and the c^j are constants that depend on the initial condition. The barrier between the two basins suggests that there is a separation of time-scales with the slowest scale corresponding to transitions between the basins. Thus, $\lambda_1 \ll \lambda_2$, and for $t \gg (\lambda_2 - \lambda_1)^{-1}$ the solution is approximated by

$$p_i(t) = \pi(i) + c^1 v_i^1 e^{-\lambda_1 t}. \tag{2.2}$$

Differentiating this expression and substituting the result into itself gives

$$\frac{dp_i(t)}{dt} = \lambda_1 \left[\pi(i) - p_i(t)\right]. \tag{2.3}$$

Since we are interested in populations of the two basins and not individual microstates, we define

$$p_A(t) = \sum_{i \in A} p_i(t) \quad \text{and} \quad \pi_A = \sum_{i \in A} \pi(i), \tag{2.4}$$

and similarly for B. Then, by conservation of mass, $p_A(t) + p_B(t) = 1$, and

$$-\frac{dp_A(t)}{dt} = \frac{dp_B(t)}{dt} = k_f p_A(t) - k_b p_B(t), \tag{2.5}$$

where

$$k_f = \lambda_1 \pi_B \quad \text{and} \quad k_b = \lambda_1 \pi_A. \tag{2.6}$$

We call k_f and k_b the *phenomenological rate constants.* These are the transition rates that would be measured in an experiment. A generalization of the method described here will allow us to compute rate constants between all pairs of basins in a NA free energy landscape.

2.2.1 Distinguishing rate constants

At equilibrium, the flow of probability from A to B is

$$k_{BA}^{eq} = \sum_{i \in A} \sum_{j \in B} k_{ji} \frac{e^{-\Delta G(i)/k_B T}}{Q_A}, \tag{2.7}$$

where $e^{-\Delta G(i)/k_B T}/Q_A$ is the probability of being in state i conditioned on being in basin A. The following relation holds:

$$\frac{\pi_B}{\pi_A} = \frac{k_f}{k_b} = \frac{k^{eq}_{BA}}{k^{eq}_{AB}}. \tag{2.8}$$

However, the rates are not equivalent: The phenomenological rates, k_f and k_r depend, through λ_1, on all microstate transitions in the system, while the equilibrium flows, k^{eq}_{BA} and k^{eq}_{AB}, depend only on those rates that cross the dividing surface between A and B. The equilibrium flows are always greater than the phenomenological rates, but when the step that crosses the border between A and B is rate limiting, the equilibrium flows may be a good approximation to the phenomenological rates [53]. We call these flows the *local equilibrium* (LE) rates.

Although they are not the correct rate constants away from equilibrium, several authors have used the equilibrium flows to construct macrostate rate matrices. Wolfinger et al. [56] identify macrostates with local minima on the barrier tree and compute the equilibrium flows between minima. Zhang and Chen [59] also compute LE rates, but they use a simplified energy model and partition the landscape by hand, searching for rate-limiting base stacks. The approach we outline in the following section uses the correct phenomenological rates.

2.3 Dominant local relaxation

Having presented basin graphs and the method to compute phenomenological transition rates, we now outline our coarsening approach. We identify a macrostate with each basin in the basin graph and compute phenomenological rates to determine transitions between them. In particular, we construct the basin graph for a sequence, identifying B basins. After rearranging indices, each basin corresponds to a block on the diagonal of the microstate rate matrix. For each pair of connected basins, we compute the transition rates by computing the smallest nonzero eigenvalue for the sub-matrix composed of the basins' two diagonal blocks and the corresponding off-diagonal blocks.

Denoting the eigenvalue for pair (a, b) by $\lambda_1^{(a,b)}$, the transition rates between a and b are

$$k_{ba} = \frac{\lambda_1^{(a,b)} Q_b}{Q_a + Q_b} \quad \text{and} \quad k_{ab} = \frac{\lambda_1^{(a,b)} Q_a}{Q_a + Q_b}.$$

For basins that are not connected, the rates are zero. The time-varying probability of basin

a is

$$p_a(t) = \sum_{b=1}^{B} \left[k_{ab} p_b(t) - k_{ba} p_a(t) \right]. \tag{2.9}$$

We call this approach *dominant local relaxation* (DLR). This has several advantages over a method that considers the entire rate matrix at once. First, the method allows us to compute eigenvalues for small sub-problems rather than for the entire rate matrix. Further, the macrostates have a clear physical interpretation as minima in the free energy landscape and group similar structures. Importantly, the transition rates are the phenomenological rates that would be measured in an experiment.

If all saddles in the free energy landscape connected only two basins, the procedure outlined above would work as described. In reality, a saddle may connect arbitrarily many basins (as is illustrated in Figure 2.3). In the case of a saddle connecting L basins, we approximate the solution for $t \gg (\lambda_L - \lambda_{L-1})^{-1}$ by

$$p_i(t) = \sum_{j=0}^{L-1} c^j v_i^j e^{-\lambda_j t}, \tag{2.10}$$

where $v^0 = \pi$, $\lambda_0 = 0$, and $c^0 = 1$ (see (1.10)). As before, we sum the micro-states within each of the L basins, to get

$$p_A(t) = \sum_{i \in A} p_i(t) = \sum_{i \in A} \sum_{j=0}^{L-1} c^j v_i^j e^{-\lambda_j t} = \sum_{j=0}^{L-1} c^j \bar{v}_A^j e^{-\lambda_j t}, \tag{2.11}$$

where $\bar{v}^j$ is a macrostate eigenvectors whose entries are the sum of v_i^j within each macrostate. Thus, a similarity transformation for these grouped eigenvectors yields

$$k_{dlr} = \bar{V} \bar{\Lambda} \bar{V}^{-1}. \tag{2.12}$$

where $\bar{V}$ is the $L \times L$ matrix formed from the vectors $\bar{v}_i^j$, and $\bar{\Lambda}$ is the matrix with $\lambda_0, \ldots, \lambda_{L-1}$ along the diagonal. The matrix k_{dlr} contains transition rates among the L basins. Since saddles may connect many basins, and the saddles may be interleaved (if basins 1,2,3 are connected by a 3-way saddle as are 3,4,5, then basins 1–5 must be treated at once), we may end up solving for the eigenvalues of the entire rate matrix at once. To avoid this, we assume that the two-way saddles are the most important in capturing the dynamics, and only perform pairwise calculations.

2.3.1 Examples

Our first example is the 21 nucleotide RNA sequence whose basin graph and master-equation solutions are shown in Figure 2.3. The size of the circles qualitatively indicate the basin or saddle's equilibrium probability. In addition to computing rates via the DLR method, we compare the LE approach and, as a control, solve the microstate master equation and group the solution into basins. We see from Figure 2.3 that the DLR solution, calculated using only two-way saddles (pairs of basins), is virtually indistinguishable from the grouped microstate solution. The LE solution, on the other hand, significantly overestimates the rate of transition into the frustrated state (2).

We now show a larger example that tests the practical limits of this method. We study a 39 nucleotide RNA with 3.67×10^7 secondary structures. This is too many structures to perform a basin decomposition, so we instead consider the 1.57×10^6 structures within 25 kcal/mol of the minimum free energy. (The starting conformation is 15.5 kcal/mol above the MFE.) The method identifies 3780 basins. Constructing the basin graph and computing the DLR and LE rates took approximately 80 hours to run on a single processor. Figure 2.4 shows the macrostate solution using both the LE and DLR rates. (This problem is too large to solve the microstate master-equation.) The LE approach seems to significantly over-estimate the transition rate out of the starting basin, labeled 283, and the approach to equilibrium, but without a microstate solution for comparison we cannot make firm conclusions.

2.3.2 Saddle assignments

As Widom [53] noted, the local equilibrium rates are sensitive to the exact boundary between the two basins. This suggests the question: How dependent is the improvement of the DLR rates over the LE rates on the choice of dividing surface? We answered this question by considering four methods for assigning each state in a saddle to a basin:

1. Add the state to the lowest-energy neighboring basin.

2. Add the state to the highest-energy neighboring basin.

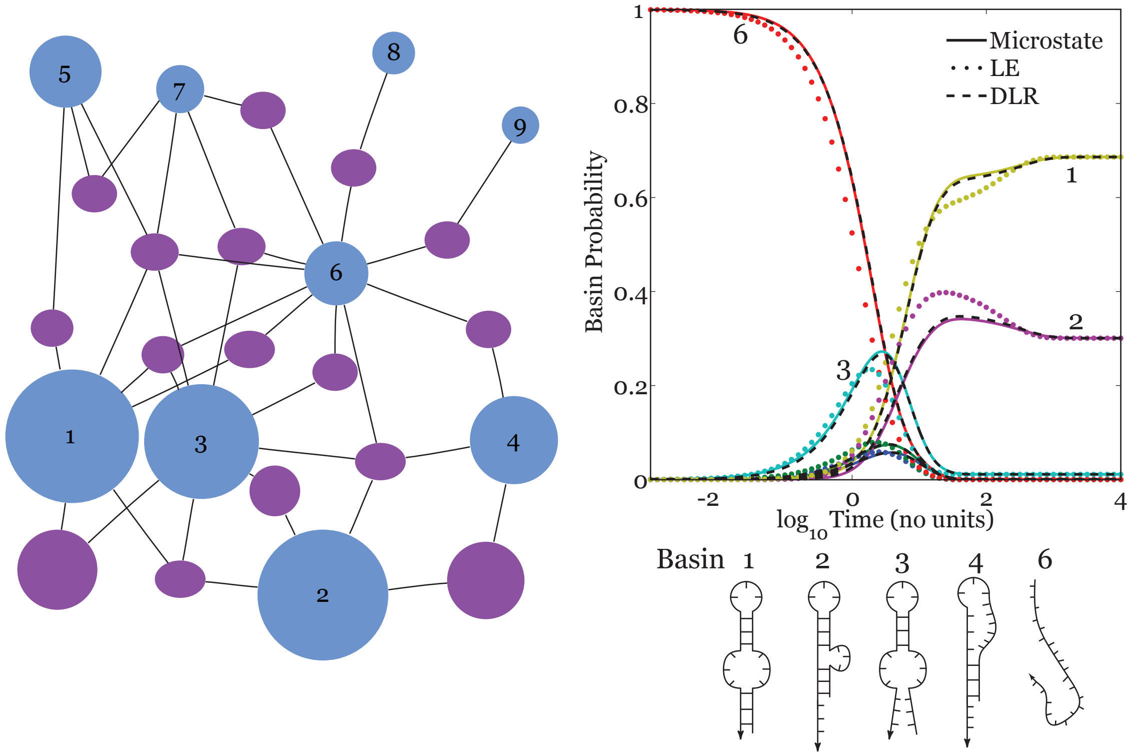


Figure 2.3: (Left) A basin graph for the 21 nucleotide RNA sequence GGAACUGGCUAUGCCUCCUCC that has 250 microstates. The numbered blue circles are basins and the purple circles saddles. The size of a circle is scaled by the basin or saddle's equilibrium probability. (Right) Solutions to the master equation on the basins by solving the microstate problem, the LE approximation, and DLR. States in saddles have been grouped into basins via steepest descent. The structure drawings show the minima corresponding to the numbered basins.

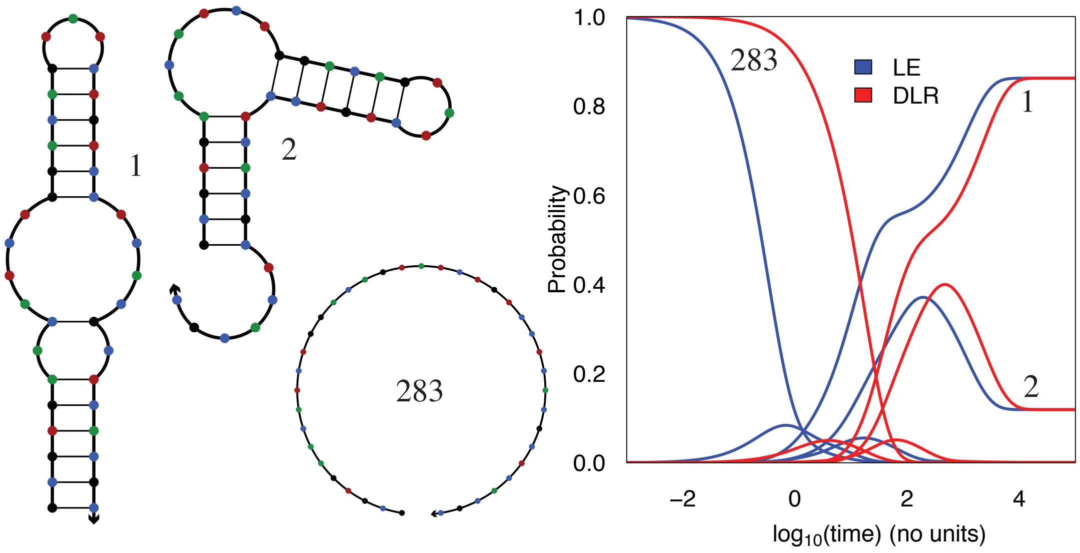


Figure 2.4: Master equation solution for the 39 nucleotide RNA `GCGUGAACAUCUGGACAGUAUCUGUCCUCACGCUCACGC` using the LE and DLR rates. The numbered structures correspond to basins in the free energy landscape. Saddle states were grouped into basins by steepest descent.

3. Add the state to the basin of a random neighbor, ensuring that each basin remains connected.

4. Add the state to the basin of its lowest-energy neighbor. This is a discrete version of steepest-descent.

To make the difference between (1) and (4) clear: The low basin method chooses the lowest basin according to the depth of the minimum in the basin. The steepest descent method looks only at neighboring structures and chooses the basin of the lowest-energy neighbor. Figure 2.5 compares the four saddle assignment methods for a 27 nucleotide RNA sequence. The DLR solution is very close to the microstate solution for all assignment methods, though it is slightly off for the high basin method. Although the computation of the DLR eigenvalue does not depend on the precise dividing surface between basins, moving states from one basin to the other changes the relative size of their partition functions slightly. More importantly, saddle assignments may affect the DLR rates because we only compute the rates between pairs of basins, even though higher-order saddles exist.

As expected, the LE rates display much greater sensitivity to the saddle assignment method, performing particularly poorly for the low basin and high basin methods. More

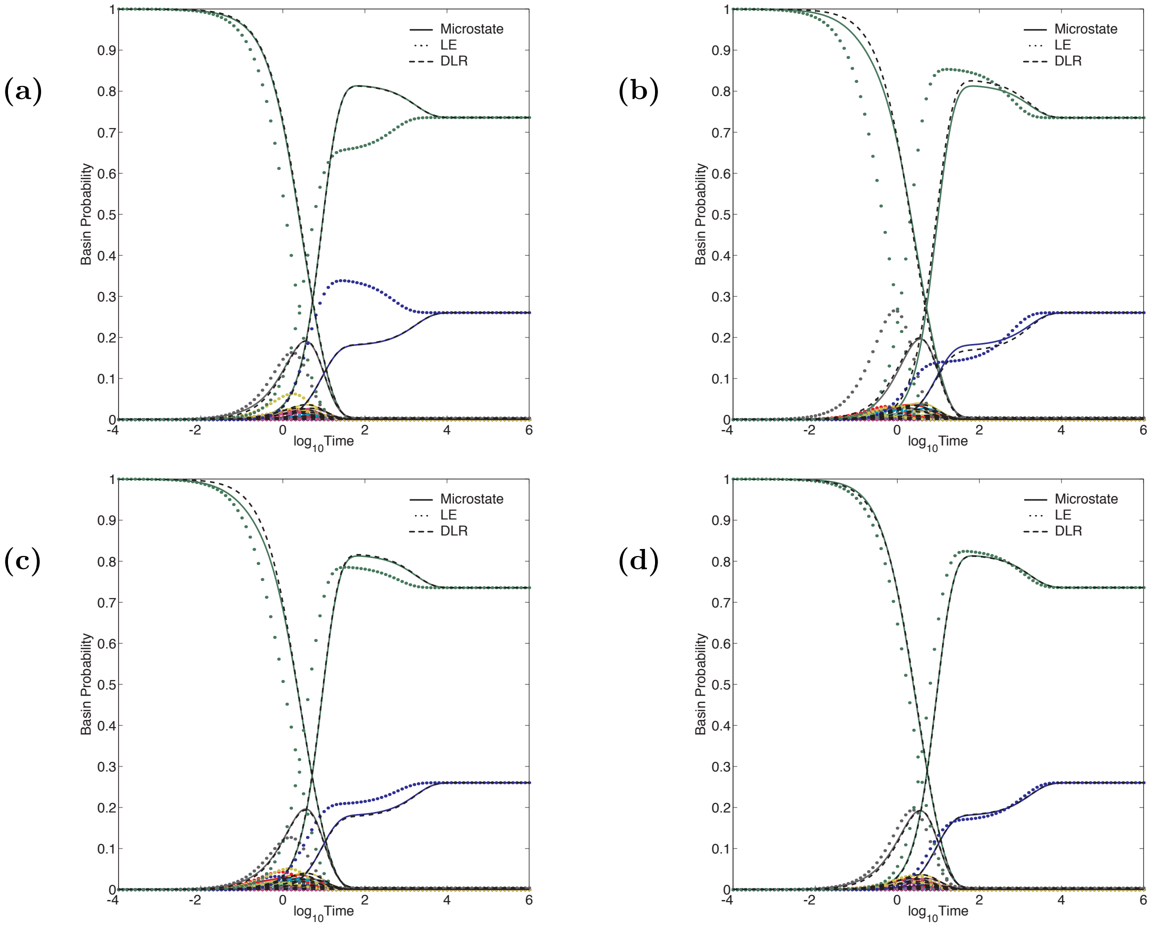


Figure 2.5: Comparing saddle assignment methods for the 27 nucleotide RNA sequence, `GUGAACCUGGACUAUGUCCUCACUCAC`: (a) low basin, (b) high basin, (c) random basin, and (d) steepest descent.

surprising is how close to the microstate solution the LE solution with steepest descent saddle assignments is, only slightly overestimating the major transition rates.

It seems that the steepest descent saddle assignment method creates basin divisions that make the microstate transitions crossing the dividing surface rate-limiting. An intuitive justification for why that would be the case is as follows. Recall from (2.6) that

$$k_f + k_b = \lambda_1 \pi_B + \lambda_1 \pi_A = \lambda_1. \tag{2.13}$$

We seek to describe the analog to λ_1 for the LE rates. Substituting (1.9) into (2.7) and replacing k_f and k_b in (2.13), we find

$$\begin{aligned}
\lambda_{EQ} &= k^{eq}_{BA} + k^{eq}_{AB} \\
&= \sum_{i \in A} \sum_{j \in B} k_{ji} \frac{e^{-\Delta G(i)/k_B T}}{Q_A} + k_{ij} \frac{e^{-\Delta G(j)/k_B T}}{Q_B} \\
&= \sum_{i \in A} \sum_{j \in B} e^{-(\Delta G(i) - \Delta G(j))/2k_B T} \frac{e^{-\Delta G(i)/k_B T}}{Q_A} + e^{-(\Delta G(j) - \Delta G(i))/2k_B T} \frac{e^{-\Delta G(j)/k_B T}}{Q_B} \\
&= \sum_{i \in A} \sum_{j \in B} \frac{e^{-(\Delta G(i) + \Delta G(j))/2k_B T}}{Q_A} + \frac{e^{-(\Delta G(j) + \Delta G(i))/2k_B T}}{Q_B} \\
&= \sum_{i \in A} \sum_{j \in B} \frac{Q_A + Q_B}{Q_A Q_B} e^{-(\Delta G(i) + \Delta G(j))/2k_B T}.
\end{aligned}$$

Since $\lambda_{EQ} \geq \lambda_1$ (Section 2.2.1), we derive the best approximation by choosing the dividing surface to minimize λ_{EQ}. Because the choice of dividing surface does not change Q_A or Q_B much, we achieve this by choosing the states on either side of the dividing surface to have as high an energy as possible. The steepest descent saddle assignment does this by keeping steep edges within a basin and placing shallow edges, which connect two high-energy states, between basins.

Thus, with an appropriate choice of dividing surface, the LE rates can perform nearly as well as the DLR rates. Further, analysis of λ_{EQ} suggested that the steepest descent method for determining the dividing surface might be a near-optimal strategy.

2.4 Summary and outlook

Dominant local relaxation is an appealing approach to constructing a reduced-size master equation for secondary structure kinetics. Identifying macrostates with nodes in the basin graph, which represent local minima in the free energy landscape, ascribes a clear physical meaning to the macrostates. By computing transition rates separately for each pair of connected basins, we replace a potentially intractable, large eigenvalue calculation on the whole rate matrix with many smaller ones. Further the rates have a physical meaning. In contrast to approaches like the local equilibrium approximation, we need not precisely fix the border between macrostates since the eigenvalue computation considers all microstate transitions whether they cross the dividing surface between basins or not. However, our analysis of saddle assignments suggests that with the proper choice of dividing surface, the LE rates perform nearly as well as the DLR rates, and they have the advantage of requiring only a sum over states rather than solving a sparse but potentially ill-conditioned eigenvalue problem.

In practice, neither approach is feasible for sequences of experimental interest because it requires enumerating all secondary structures in the free energy landscape. Since the number of structures grows exponentially with sequence length, no method that requires enumerating secondary structures will be computationally practical. Wolfinger et al. [56] seek to address this problem by considering only secondary structures within a given energy gap of the minimum free energy. Though this allows for the analysis of somewhat longer sequences, the energy gap must decrease as the sequence length increases. Synthetic DNA machines rely on large changes in free energy for their functioning, and any approach that considers only structures near the minimum free energy will have little to say about such a system. An alternative approach might be to sample structures from the equilibrium distribution, then identify basins from the sample. This requires fewer structures than the enumeration approach, but landscape features that are not significantly represented in the equilibrium ensemble, such as a high energy starting state or metastable intermediate states, will be missed. These limitations led us to pursue an alternative approach based on simulations, which is presented in the next chapter.

Chapter 3

Simulation-Based Coarse-Graining of Nucleic Acid Free Energy Landscapes

The coarse-graining approach presented in the previous chapter and related work are all limited by the fact that for problems of interest, the list of microstates is too large to write down. Answering this question of how to solve a problem that is too large to write down is the central goal of this work, and in this chapter we present a solution. Clearly, we cannot solve the microstate equations, but we identify physically meaningful macrostates and compute transition rates between them.

The approach relies on the ability to efficiently simulate secondary structure kinetics as a continuous time Markov chain [25]. Simulations of the kinetics do not immediately give an useful picture of the free energy landscape because the simulations are difficult to interpret. However, an advantage of simulations is the ability to start them at a structure of experimental interest and explore the features of the landscape important to the folding from that starting point, which may not be well represented in the equilibrium ensemble. Our method addresses the issue of interpretation by identifying physically meaningful macrostates and computing transition rates between them.

3.1 Method

Simulating the continuous-time Markov chain generated by the secondary structure rate matrix, K (Section 1.3), can be done efficiently, without storing K, by computing transition rates as needed at each step [25]. That is, we construct the single sparse column of

K corresponding to the current state, and use the rates to determine the exit time from the current state and the subsequent state. Recent work has extended the simulations to multiple interacting strands and improved the computational complexity of computing the rates by exploiting the loop-based structure of the energy model [44]. This means that we can simulate the chain long enough to observe several macrostate transitions. Thus, we seek a method to identify these macrostate transitions within a simulation and characterize the secondary structures in each macrostate. To do this we search for segments of the trajectory over which the Markov Chain appears stationary and points at which large scale transitions occur (Step 1). Then we cluster the resulting pair probability distributions over each stationary segment into macrostates and find a centroid structure for each macrostate (Step 2), compute macrostate partition functions via a large number of very short simulations (Step 3), and compute transition rates by estimating first-passage times from simulations between macrostates (Step 4). Finally, we compute macrostate initial conditions from additional short simulations (Step 5). The following sections address each of these steps.

3.1.1 Locating transitions

To identify macrostate transitions we must develop a measure of how close the chain is to local equilibrium over a short period of time. Given a simulation trajectory $X_t,\ 0 \le t \le T$, with $X_t \in \Omega$, we choose a length of time, τ, longer than the time to reach local equilibrium, and compare the vector of empirical measures for sliding sub-trajectories of length τ. For the sub-trajectory from time a to b, $X_{[a,b]}$, the empirical measures are computed by

$$\mu_s(X_{[a,b]}) = \frac{1}{b-a} \int_a^b \mathbb{1}[X_t = s]dt,\ \forall s \in \Omega. \tag{3.1}$$

We compare these empirical measures at each time t via the distance in variation,

$$d_V\left(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]})\right) = \frac{1}{2} \sum_{s \in \Omega'} |\mu_s(X_{[t-\tau,t]}) - \mu_s(X_{[t,t+\tau]})|, \tag{3.2}$$

summing only over $\Omega' \subset \Omega$, the structures with nonzero empirical measure. In practice, $|\Omega'| \ll |\Omega|$, which is why the calculation of (3.2) is practical. Since $\mu_s(X_{[a,b]})$ is a probability measure, $d_V \in [0,1]$. Assuming that the trajectory is ergodic within each macrostate, if τ is

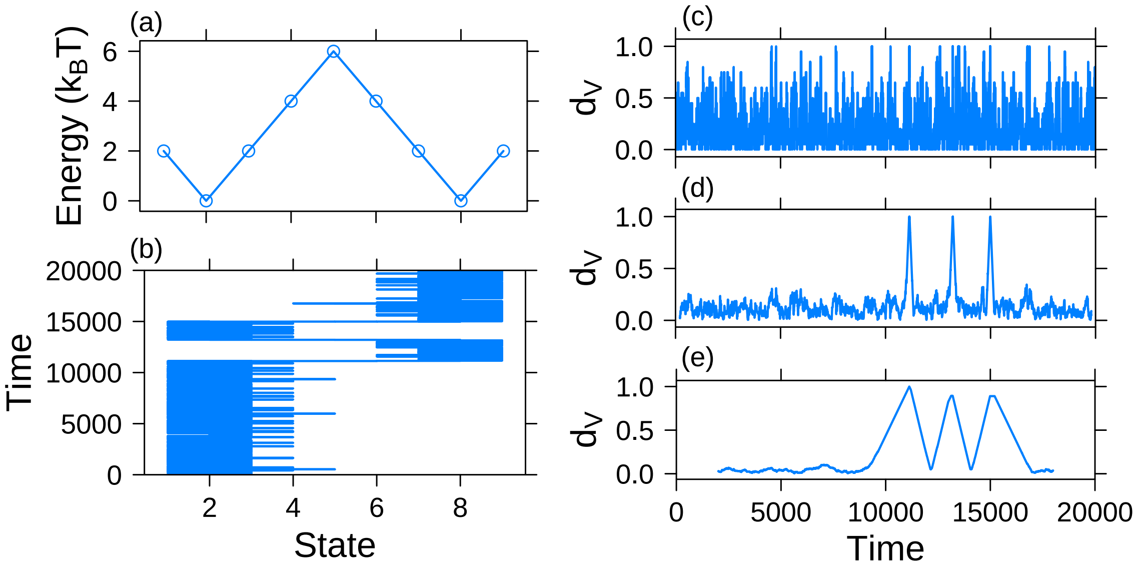


Figure 3.1: Discrete two-well free energy landscape (a), and a simulation of Kawasaki dynamics on that landscape (b). The distance in variation computed for that simulation with $\tau = 20$ (c), 200 (d), and 2000 (e).

longer than the local relaxation time and the trajectory remains in a single macrostate over the interval $[t-\tau, t+\tau]$ then $d_V \approx 0$. Alternatively, if a macrostate transition occurs at time t, $\mu(X_{[t-\tau,t]})$ and $\mu(X_{[t,t+\tau]})$ represent the local equilibrium distributions for different macrostates and $d_V \approx 1$. To gain an intuitive understanding of the identification step, consider Figure 3.1. Panel (a) shows a discrete two-well free energy landscape, and (b) shows a simulation of the Kawasaki dynamics (1.9) on that landscape. Note that there are three transitions between wells. Panels (c)–(e) show the distance in variation computed for this simulation with $\tau = 20,\ 200$, and 2000. For $\tau = 20$ there is too much noise to clearly identify the macrostate transitions, but they are clearly identifiable for $\tau = 200$. When $\tau = 2000$ the transitions are over-smoothed and, though the transitions are visible to the eye, the measure d_V does not reach unity for the second two transitions.

For nucleic acid problems, we do not know which τ should be chosen *a priori* since that would require prior knowledge of the relevant timescales in the kinetics for a particular sequence. In practice we make an initial guess; if the method identifies macrostates with mean exit time on the order of or shorter than τ, a longer τ should be chosen. If we identify few or no macrostates, we may try a shorter τ to see if there are additional macrostates that are important at shorter timescales.

Thus, we scan the trajectory X_t for $0 \leq t \leq T$ and identify points where the distance in variation is close to unity. In practice we must define a threshold γ that quantifies the notion "close to unity." We find the set of intervals

$$T^{(\gamma)} = \left\{ t \mid d_V \left(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]}) \right) \geq \gamma \right\}, \tag{3.3}$$

and choose the time of maximum d_V within each interval

$$T_k^* = \operatorname*{argmax}_{t \in T_k^{(\gamma)}} d_V \left(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]}) \right). \tag{3.4}$$

Define $m = |T^*| + 1$ and augment T^* with the points $T_0^* = 0$ and $T_{m+1}^* = T$. The subtrajectories between successive points of T^* are contained within a single macrostate, so we slice the trajectory at these points, defining

$$X^i = X_{[T_{i-1}^*, T_i^*]}, \; i = 1, \ldots, m. \tag{3.5}$$

3.1.2 Clustering macrostates and finding macrostate centroids

Macrostates could be characterized by their empirical measures; however, representing the macrostates by pair probability matrices is both more memory-efficient and lends itself to finding a representative secondary structure for each macrostate. For each segment, we compute the pair probability matrix from the empirical measure over that segment (Section 1.1),

$$P^i = \rho(\mu(X^i)), \; i = 1, \ldots, m. \tag{3.6}$$

The trajectory may visit the same macrostate several times. To avoid over-counting macrostates, we cluster similar pair probability distributions to find the distinct macrostates. We use a hierarchical agglomerative clustering algorithm. To begin, each pair probability matrix is its own cluster. All inter-cluster distances are computed and the closest two clusters are merged. This is repeated until a stopping criterion is satisfied [31]. (See Figure 3.2 and Jain et al. [29] for a more detailed explanation of the clustering procedure.)

Though we are interested in defining macrostates, the simulations underlying our method are at the level of secondary structures. In particular, to compute first-passage times (Step 4), we must define starting and ending configurations as particular secondary structures. For this reason, we seek to define a representative secondary structure for each macrostate.

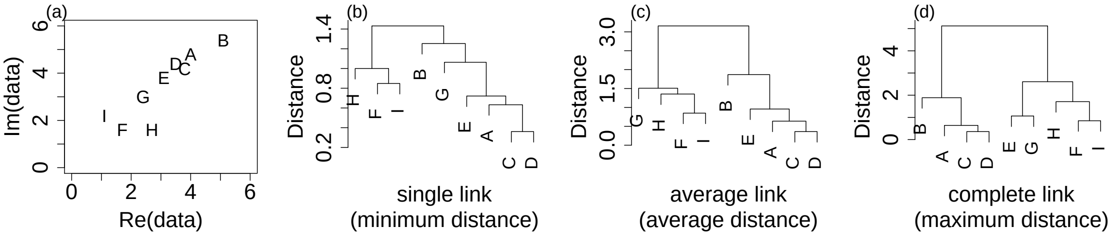


Figure 3.2: Hierarchical clustering of two-dimensional data. (a) Points in the complex plane that are to be clustered. The hierarchical agglomerative clustering algorithm begins with each data point as the representative of its own cluster and repeatedly merges the two closest clusters. The output can be represented as a dendrogram (b)–(d) where two nodes are joined in the tree at the distance at which they were merged. The three panels differ in the method used to compute the inter-cluster distances. (b) The "single link" method uses the minimum distance between two clusters; (c) the "average link" uses the average distance between points in each cluster; (d) the "complete link" uses the maximum distance between points.

Such representative structures also aid in interpreting the results.

We simultaneously address the problem of determining a stopping criterion and finding a representative secondary structure. To choose a representative structure we find the centroid of the macrostate with respect to the ℓ^1 norm, that is, the structure with the smallest probability-weighted distance to all structures in the macrostate. If we denote the probability space of macrostate k by (Ω, p^k), the centroid structure satisfies

$$\begin{aligned} s^{\text{cent}} &= \operatorname*{argmin}_{s\in\Omega} n(s) \\ &= \operatorname*{argmin}_{s\in\Omega} \sum_{\sigma\in\Omega} p^k(\sigma)\|S(\sigma) - S(s)\|_1 \\ &= \operatorname*{argmin}_{s\in\Omega} \left[N - \sum_{j=1}^{N}\sum_{i=1}^{N+1} S(s)_{i,j} P^k_{i,j} \right]. \end{aligned}$$

This optimization problem can be solved efficiently via dynamic programming. The quantity $n(s^{\text{cent}})$ gives a measure of how tightly clustered the secondary structures in the macrostate are. We will explain how to find s^{cent} in more detail in Chapter 5.

A wide variety of approaches exist to determine the optimal number of clusters to choose in a hierarchical clustering procedure. (See Maulik and Bandyopadhyay [36] for explanations of several approaches.) These all seek to balance having a small number of well-separated clusters with having tightly packed clusters. In our situation, the pair probability matrices being clustered are themselves distributions of secondary structures. Our goodness-of-clustering function should take into account the closeness of the secondary structures within the macrostate and not just the closeness of the pair probability matrices.

With this in mind, we choose the number of macrostates that maximizes the following quantity:

$$\mathcal{C}(B) = \frac{\displaystyle\min_{\substack{m,n=1,\ldots,B\\ m\neq n}} \|s_m^{\text{cent}} - s_n^{\text{cent}}\|_1}{\left(\displaystyle\frac{1}{B}\sum_{m=1}^{B} \min_{\substack{n=1,\ldots,B\\ m\neq n}} \|s_m^{\text{cent}} - s_n^{\text{cent}}\|_1\right)\left(\displaystyle\frac{1}{B}\sum_{m=1}^{B} n(s_m^{\text{cent}})\right)}. \tag{3.7}$$

The sum of $n(s^{\text{cent}})$ over the macrostates on the bottom rewards tightly packed clusters. The minimum inter-cluster distance over all pairs, in the numerator, divided by the average minimum inter-cluster distance rewards clusters that are evenly spaced. In particular, the minimum in the numerator ensures that a number of macrostates will not be chosen so that two macrostate centroids coincide.

3.1.3 Computing partition functions

Next, we wish to approximate the partition function for each macrostate. For (Ω, π) this can be calculated via dynamic programming [37], but since secondary structures are not explicitly enumerated, we cannot easily find partition functions for individual macrostates. We can, however, compute approximations to the macrostate partition functions. The insight is that a simulation started at the centroid of a macrostate and run for time τ should end within the macrostate, since the transition time is much greater than τ. Additionally, the end point of that simulation should be an independent sample from the equilibrium distribution within the macrostate, since the mixing time is, by construction, shorter than τ. Thus, to compute the partition function we run a simulation of length τ and form the set A of the distinct structures visited, computing

$$Q_A = \sum_{s\in A} e^{-\Delta G(s)/k_B T}. \tag{3.8}$$

Then we perform L additional simulations, recording the final state at time τ as X^i_τ, $i = 1, \ldots, L$. We estimate the partition function for the macrostate by

$$Q_m = \frac{Q_A}{\frac{1}{L}\sum_{i=1}^{L} \mathbb{1}[X^i_\tau \in A]}. \tag{3.9}$$

Intuitively, to estimate the size of the macrostate, we compute the size of a subset, A, then estimate the fraction of the total that A represents. Dividing the size of A by the fraction of the total that A represents gives the total size. By computing confidence intervals for the probability in the denominator, we obtain confidence intervals for the partition function. We compute confidence intervals using the Hoeffding [27] bound, which states that for independent random variables, $Z_1, \ldots, Z_n$ with $0 \leq Z_i \leq 1$, for all i,

$$\mathbf{P}\left[\bar{Z} - \mathbf{E}[Z] \geq t\right] \leq e^{-2nt^2}. \tag{3.10}$$

Rearranging to get bounds on t for a predetermined error probability, α, we find

$$t \leq \sqrt{\frac{-\log \alpha}{2n}}.$$

From the macrostate partition functions we define the macrostate equilibrium distribution

$$\pi^M(i) = Q_i \Big/ \sum_{j=1}^{B} Q_j. \tag{3.11}$$

3.1.4 Computing transition rates

Once we have identified a representative secondary structure for each macrostate, we compute transition rates by simulating transitions between the representative secondary structures and computing first-passage times. The passage times, along with the requirement that the macrostate rate matrix be reversible with respect to π^M allow us to calculate forward and reverse transition rates for each pair of macrostates. We could simulate both forward and backward rates for each pair of macrostates and then check against π^M for reversibility, or we could use the ratio of forward to backward rates to estimate π^M instead of (3.11). However, due to large changes in energy, transitions are often nearly irreversible, and it is impractical to simulate backward rates. We always simulate transitions from states with small π^M to states with larger π^M. In addition, we compute confidence intervals for the rates.

To calculate macrostate transition rates by first-passage time simulations, we assume that first-passage times between macrostates are a good approximation of the transition rates, and that the first passage times are exponentially distributed with a rate λ that we wish to estimate. The assumption of an exponential distribution is consistent with the goal of developing a macrostate master equation, since the transition times in a first-order master equation are exponentially distributed. That is, exponentially distributed transition times are a consequence of the separation of time scales and high energy barrier that characterize correctly identified macrostates (Section 2.3). Given observed passage times, T_i, we can compute the maximum-likelihood estimate of λ,

$$\hat{\lambda} = \frac{1}{\overline{T}_n}, \text{ where } \overline{T}_n = \frac{1}{n}\sum_{i=1}^{n} T_i. \tag{3.12}$$

In addition to an estimate for λ, we would like to compute confidence intervals. Assume that there are only two macrostates. The quantity $\sum_{i=1}^{n} T_i$, a sum of exponentially distributed random variables, is by definition distributed according to a gamma distribution with parameters n and λ,

$$\sum_{i=1}^{n} T_i \sim \text{Gamma}(n, \lambda).$$

Multiplying both sides of this expression by λ/n, and noting the homogeneity of the gamma density implies that

$$\lambda\overline{T}_n \sim \text{Gamma}(n, n).$$

The distribution of $\lambda\overline{T}_n$ is independent of λ so we can use quantiles of Gamma(n, n) to construct a confidence interval for λ with confidence level $1-\alpha$. Noting that by definition Gamma$(n, n) \equiv \chi^2_{2n}$, we have

$$\mathbf{P}\left(\frac{\chi^2_{2n,\alpha/2}}{\overline{T}_n} < \lambda < \frac{\chi^2_{2n,1-\alpha/2}}{\overline{T}_n}\right) = 1-\alpha. \tag{3.13}$$

See Ross [42], Chapter 5, for a complete description of estimating exponential rates and confidence intervals. In practice, we must designate a maximum time for the first-passage simulations, but the simulation may not leave the macrostate before this time (this is

Type I censoring). We can incorporate these non-transitions into our estimate for λ, where trajectories that do not exit a macrostate give estimates of the probability that no transition occurs by time $T_{\max}$. Thus, we compute

$$\overline{T}'_n = \frac{1}{r}\sum_{i=1}^{n} T_i$$

where r is the number of successes and $T_i = T_{\max}$ if no transition occurred. In other words, we can estimate the rate by dividing the total time simulated by the total number of transitions observed.

To construct confidence intervals for Type I censored simulations, we use an approximate method, which is the expression of (3.13) but with the number of successes, r, in place of n. This has the advantage of not requiring a separate implementation from the case where all simulations are successful. It happens to be exact under the related Type II censoring (when one simulates until a given number of passages are observed), and performs well in practice [48].

When there are more than two macrostates in the system, transitions between them are not independent. For example, if we simulate transitions from macrostate 3 to 2 or 1, these transitions are competing processes with rates λ_{23} and λ_{13}. Thus exit times from macrostate 3 will be exponentially distributed with rate $(\lambda_{23} + \lambda_{13})$. For B macrostates the exit time from macrostate i has the following distribution

$$\text{exit time from } i \sim \text{Exp}\left(\sum_{\substack{j=1 \\ j\neq i}}^{B} \lambda_{ji}\right). \tag{3.14}$$

To estimate the individual λ_{ji} we estimate the total exit rate via (3.12) and form confidence intervals via (3.13). Then we estimate the probability for each macrostate that an exit from macrostate i ends in macrostate j. The probability that a transition from macrostate i ends in macrostate j is

$$p_{ji} = \lambda_{ji} / \sum_{\substack{l=1 \\ l\neq i}}^{B} \lambda_{li}, \tag{3.15}$$

and we estimate it by

$$\hat{p}_{ji} = \frac{\#\text{jumps from } i \text{ to } j}{\#\text{exits from } i}. \tag{3.16}$$

In general we compute M passages from s_i, $i = B, \ldots, 2$, and define T_i^m to be a passage time from s_i to one of s_j, $j \neq i$. Let $E_{ji}^m = 1$ if simulation m from s_i ends at s_j and zero otherwise. Then we define,

$$\begin{aligned} \overline{T}_i &= \frac{1}{M} \sum_{m=1}^{M} T_i^m, \; i = 2, \ldots, B \\ p_{ji} &= \frac{1}{M} \sum_{m=1}^{M} E_{ji}^m, \; i = 2, \ldots, B \end{aligned}$$

and define the macrostate rate matrix, $\hat{K}$, by

$$\begin{aligned} \hat{k}_{ji} &= p_{li}/\overline{T}_i \;\; \text{if } i > j \\ \hat{k}_{ij} &= \hat{k}_{ji} Q_i / Q_j \;\; \text{if } i < j \\ \hat{k}_{ii} &= -\sum_{\substack{i=1 \\ i \neq j}}^{B} \hat{k}_{ji}. \end{aligned} \tag{3.17}$$

In constructing confidence intervals for the p_{ji}, the situation where there are three macrostates is a special case. With three macrostates the number of jumps from i to j has a binomial distribution with propensity given by (3.15). We use Wilson's method for confidence intervals, which performs well in practice [1]. Given an estimated value, $\hat{p}$, the ends of the $1 - \alpha$ confidence interval are

$$\frac{\hat{p} + \frac{z_{\alpha/2}^2}{2n} \pm z_{\alpha/2} \sqrt{\frac{1}{n}\left[\hat{p}(1 - \hat{p}) + \frac{z_{\alpha/2}^2}{4n}\right]}}{1 + \frac{z_{\alpha/2}^2}{n}},$$

where $z_{\alpha/2}$ is the $\alpha/2$ quantile of the standard normal distribution.

When there are more than three macrostates, the binomial distribution becomes a multinomial distribution, a problem for which confidence intervals are not well understood. We use a conservative approximate confidence interval [24], with endpoints

$$\hat{p}_{ji} \pm \frac{1.13}{\sqrt{n}},$$

which gives a coverage probability of at least 0.95.

3.1.5 Computing initial conditions

The start structure that we specify for our simulations may not be the centroid of a macrostate. This is not a shortcoming of the method, since by averaging the trajectories over time τ we cannot resolve kinetics on a time-scale shorter than τ. To compute the macrostate distribution at time τ, we run L simulations of length τ and record which macrostate the end structure, X_τ, is closest to. Thus, we estimate

$$p_i(\tau) = \frac{1}{L} \sum_{l=1}^{L} \mathbb{1} \left[\operatorname*{argmin}_{b=1,\ldots,B} \|P^b - S(X_\tau^i)\|_1 = i \right], \; i = 1, \ldots, B. \tag{3.18}$$

3.1.6 Algorithmic implementation and complexity

Algorithm 3.1 shows pseudo-code for the method presented in this section. The pseudo-code assumes that there is one simulation trajectory, X, but in practice we run several simulations from the starting conformation of interest. Step 1 is repeated for each trajectory, giving a larger collection of X^i; Ω' is enlarged to include all secondary structures visited by any of the simulations. Because much of the method involves running many independent simulations, it is easily parallelized. The clustering step must be done on a single processor, but as we will see in the next section, it accounts for a small portion of the overall running time.

Precise analysis of the time and space complexity is impossible because of the stochastic nature of the method and the sequence dependence of the number of macrostates and transition rates. However, we can provide an upper bound on the simulation time, which dominates the overall running time.[1] Assume that the sequence is N nucleotides long, there are b initial trajectories of length T, the averaging window is τ, the number of macrostates found is B, and the maximum time for rate simulations is $T_{\max}$. In the worst case, running a simulation for time t is $O(tN^3)$ [44]. The steps of the method require simulating for the following times:

1. Identification: bT

2. Clustering: no simulation

3. Partition functions: $(L + 1)B\tau$

[1] The clustering step scales quadratically with the number of non-transition segments. Each centroid computation is $O(N^3)$.

```
Step 1 (Identify transitions):
  Input: Simulation trajectory, $X_t, 0 \leq t \leq T$
  Parameters: Averaging window size, $\tau$, and transition threshold, $\gamma$
  Output: Non-transition segments, $X^i, i = 1, \ldots, m$
  Procedure:
    $D_t = d_V\left(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]})\right),\ \tau \leq t \leq T - \tau$
    $T^* = \{t | D_t > \gamma, D_t \text{ is locally maximal}\}$
    $m = |T^*| + 1$
    $T_0^* = 0,\ T_s^* = T$
    $X^i = X_{[T_{i-1}^*, T_i^*]}, i = 1, \ldots, m$
Step 2 (Cluster macrostates and find centroid structures):
  Input: $X^i$ from step 1
  Output: Number of macrostates, $B$, and centroid structures, $s_i, i = 1, \ldots, B$
  Procedure:
    $P^i = \rho\left(\mu\left(X^i\right)\right), i = 1, \ldots, m$
    $B = m$
    repeat
      $d_{ij} = \|P^i - P^j\|_1, i, j = 1, \ldots, B, i \neq j$
      $(i^*, j^*) = \operatorname{argmin}_{i \neq j} d_{ij}$
      $P^{i^*} = \frac{1}{2}\left(P^{i^*} + P^{j^*}\right)$
      $P^k = P^{k+1}, k = j^*, \ldots, B - 1$
      $s_i = \operatorname{argmin}_{s \in \Omega} N - \sum_{j=1}^{N} \sum_{k=1}^{N+1} S(s)_{j,k} P_{j,k}^i, i = 1, \ldots, B$
      $B = B - 1$
    until $\mathcal{C}(B)$ is maximized
Step 3 (Compute partition functions):
  Input: Number of macrostates, $B$, and centroid structures, $s_i, i = 1, \ldots, B$, from step 2
  Parameters: Number of simulations, $L$
  Output:
    Macrostate partition functions, $Q_i, i = 1, \ldots, B$
  Procedure:
    for $i = 1, \ldots, B$
      Simulate $X_t, 0 \leq t \leq \tau$
      $A = \text{unique}(X_t, 0 \leq t \leq \tau)$
      $c = 0$
      for $i = 1, \ldots, L$:
        Simulate $X_t, 0 \leq t \leq \tau$
        $c = c + \mathbb{1}[X_\tau \in A]$
      end
      $Q_A = \sum_{s \in A} e^{-\Delta G(s)/k_B T}$
      $Q_i = L Q_A / c$
    end
    quicksort $s_i$ by $Q_i$
Step 4 (Calculate transition rates):
  Input: Centroid structures, $s_i$, and partition functions, $Q_i$, from steps 2 and 3
  Parameters: Number of simulations, $M$
  Output:
    Transition rates $\hat{k}_{ij},\ i, j = 1, \ldots, B$ (with confidence intervals)
  Procedure:
    for $i = B, \ldots, 2,\ m = 1, \ldots, M$
      $T_i^m$ = simulated passage time from $s_i$ to one of $s_l, l \neq i$
      $E_{li}^m = \mathbf{1}\left[T_i^m \text{ ends at } s_l\right],\ l = 1, \ldots, B,\ l \neq i$
    end
    $\overline{T}_i = \frac{1}{M} \sum_{m=1}^{M} T_i^m,\ i = 2, \ldots, B$
    $p_{ji} = \frac{1}{K} \sum_{k=1}^{K} E_{ji}^k,\ i = 2, \ldots, B,\ j = 1, \ldots, B,\ i \neq j$
    for $i, j = 1, \ldots, B$
      $\hat{k}_{ji} = p_{jii} / \overline{T}_i$       if $i > j$
      $\hat{k}_{ij} = \hat{k}_{ji} Q_i / Q_j$   if $i < j$
      $\hat{k}_{ii} = -\sum_l k_{li}$     if $i = j$
    end
Step 5 (Compute initial conditions):
  Input: Macrostate pair probability matrices, $P^i$
  Parameters: Number of simulations, $L$
  Output: Initial conditions, $p_i(\tau),\ i = 1, \ldots, B$
  Procedure:
    for $l = 1, \ldots, L$
      Simulate $X_t, 0 \leq t \leq \tau$
      for $i = 1, \ldots, B$
        $p_i(\tau) = p_i(\tau) + \frac{1}{L} \mathbb{1}\left[\operatorname{argmin}_{b=1,\ldots,B} \|P^b - S(X_\tau^i)\|_1 = i\right]$
      end
    end
```

Algorithm 3.1: Pseudocode for the trajectory-based method

4. Transition rates: $M(B-1)T_{\max}$

5. Initial conditions: $L\tau$

where in practice we choose $M = 500$ and $L = 1000$, which give appropriately tight confidence intervals for any sequence. In practice, only the identification step and computing the transition rates are significant, so a worst-case estimate of simulating time is $O\left((bT + MBT_{\max})N^3\right)$. Importantly, all steps of the method that process simulations are linear or constant in the length of the simulation. Thus, in terms of factors of T or N, our method adds no extra time complexity.

Only the identification step uses significant memory. The other steps require memory only for pair probability matrices, representative secondary structures for the basins, and vectors of passage times. An unsophisticated implementation of the identification step would require storing the simulation trajectories, the list of distinct secondary structures visited, and the pair probability matrices for each non-transition region of the trajectory. We can avoid storing the complete trajectories by computing the distance in variation and the pair probability matrices as the simulation proceeds. In addition, we can clear the list of visited structures each time we identify a transition. Like the time complexity, the memory limitations on sequence length are sequence dependent, depending on N, τ, the number of moves in a simulation of length τ, and the time between macrostate transitions.

3.2 Examples

3.2.1 RNA hairpin

We first demonstrate the effectiveness of the method on an RNA sequence for which we can solve the microstate master equation. Figure 3.3 shows the solution to the macrostate master equation for the 23 nucleotide RNA sequence `GUCGCGUCGCGUCGCUAUGCGAC`. The secondary structure drawings show the representative structure for each macrostate. As a control, we also show the solution to the microstate master equation where the solution has been grouped into local minima by the method of Wolfinger et al. [56]. Visually the solutions are very close, and the trajectory-based method groups three local minima into a single macrostate. Comparing these three structures, we see that they are all related by a $3'$ stem with other structure on the $5'$ end. Transitions between these basins occur in time shorter than τ, so it is expected that the trajectory-based method groups them into a single macrostate. The macrostate rate matrix should reflect the long time dynamics of the microstate system, which is reflected in the smallest nonzero eigenvalues. Computing the eigenvalues for the micro- and macrostate rate matrices (d) shows that they are indeed very close, and that the confidence intervals contain the microstate eigenvalues.[2] Panel (c) shows the fraction of time spent at each step in the method running on a single processor. The largest fraction is spent in the simulations from which we compute rates, and the other significant step is the initial simulation and transition identification. Solving the microstate master equation took 204 minutes on a single processor, while the trajectory-based method took 425 minutes. Though slower for short sequences, the trajectory-based method can find solutions for sequences that are too long to solve the microstate master equation.

Panel (b) shows the speed gained from running on multiple cores. We see an increase in speed that is sub-linear in the number of nodes. Though the independence of the simulations run on each processor suggests that we should have near-perfect speedup, a simple probabilistic argument shows why that is not the case: The length of each of the simulations from which we estimate transition rates is approximately exponentially distributed. By default we simulate 500 passage times to estimate the rate. On a single processor the expected running time is

$$T_{\text{run}} = \frac{500}{\lambda},$$

where λ is the transition rate. On L processors, the expected running time for a single processor, j, is

$$T^j_{\text{run}} = \frac{500}{L\lambda},$$

but we must wait until all processors have finished to proceed. As L increases, the probability that one of the T^j_{run} is significantly larger than its expectation is large.

As an analogy, imagine that a group of people are waiting to checkout at a supermarket. Customers take 1 minute or 3 minutes to checkout with 50% probability. If there are ten customers in one line, with high probability it will take about 20 minutes for everyone to leave the store. Alternatively, with 10 registers it will take 3 minutes for everyone to leave

[2] We compute confidence intervals for each transition rate, but when there are more than two macrostates this does not translate into confidence intervals for the eigenvalues. In this case, we report the complete range of eigenvalues calculated from 1000 rate matrices whose rates have been sampled uniformly from their 95% confidence intervals.

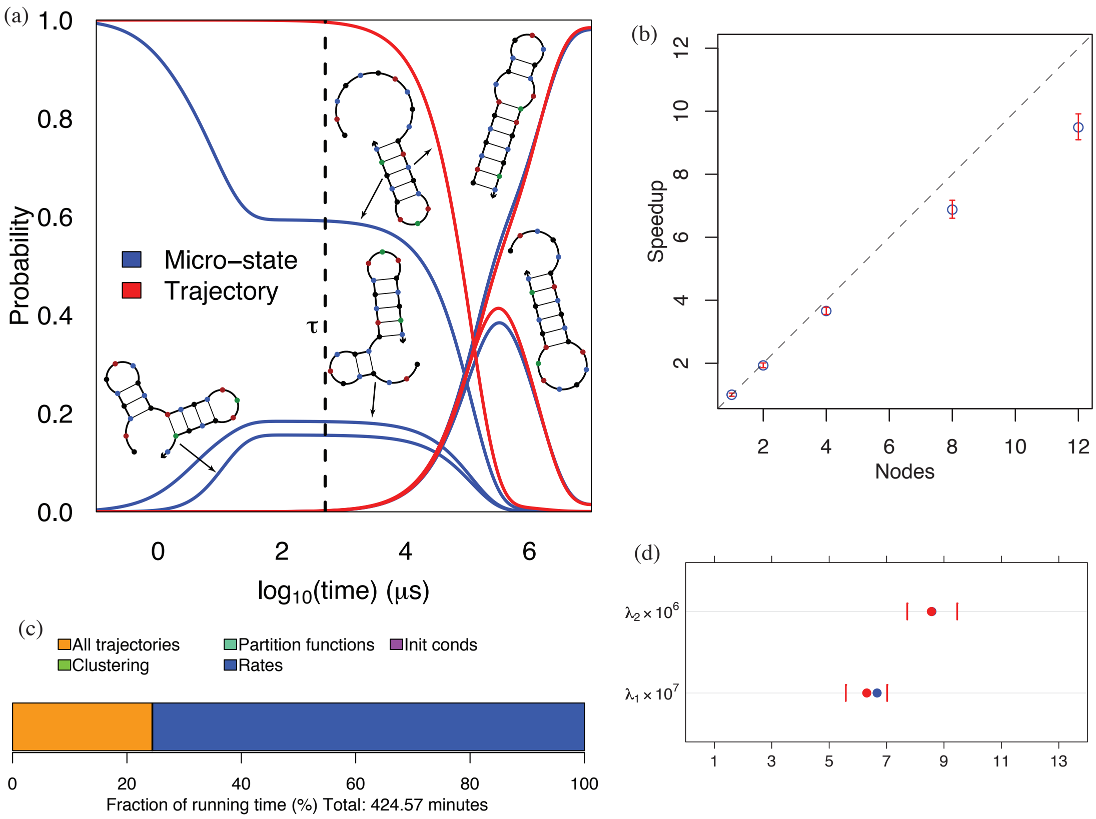
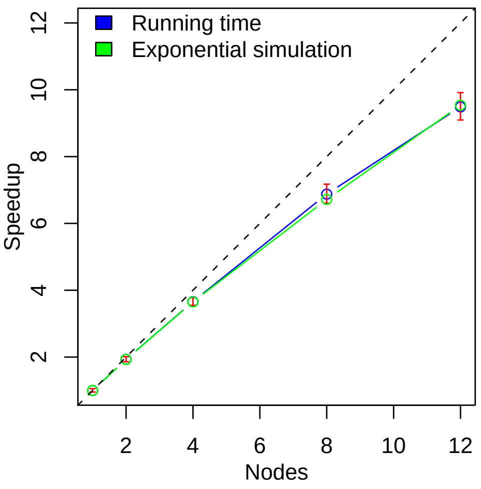


Figure 3.3: (a) Microstate and trajectory-based solutions to the master equation for the 23 nucleotide RNA `GUCGCGUCGCGUCGCUAUGCGAC`. The secondary structure drawings show the representative structures for each macrostate. (b) Speedup from running the trajectory-based method on multiple nodes as compared to the time on a single node. (c) The fraction of total running time spent in each step of the method. (d) Nonzero eigenvalues of the macrostate rate matrix compared with the smallest nonzero eigenvalues of microstate matrix. The bars shows the range of eigenvalues corresponding to 95% confidence intervals for the macrostate transition rates.



Nodes	1	2	4	8	12
Ideal (s)	16288	8144	4072	2036	1357
Actual (s)	16288	8875	4317	2551	1541
		7921	4290	2286	1524
			4048	2247	1516
			3604	2222	1492
				2093	1444
				2008	1418
				1833	1404
				1715	1399
					1397
					1309
					1163
					1136

Figure 3.4: The plot shows a comparison of the speedup in Figure 3.3 (b) to simulations of exponential random variables. The red error bars are for the actual running times. The table shows actual running time for each node from a representative run on $1, 2, 4, 8$, and 12 nodes.

the store—less than perfect speedup.

To test this hypothesis, we drew 504 exponential random variables then grouped them into L 'processors' for $L = 1, 2, 4, 8, 12$. For each L we computed the maximum sum of a group, that is, the computational time for running the simulations. This experiment was averaged over 100 realizations and compared with the actual running times averaged over 10 runs. Figure 3.4 shows strong agreement between the simulated and actual running times. The table in Figure 3.4 shows per-node timing information of a representative run for each number of nodes. As the number of nodes increases, we can clearly see the deviation of the slowest node's run time from the ideal. Thus, though our method should parallelize almost perfectly, it does so only in expected run time, and the actual speedup is less than optimal.

An alternative to assigning an equal number of trajectories to each node is to designate a head node that assigns trajectories to nodes one at a time. Nodes whose simulations take longer will run a smaller total number of trajectories than nodes with faster simulations. In simulating the transitions between a pair of basins, the maximum time that a node would be idle is the length of a single simulation. Designating a node as a master node would reduce the running time by $1/(N-1)$ on N nodes. To reduce running time by $1/N$, we run

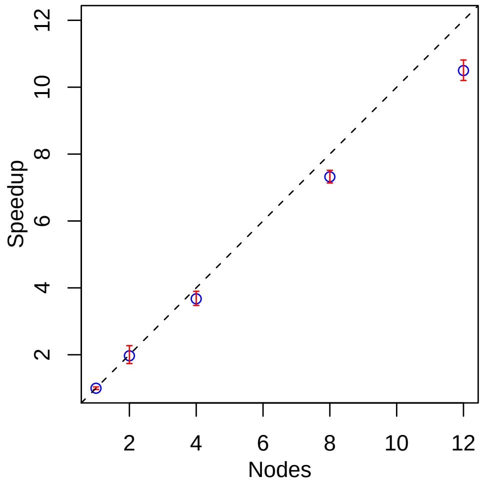


Figure 3.5: Speedup under parallelization using the revised scheduling of rate-calculating trajectories.

$N+1$ MPI processes on the N nodes, designating the first process as the master. Since the time spent assigning trajectories is small, the master MPI process can share a node with a computing process without significant slowdown. Figure 3.5 shows the speedup obtained with this improved trajectory assignment scheme. The scaling is much closer to optimal than when assigning equal numbers of trajectories to the nodes.

3.2.2 Hybridization chain reaction

As a second example, we compare the hybridization chain reaction (HCR) system [23] and a poorly designed variant that has structure in the initiator regions. In this system, the single-stranded initiator (I) opens the hairpin (H1), exposing a single-stranded region that can open hairpin (H2). In this way, a polymer is formed, incorporating as many H1 and H2 hairpins as are present. We would expect that structure in the single-stranded initiator regions would slow the polymerization down. Figure 3.6 compares macrostate solutions for these two systems. Though the polymerized state (1) in the alternate system (b) has higher equilibrium probability, the secondary structure in the initiator regions markedly slows the kinetics. The standard HCR system as shown has 1.16×10^{27} secondary structures, so no enumerative method could be employed to find macrostates.

Focusing just on the standard HCR system, we find the macrostates and transition rates for a system with an initiator, and two copies each of the hairpins $H1$ and $H2$. The total

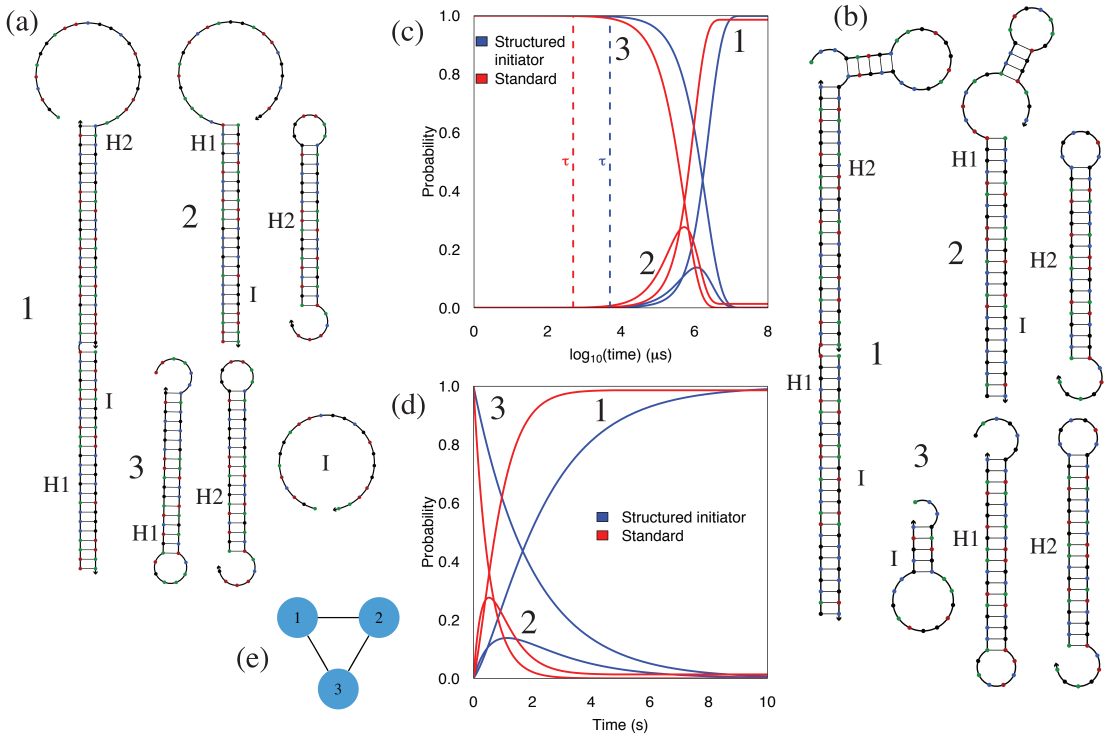


Figure 3.6: HCR [23] (a) and a variant (b) where the initiator sequences have secondary structure in the initiator regions. (c) compares solutions of the two macrostate rate matrices. (d) shows the solution in more detail on a linear time scale. (e) a graph showing the connectivity of the macrostates for both systems. Note that, unlike in Chapter 2, the size of the circles is not scaled by the equilibrium probability of the macrostate. To construct these solutions, 16 trajectories half a second (a) and 5 seconds (b) long were run at $10\mu M$ and $23°C$. The macrostate transition rates are summarized in the following table.

	Standard		Structured initiator	
Macrostate pair	Forward	Reverse	Forward	Reverse
$(2,1)$	1.94×10^{-6}	2.56×10^{-8}	1.41×10^{-6}	2.37×10^{-10}
$(3,1)$	4.69×10^{-7}	6.80×10^{-12}	1.52×10^{-7}	3.85×10^{-16}
$(3,2)$	1.44×10^{-6}	1.58×10^{-9}	3.44×10^{-7}	5.18×10^{-12}

length of the system is 216 nucleotides, and the system has 1.60×10^{51} secondary structures. In an experimental setting the hairpins of like sequence would be indistinguishable, but they must be explicitly labeled for the kinetic simulations. As a result there are 15 macrostates, corresponding to successively adding each of the $H1$ and $H2$ hairpins to the polymer. Figure 3.7 shows both the solution over the 15 macrostates and the solution where indistinguishable macrostates have been grouped. The top plot (b) shows the log time scale in microseconds, while the lower plot (c) shows a linear plot with a scale of seconds. It is notable that the maximum probability reached in macrostate (4) is significantly lower than in (3) or (2). This is for two reasons: First, many of the trajectories leaving macrostate (5) visit one of the lower-energy macrostates without visiting (4) first. Second, from each of (4 a,b) the system has two choices of $H2$ to add to the polymer, effectively doubling the transition rate. Once the polymer has reached state (3) it is committed to an ordering of $H1$s and $H2$s. Note that the running time (e) is dominated by the computation of the transition rates. This is because exit times from 14 of the basins must be computed. Fortunately this step is trivial to parallelize, and could be performed on as many processors as are available.

3.2.3 Three-arm junction

A second example from our laboratory is a catalytic three-arm junction [57]. In this system, an initiator strand allows three hairpins to form a three-way junction. At the last step, the initiator is released so that it can catalyze the formation of additional junctions. Figure 3.8 summarizes the results of running the trajectory-based algorithm on this system. The macrostate identities (a) correlate well with our expectations of how the system functions. Starting in macrostate (5), each step entails opening a hairpin and adding it to the growing structure. The final step from (2) to (1) entails displacing the initiator strand from the junction. Time courses of the macrostate solution (c,d) show that the displacement of the initiator is the rate-limiting step in the reaction. All other steps involve binding to a toehold before the displacement reaction occurs, so it is not surprising that this final step is rate limiting.

As with HCR, the trajectory-based method allows us to easily evaluate alternative designs computationally. Figure 3.9 shows a comparison between the design of Figure 3.8 and an alternative design where a two-nucleotide toehold is introduced to mediate the final step in the reaction. The kinetics of the modified system are significantly faster as

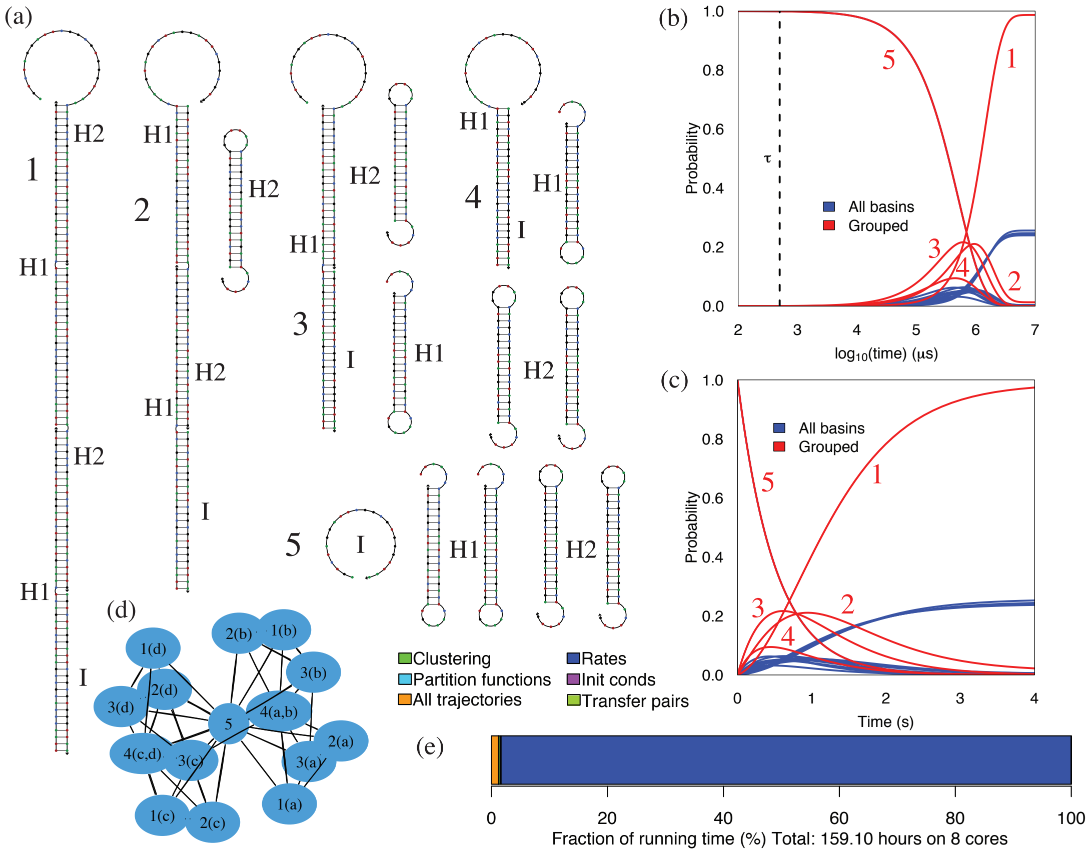


Figure 3.7: HCR[23] with two copies of each hairpin. There are fifteen macrostates corresponding to the various ways of adding the $H1$ and $H2$ hairpins, four copies each of (1), (2), and (3), two copies of (4), and a single copy of (5). The red solution in (b,c) has like basins grouped, while each of the 15 basins is plotted separately in the blue solution. (d) Graph showing the connectivity of the macrostates. (e) Running time. Sixteen trajectories of 1.5 seconds were run at a concentration of $10\mu M$.

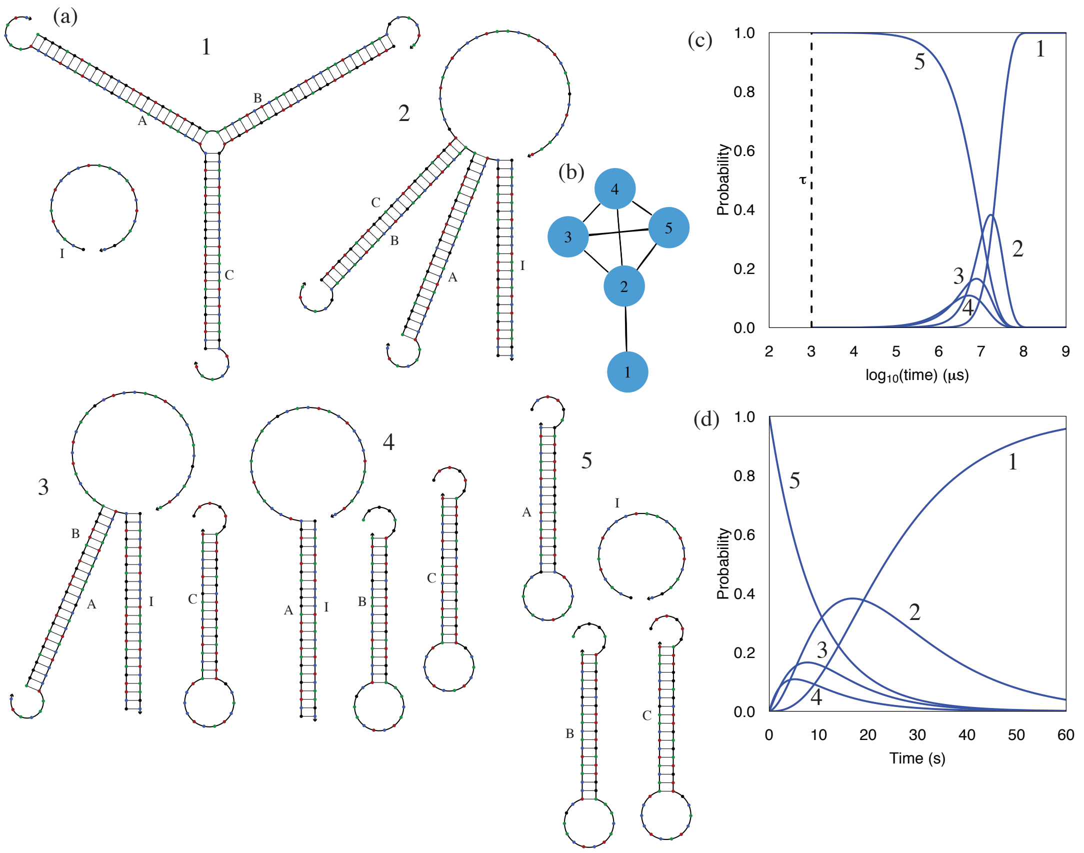


Figure 3.8: Catalytic three-arm junction [57]. The system is 186 nucleotides long. (a) Representative secondary structures for each macrostate. (b) Graph showing the connectivity of the macrostates. Solution of the master equation on log (c) and linear (d) plots. Sixteen trajectories 5 seconds long were run at 23°C and $1\mu M$. The macrostate transition rates are summarized in the following table.

Macrostate pair	Forward	Reverse
$(2, 1)$	8.03×10^{-2}	1.51×10^{-5}
$(3, 2)$	2.88×10^{-1}	9.47×10^{-4}
$(4, 2)$	2.37×10^{-2}	5.16×10^{-8}
$(5, 2)$	6.01×10^{-3}	2.03×10^{-11}
$(4, 3)$	2.94×10^{-1}	1.94×10^{-4}
$(5, 3)$	4.16×10^{-2}	4.28×10^{-8}
$(5, 4)$	6.01×10^{-2}	9.38×10^{-5}

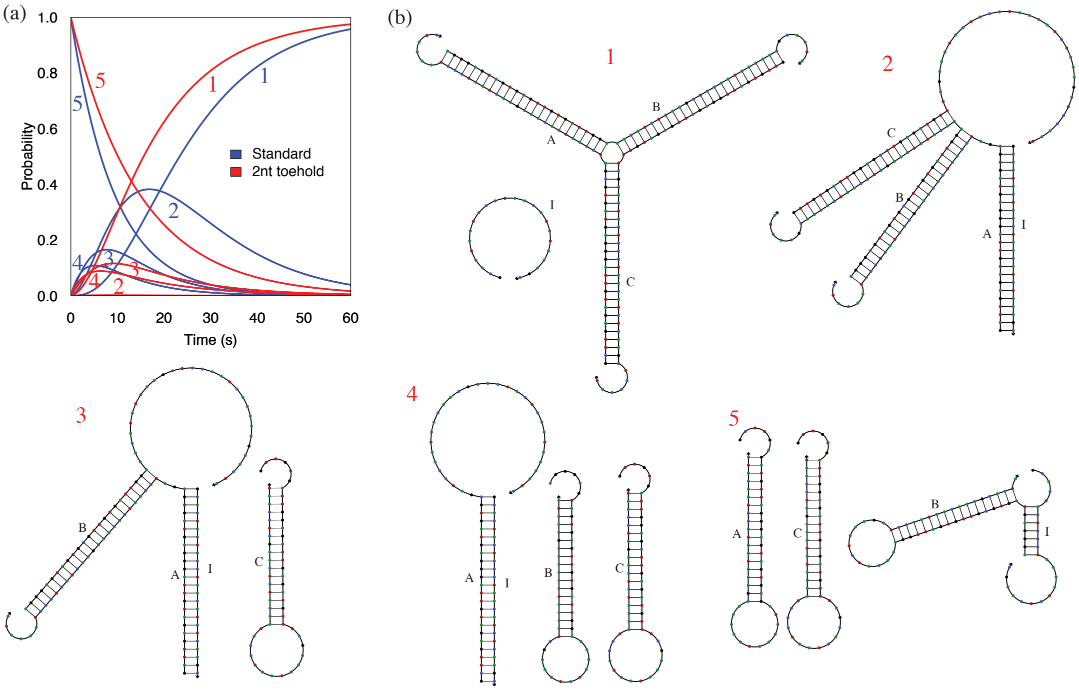


Figure 3.9: Alternative design of a catalytic three-arm junction. (a) Comparison of macrostate solutions for the alternative (red) system and the system of Figure 3.8. (b) Representative secondary structures for each macrostate. All inputs to the method are as in Figure 3.8.

the macrostate solution (a) shows. The macrostates identified are analogous to those of the unmodified system, though there is an off-pathway interaction between I and B in macrostate 5.

These two examples suggest the power of the trajectory-based method in rapidly (compared to doing experiments) evaluating systems of sequences for their kinetic properties, and, in particular, identifying off-pathway traps and rate-limiting steps.

Chapter 4

Model problem

To better understand how, and under what conditions the trajectory-based method works, we develop a model problem for which we can explicitly control all of the relevant parameters. The model we propose is the random walk on the d-cube, $\{0,1\}^d$. To create two macrostates, we consider two copies of the cube, represented by $\{0,1\}^d$ and $\{0,-1\}^d$ (by construction, we do not allow the all-zeros corners to coincide). We introduce a bias, $\beta \ll 1$, towards vertices with more 1s. The equilibrium distribution within each cube is

$$\pi(s) \propto \beta^{(\#0'\text{s in } s)}. \tag{4.1}$$

This model has a combinatorial flavor similar to the nucleic acid system, but all eigenvalues and transition rates can be written down explicitly. The mixing time within each macrostate is dictated by the dimension, d, and the bias, β. The transition rate between macrostates is a parameter, α.

A related model is the Random Energy Model (REM) [12, 13], which exhibits interesting metastable behavior without the introduction of a second cube. In the REM the energy of each vertex is an independent Gaussian random variable, and the equilibrium measure on the cube is

$$\pi_{\gamma,d}(s) = \frac{e^{-\gamma\sqrt{d}E_s}}{Z_{\gamma,d}}$$

where γ is the inverse temperature and Z is the partition function. As in our model, moves are allowed along the edges of the cube. The metastability arises, for large γ, because the system tends to spend most of its time in the states of lowest energy and little time jumping from one to another. To capture this behavior, Monthus and Bouchaud [39] developed a trap model where the dynamics on the cube is modeled by M traps between which the system jumps. Recent work has focused on establishing a rigorous connection between the trap model and the underlying walk on the cube [6].

For our purposes, our two-cube model has the advantage that all mixing and transition times can be explicitly calculated. The one-dimensional chain in continuous time has generator

$$K^{(1)} = \begin{pmatrix} -1 & \beta \\ 1 & -\beta \end{pmatrix} \tag{4.2}$$

and, thus, eigenvalues, $\lambda = 0, -(1+\beta)$. Thus, the d-dimensional cube has second-largest eigenvalue, $\lambda_1 = -(1+\beta)$. It is more convenient to deal with a discrete-time chain, so we uniformize the chain, and, to ensure aperiodicity, increase the holding probability in each state by $1/(d+1)$ at each step. Thus, we define the generator

$$A^{(d)} = \frac{1}{d+1} K^{(d)} + I. \tag{4.3}$$

The second-largest eigenvalue of $A^{(d)}$ is

$$\nu_2 = \frac{1}{d+1}(d-\beta) + 1 = \frac{d-\beta}{d+1}. \tag{4.4}$$

Our goal is to understand both analytically and numerically for which regions in parameter space the trajectory-based method identifies the correct macrostate dynamics, and what characterizes the regions where it fails. To begin we apply the trajectory-based method to the model problem for a range of parameters and construct maps of the parameter regions for which each step of the method succeeds. Figure 4.1 shows contour plots giving the probability of success in identifying all transitions, finding the correct clusters, or computing the correct transition rate for the model problem over a wide range of τ and α; in this case $d = 10$ and $\beta = 0.05$. (Unless otherwise noted, we use $\beta = 0.05$, which corresponds to an energy difference between levels of approximately $3k_BT$, and the transition threshold $\gamma = 0.95$.) Notice that the success regions for clustering and computing the transition rate are larger than that for transition identification, suggesting that later steps in the trajectory-based method are robust in that they can recover from improper identification or clustering earlier in the algorithm. This robustness of later steps to errors in earlier steps

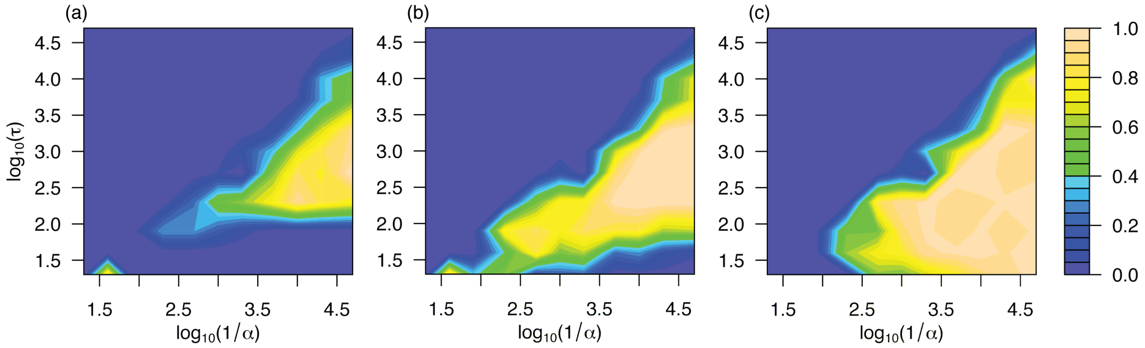


Figure 4.1: Numerical success probabilities for transition identification (a), cluster identification (b), and rate calculation (c) for the model problem with $d = 10$ and $\beta = 0.05$.

is important in applications where we may not have the control over parameters that we do here. If we simulate long enough to observe many macrostate transitions, then even if some are incorrectly identified, the clustering procedure in averaging over non-transition regions can mitigate the effect of the incorrect transitions.

4.1 Computational and analytic results

To understand the success regions in Figure 4.1, we will address errors encountered in each step.

4.1.1 Transition identification

Identifying macrostate transitions in a simulation is the cornerstone of the method and its biggest contribution. There appear to be two situations in Figure 4.1 (a) where the method fails. One is below a particular value of τ, regardless of α, and the other is when $\tau > 1/\alpha$. We will see that these regions correspond to Type I (false positive) and Type II (false negative) errors, respectively, and with some assumptions we can calculate bounds on both. With these bounds we can provide rigorous statements about the regions in parameter space for which the method will work with confidence.

4.1.1.1 Type I errors

A Type I error occurs when the method identifies a macrostate transition when there has been none, that is, when the distance in variation between the occupancy measure over two windows $[t-\tau, t]$ and $[t, t+\tau]$ is larger than γ even though there is no macrostate transition near t.

Before presenting a Type I error bound we need to introduce some notation. Denote the vertex with d 1s by $\underline{1}^{(d)}$; denote the partition function within one macrostate by Z; and denote the partition function restricted to states other than $\underline{1}^{(d)}$ by Z'. Thus, the partition functions are

$$Z = \sum_{i=0}^{d} \binom{d}{i} \beta^i \quad \text{and} \quad Z' = \sum_{i=1}^{d} \binom{d}{i} \beta^i. \tag{4.5}$$

Then define,

$$\mu = \frac{Z'}{Z}, \quad \bar{\mu} = 1 - \mu, \quad \epsilon = \gamma - \mu. \tag{4.6}$$

Recall that γ is the threshold for d_V above which we say there is a transition. The quantity μ is the equilibrium measure of all states except $\underline{1}^{(d)}$, which is small since $\beta \ll 1$. The following proposition gives an upper bound on the probability of a Type I error as a function of the parameters d, β, γ, and T.

Proposition 1 (Type I Error). *The probability that the distance in variation is greater than the transition threshold γ at some time in a simulation of length T with no macrostate transitions is*

$$\mathbf{P}[\textit{Type I}] \leq 1 - \exp\left\{ -T \frac{M}{C} \right\}, \tag{4.7}$$

where

$$M = \left[\frac{\mu + \bar{\mu}\nu_2}{1 - 2(\bar{\mu} - \epsilon)/(1 + \sqrt{\Delta})} \right]^{\tau(\mu+\epsilon)} \left[\frac{\bar{\mu} + \mu\nu_2}{1 - 2(\mu + \epsilon)/(1 + \sqrt{\Delta})} \right]^{\tau(\bar{\mu}-\epsilon)}, \tag{4.8}$$

$$\Delta = 1 + \frac{4\nu_2(\mu + \epsilon)(\bar{\mu} - \epsilon)}{\mu\bar{\mu}(1 - \nu_2)^2}, \tag{4.9}$$

and

$$C = \left(\frac{\gamma d\beta}{(d+1)Z'} - \frac{(1-\gamma)d\beta}{d+1} \right)^{-1}. \tag{4.10}$$

This error estimate comes from the Poisson clumping heuristic [4] whose main insight is that the expected hitting time of a Markov Chain on a set A of very small measure is approximately exponentially distributed with mean

$$\mathbf{E}[T_A] \approx C/\zeta(A), \tag{4.11}$$

where $\zeta(A)$ is the measure of A and C is the "clump size," a measure of how closely grouped the elements of A are. We will see that a Type I error can be characterized by a set in the space $[0,1]^{2^d}$, and thus, the time we must simulate to see a Type I error is exponentially distributed.

For any Markov chain, X_t, the tuples $(X_{t-\tau}, X_{t-\tau+1}, \ldots, X_t)$ are also a Markov chain, and the event $d_V(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]})) > \gamma$ represents a set in the space of τ-tuples. Thus, it makes sense to frame the problem of estimating the Type I error probability in terms of calculating an expected hitting time for a Markov Chain. To apply this heuristic we need to calculate C and $\zeta(A)$.

Measure of A: To estimate $\zeta(A)$, first note the following reformulation of the distance in variation: Let v and w be probability measures on a state space E. Then [7],

$$d_V(v,w) = 1 - \sum_{i \in E} \min(v(i), w(i)) \leq 1 - \min(v(i^*), w(i^*)), \forall i^* \in E. \tag{4.12}$$

That is, we can bound the distance in variation by monitoring only the empirical measure of a single state. Since the equilibrium measure on the d-cube is heavily concentrated at the vertex $\underline{1}^{(d)}$, we can choose $i^* = \underline{1}^{(d)}$ to get the best possible upper bound on d_V using (4.12). Using the bound, the set A is

$$A = \{X_{[t,t+\tau]} | \mu(X_{[t,t+\tau]})(\underline{1}^{(d)}) < 1 - \gamma\}. \tag{4.13}$$

We cannot compute $\zeta(A)$ explicitly, but we can bound it from above via a Hoeffding bound for Markov chains [32]. Given some function, $f : E \to \mathbb{R}$, this bound controls the probability $\mathbf{P}[S_n \geq n(\mu + \epsilon)]$, where $S_n = \sum_{t=1}^{n} f(X_t)$, $\mu = \mathbf{E}[f]$, and ϵ is a given deviation. In this case, $f(X_t) = \mathbb{1}[X_i = \underline{1}^{(d)}]$. Then (4.6) and (4.4) directly give (4.8) and (4.9).

Cluster size: To compute the clump size, C, we approximate the process around the set A by a symmetric random walk on the integers (see Aldous [4], Sections B2 and B10). In this case the integers represent the number of instances of $\underline{1}^{(d)}$ and all other states in the time slice τ. Then the clump size is,

$$C = (P_{\text{more } \underline{1}^{(d)}} - P_{\text{less } \underline{1}^{(d)}})^{-1} \tag{4.14}$$

To compute these transition probabilities, assume that the visits to $\underline{1}^{(d)}$ are uniformly distributed throughout the interval so that the probability that the first structure is $\underline{1}^{(d)}$ and the last structure is $\underline{1}^{(d)}$ are both 0.05. We calculate $P_{\text{more } \underline{1}^{(d)}}$ and $P_{\text{less } \underline{1}^{(d)}}$ by conditioning on the first and last states in the sequence. For ease of notation, label $\underline{1}^{(d)}$ as 1 and group all other states under label 0. Thus, the probabilities are:

$$\begin{aligned} P_{11} &= \frac{1 + d(1-\beta)}{d+1}, && (4.15) \\ P_{10} &= \frac{d\beta}{d+1}, && (4.16) \\ P_{01} &= \frac{d\beta}{(d+1)Z'}, && (4.17) \\ P_{00} &= \left(1 - \frac{d\beta}{(d+1)Z'}\right). && (4.18) \end{aligned}$$

Then,

$$\begin{aligned} P_{\text{more } \underline{1}^{(d)}} &= \mathbf{P}[\text{first is } 0]\left(\mathbf{P}[\text{last is } 0]P_{01} + \mathbf{P}[\text{last is } 1]P_{11}\right) && (4.19) \\ P_{\text{less } \underline{1}^{(d)}} &= \mathbf{P}[\text{first is } 1]\left(\mathbf{P}[\text{last is } 0]P_{00} + \mathbf{P}[\text{last is } 1]P_{10}\right). && (4.20) \end{aligned}$$

Substituting (4.15–4.18) into (4.19 and 4.20) and then into (4.14) gives (4.10).

Figure 4.2 compares this estimate with simulated success probabilities. To compare the bound over a range of d we interpolated to find the τ required for 50% success probability. For small dimensions the τ required for 50% success is estimated to be too large by a factor of about 1.5; this factor grows with d, but nevertheless it does provide a lower bound for choosing τ. A likely source of the error as d grows is the upper bound (4.12), which becomes less tight since, for fixed β, the equilibrium measure of $\underline{1}^{(d)}$ decreases as d increases. In addition, the Hoeffding bound, being a general bound, is unlikely to be tight, and the Poisson clumping heuristic is, after all, a heuristic. The mixing time within a cube also increases with d (this is seen in a larger ν_2), and we see this effect in the numerical simulations, but it is difficult to tease apart the effects of the approximation (4.12) and the increase in mixing time due to a larger ν_2.

Since the approximation (4.12) prevents us from characterizing the contributions from d and β to the mixing time, the question arises whether it is possible to estimate Type I error probabilities without this approximation. The problem is that to use the Hoeffding bound

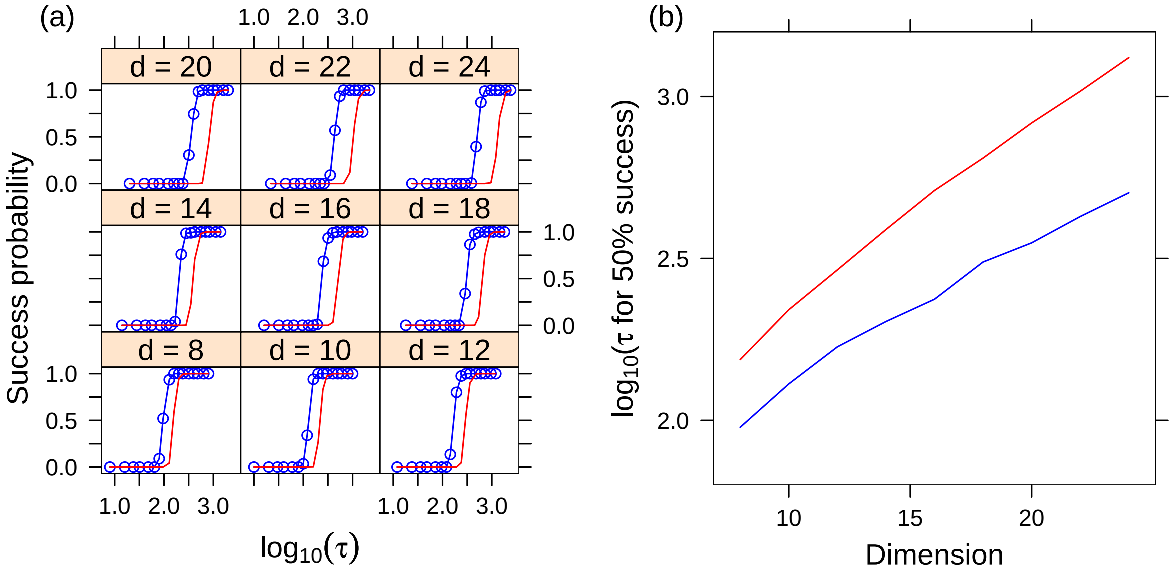


Figure 4.2: Probability that no Type I errors occur (a), and interpolated value of τ for 50% success probability (b) from simulation (blue) and calculation (red).

[32], we must have a scalar function of the trajectory X, and keeping track of multiple components simultaneously requires a vector function. A large deviation principle for the vector of empirical measures of X exists [11], but the estimate is asymptotic and does not apply for short τ. To our knowledge there are no similar results that apply for short times, as the Hoeffding bound does.

4.1.1.2 Type II errors

A Type II error occurs when the distance in variation between the occupancy measure over the two windows $[t-\tau, t]$ and $[t, t+\tau]$ is lower than the threshold even though a transition has taken place at t. The following proposition gives an estimate on such an error.

Proposition 2 (Type II Error)**.** *In a simulation of time T, with τ chosen to avoid Type I errors, the probability of a Type II error is*

$$\mathbf{P}[\mathit{Type\ II}] \leq 1 - \sum_{i=0}^{\lfloor \frac{T}{\tau} \rfloor + 1} e^{-\alpha T} \frac{\alpha^k (T - \tau(k-1))^k}{k!}. \tag{4.21}$$

Assuming that τ is chosen so that a Type I error is very unlikely, then a Type II error will occur when two transitions occur within time less than τ. In that case, one of $X_{[t-\tau,t]}$

and $X_{[t,t+\tau]}$ will be an average over both macro states, so $d_V(\mu(X_{[t-\tau,t]}), \mu(X_{[t,t+\tau]})) < 1$. The transitions must be strictly closer than τ in order to have $d_V < \gamma$, but by considering transitions closer than or equal to τ, we get an upper bound. To compute the estimate, condition on k, the number of transitions in time T. The number of transitions is a Poisson random variable and the times between transitions are exponentially distributed with parameter α. The distribution of inter-transition times conditioned on them being longer than τ is also exponential with rate α. Thus, the probability of k transitions with no two closer than τ is the product of the probability of k transitions in time $T - \tau(k-1)$ and the probability of $k-1$ waiting times of length τ, that is,

$$\begin{aligned}\mathbf{P}[k \text{ transitions, none within } \tau] &= e^{-\alpha(T-\tau(k-1))}\frac{\alpha^k(T-\tau(k-1))}{k!}e^{-\alpha\tau(k-1)} \\ &= e^{-\alpha T}\frac{\alpha^k(T-\tau(k-1))^k}{k!}. \qquad (4.22)\end{aligned}$$

Sum this over the possible number of transitions k and take the difference from unity to get (4.21). Figure 4.3 compares this estimate with the actual success probability from simulations. We expect the Type II error to be independent of d, and Figure 4.3(b) compares the total range over d of calculated values to the estimate (4.21). The differences are small, suggesting that this is indeed the mechanism by which Type II errors occur.

4.1.2 Clusters and partition functions

The complexity of the clustering step means that there are too many influencing factors to permit a detailed analysis of the success of that step, but we can examine the error in calculating the partition function.

To compute the partition function we run a short simulation to generate a set of visited structures, A. Then we run additional simulations to determine how much of the macrostate the set A represents, that is, we estimate π_A. Figure 4.4 shows the results from one run each at different dimensions. The estimate of π_A is very close to the actual value, and indeed the confidence intervals cover the true value in all instances. We can see that as the macrostate gets larger (d grows), π_A decreases and the confidence intervals become correspondingly less tight. In addition, as τ grows, π_A increases – we visit more of the macrostate – and the confidence interval gets tighter.

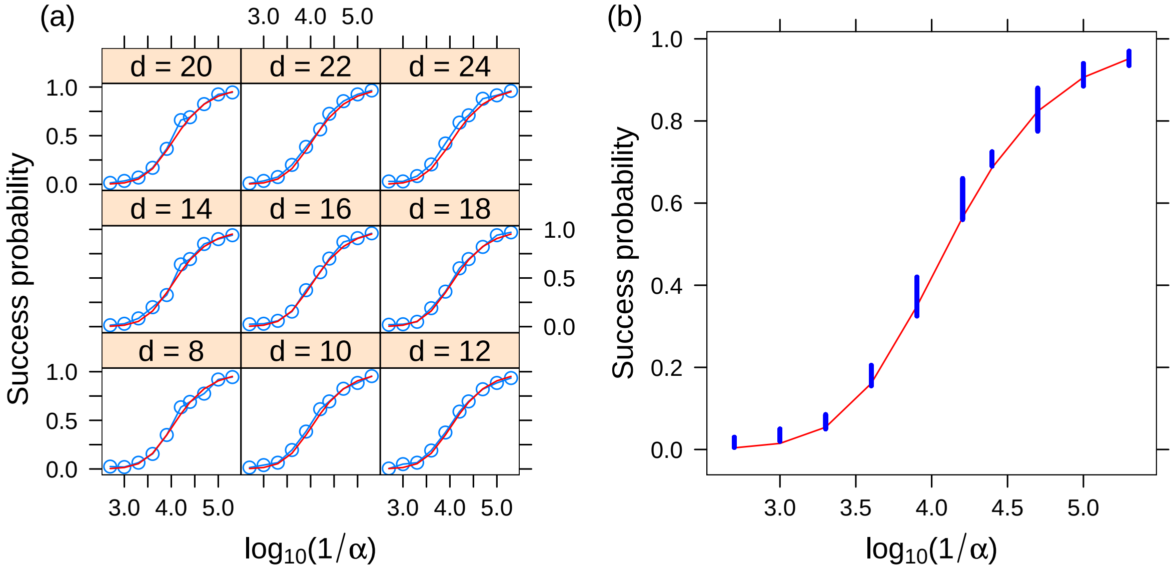


Figure 4.3: Probability that all transitions are identified for a range of d-values (a); probabilities were averaged over 200 trials with τ fixed at 1000. Calculated error probability (b); the error bars show the total range of simulated probabilities over all d. Blue shows simulated values and red computed.

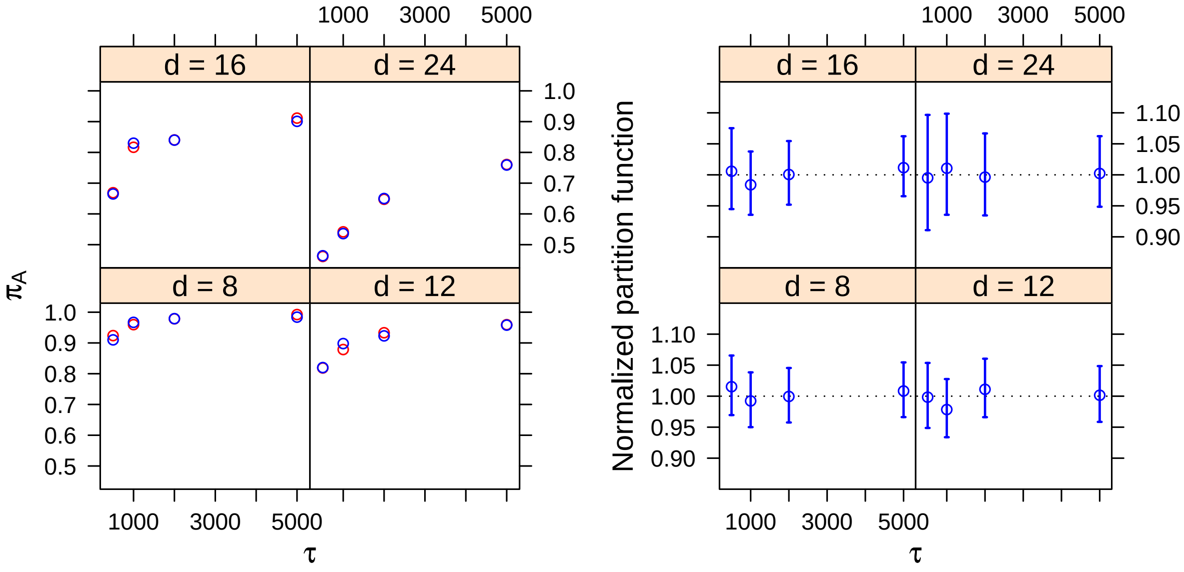


Figure 4.4: Estimated partition functions. Left panel shows estimated (blue) and true (red) values of π_A, the equilibrium fraction of states visited in the first simulation of length τ. Estimates of π_A are from 1000 additional simulations of length τ. Right panel shows normalized partition function estimates and confidence intervals.

4.1.3 Rate constants

Macrostate transition rates are computed based on first-passage times between macrostates; however, we must simulate passage times between pairs of microstates. Here we address the bias this introduces in our estimate. The macrostate transition probability is uniform in the starting macrostate, so the choice of starting state introduces no bias. After a macrostate transition, a new microstate is chosen according to the equilibrium distribution within the new macrostate. Thus, the expected first passage time consists of the time for a macrostate transition to occur plus the time to reach the representative state in the second macrostate. Note that the first passage time from a particular microstate to a representative state depends only on the number of 1s in the state. Thus we have

$$\mathbf{E}[T_{\text{fp}}] = \mathbf{E}[T_{\text{macro}}] + \mathbf{E}[T_{\text{repr}}] = \frac{1}{\alpha} + \frac{1}{Z}\sum_{i=0}^{d} f_i^{\text{repr}} \binom{d}{i} \beta^{(d-i)}. \tag{4.23}$$

That is, $\mathbf{E}[T_{\text{repr}}]$ is the sum over d of the probability at equilibrium of choosing a state with i 1s multiplied by the first passage time from a state with i 1s to the representative state.

The representative structure is the ensemble average of the number of 1s in the state, which can be calculated as

$$s_{\text{repr}} = \frac{1}{Z}\sum_{i=1}^{d} i \binom{d}{i} \beta^{(d-i)}. \tag{4.24}$$

We round s_{repr} to the nearest integer since all microstates have an integral number of 1s. To compute the first-passage times to s_{repr} from each state in the macrostate, we have the recurrence

$$f_i^{\text{repr}} = 1 + \frac{1+(1-\beta)i}{d+1} f_i^{\text{repr}} + \frac{\beta i}{d+1} f_{i-1}^{\text{repr}} + \frac{d-i}{d+1} f_{i+1}^{\text{repr}}, \tag{4.25}$$

with the boundary condition $f_{\text{repr}}^{\text{repr}} = 0$. This leads to a linear system of size d that is easily solved. If $d = 12$ and $\beta = 0.05$ then $\mathbf{E}[T_{\text{repr}}] = 13.0$, which is small compared to the mean transition times of interest (which range from 100 to 10,000 in Figure 4.1, for example).

Though we can calculate $\mathbf{E}[T_{\text{repr}}]$ exactly, a simple bound also shows that $\mathbf{E}[T_{\text{repr}}]$ will be small if the local mixing time is faster than the mean time for macrostate transitions. Aldous and Fill [5, Ch. 3, p. 24] give a bound on the mean hitting time of a state that depends only on the mixing time of the chain and the equilibrium measure of the state. Denote the equilibrium probability of the representative structure by π_{repr}. Then the expected hitting time on s^{repr} obeys

$$\mathbf{E}[T_{\mathrm{repr}}] \leq \frac{1-\pi_{\mathrm{repr}}}{(1-\nu_2)\pi_{\mathrm{repr}}}. \tag{4.26}$$

Since the chain is biased towards $\underline{1}^{(d)}$, π_{repr} is not small, and thus the bias due to first passage sampling is also small compared to the macrostate transition time. This second approach applies to the nucleic acid case as well, showing that in the typical case where the centroid structure is at or near a local minimum, the bias due to first passage time sampling between secondary structures should be small as long as there is a separation of timescales.

4.2 Hierarchical macrostates

We conclude with an example that shows how the choice of τ influences which macrostates the method finds. We ran the method on the hyper-cube with $d = 12$ for $\tau = 1000$ and 30000. Here the method was run with four macrostates, $A = \{0,1\}^d$, $B = \{0,i\}^d$, $C = \{0,-1\}^d$, and $D = \{0,-i\}^d$, with fast transitions between the pairs (A,B) and (C,D) with $1/\alpha_1 = 10000$ and slow transitions between the pairs (A,D) and (B,C) with $1/\alpha_2 = 316000$ (1.5 orders of magnitude higher). Figure 4.5 shows the results. The blue dots correspond to representative structures found when $\tau = 1000$ and the red dots correspond to representative structures found with $\tau = 30000$. As τ becomes longer than the mean time for faster transitions we see pairs of macrostates merge. This suggests that in some situations τ acts as a "coarseness" knob and allows us to extract information about macrostates at several timescales. Of course, for many systems there may only be one timescale at which multiple macrostates are active. In terms of the eigenvalue analysis of Deuflhard et al. [14] and others, if the system contains several gaps in the eigenvalues, we can tune τ to fall within the gap of our choosing.

4.3 Outlook and extensions

Analysis of this relatively simple model problem has allowed us to construct a nearly complete picture of the error behavior of our method. We found that computing macrostate transition rates via microscopic simulations introduces little bias (Section 4.1.3). Our method for computing partition functions gives accurate estimates and tight error bounds for a range of problem sizes and values of τ (Section 4.1.2). The error estimates for the

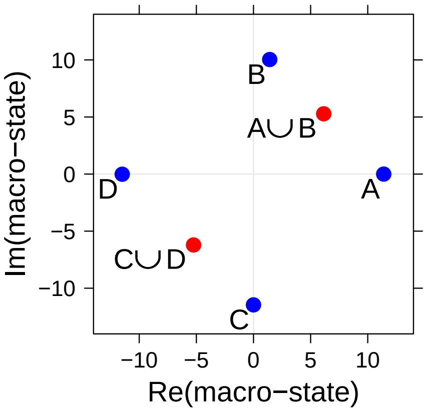


Figure 4.5: The method run on a hypercube with $d = 12$ and four macrostates. $\tau = 1000$ (blue) and $\tau = 30000$ (red). The dots show representative structures for the macrostates found by the method. For $\tau = 30000$ macrostates A and B merge, as do C and D.

identification step (Section 4.1.1) show that the averaging time, τ, must occupy a middle ground, being longer than the mixing time within a macrostate, but shorter than the mean time between transitions. Luckily, for a range of interesting nucleic acid problems, such a τ exists (Section 3.2).

One of the main innovations of the algorithm is the ability to explicitly locate transitions in the simulation by computing the distance in variation between state occupancy distributions along sliding windows in the trajectory. The results for the hyper-cube model problem suggest that this method might apply for many discrete systems that have a separation of time-scales. Applying it to systems in continuous space would require developing an appropriate metric with which to calculate the distance in variation, perhaps discretizing the state space or deriving a metric from diffusion maps [8]. The next chapter describes in detail the method by which we obtain representative structures for each macrostate.

Chapter 5

Centroids in Ensembles of Nucleic Acid Structures

In this chapter we turn to the problem of choosing the best structure to characterize an ensemble of secondary structures. The default choice is the *minimum free energy* (MFE) structure, which has the lowest free energy and highest probability at equilibrium. That is,

$$s^{\mathrm{MFE}} = \operatorname*{argmax}_{s\in\Omega} \pi(s) = \operatorname*{argmin}_{s\in\Omega} \Delta G(s).$$

A more complete picture of the equilibrium ensemble is gained by considering the pair probability matrix, $P(\pi)$, which shows the probability of each base pair in the ensemble. This $N \times (N+1)$ matrix has entries $P_{i,j} \in [0,1]$ which represent, for $1 \leq j \leq N$, the probability that bases i and j are paired and, for $j = N+1$, the probability that base i is unpaired. Though providing more information about the ensemble, the pair probability matrix is not as easy to interpret as a single secondary structure. Since the pair probability matrix contains complete information about the ensemble, the structure that is, by an appropriate measure, closest to the pair probability matrix should best represent the ensemble.

For a single secondary structure, s, we construct an $N \times (N+1)$ structure matrix $S(s)$. The entries $S_{i,j} \in \{0,1\}$ are unity when bases i and j are paired, and $S_{i,N+1} = 1$ if base i is unpaired; all other entries are zero. The average nucleotide distance between s and all structures in the ensemble is [20]

$$
\begin{aligned}
n(s) &= \sum_{\sigma \in \Omega} \pi(\sigma) \| S(s) - S(\sigma) \|_1 \\
&= \| S(s) - \sum_{\sigma \in \Omega} \pi(\sigma) S(\sigma) \|_1 \\
&= \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N+1} | S(s)_{i,j} - P_{i,j} | \\
&= N - \sum_{i=1}^{N} \sum_{j=1}^{N+1} P_{i,j} S_{i,j}(s), \qquad (5.1)
\end{aligned}
$$

where the last step holds because S is composed of zeros and ones.

Thus, the structure that minimizes $n(s)$ is the structure that is closest to the pair probability matrix and closest, on average, to all structures in the ensemble. Put otherwise, the structure that satisfies

$$
s^{\text{cent}} = \operatorname*{argmin}_{s \in \Omega} n(s), \qquad (5.2)
$$

is the centroid of the ensemble with respect to the nucleotide distance.

Though we have presented the centroid structure in the context of the equilibrium ensemble, it can be computed for any ensemble once we have the derived pair probability matrix. For example, in Chapter 3 we computed centroids from pair probability matrices generated by kinetic simulations. The fact that all we need is the pair probability matrix offers a significant memory savings compared to storing each structure visited in the simulation. Section 5.2 describes how to calculate exactly and efficiently the centroid for a single nucleic acid strand and a complex of interacting strands. First, we will compare the centroid and MFE structures in their ability to effectively characterize the ensemble and their robustness to uncertainty in the energy parameters. Ding et al. [16] present a centroid structure computed using a different metric via sampling and examination of the pair probability matrix. Section 5.1.3 compares this approach with ours.

5.1 Comparing representative structures

In this section we will compare the MFE and centroid structure as a representative of the equilibrium ensemble of secondary structures by evaluating $n(s^{\text{MFE}})$ and $n(s^{\text{cent}})$. Though we will be considering averages over many sequences, we note that for some sequences, the MFE and centroid can be quite different. Figure 5.1 shows such a sequence. The pair probability matrices have the base pairs making up the MFE (top) and centroid (bottom) structures circled. The base pairs that make up the centroid structure are of higher probability than those that make up the MFE, and thus better represent the pair probability matrix.

5.1.1 How the MFE and centroid characterize the ensemble

In this section we compare how well the MFE and centroid structures characterize the ensemble as measured by $n(s)$. We consider two sets of 300 sequences 200 nucleotides long. The first set is comprised of random sequences and the second of sequences that were designed to assume a particular target secondary structure by optimizing $n(s^{\mathrm{target}})$ [20]. We compared $n(s^{\mathrm{MFE}})$, $n(s^{\mathrm{cent}})$, the improvement of the centroid over the MFE, the degeneracy of the MFE, and the distance between the MFE and centroid structures. Histograms for each measurement are shown in Figures 5.2 and 5.3.[1]

In comparing $n(s^{\mathrm{MFE}})$ and $n(s^{\mathrm{cent}})$ for designed sequences (Figure 5.2 (a,b)), we see similar distributions. Though the MFE has higher maximum values, the means are close. Indeed, for the vast majority of sequences, the centroid improves little over the MFE (c). In addition to having similar values of $n(s)$, the MFE and centroid are nearly always close to each other, suggesting that the ensemble is dominated by a single basin that is well characterized by a single structure.

The situation for random sequences is quite different. Both $n(s^{\mathrm{MFE}})$ and $n(s^{\mathrm{cent}})$ (Figure 5.3 (a,b)) are much higher—greater than 20% versus less than 5% for designed sequences. Still, the improvement by the centroid structure is small (c). Unlike for the designed sequences, a significant number of the MFE structures are degenerate (d). Most interesting is that although $n(s^{\mathrm{MFE}})$ and $n(s^{\mathrm{cent}})$ are comparable, the structures are themselves quite different, having on average more than 10% of bases paired differently and as much as 60% (e). These facts taken together argue that for random sequences, no single structure characterizes the ensemble well. Thus, for either class of sequences the choice of representative structure is unimportant—for designed sequences because both choices do well and

[1] When there are multiple MFE structures for a sequence, we use the structure that does best with respect to the current measure. Averaging over MFE structures does not significantly affect the results. In practice, there are never multiple centroid structures for a single sequence.

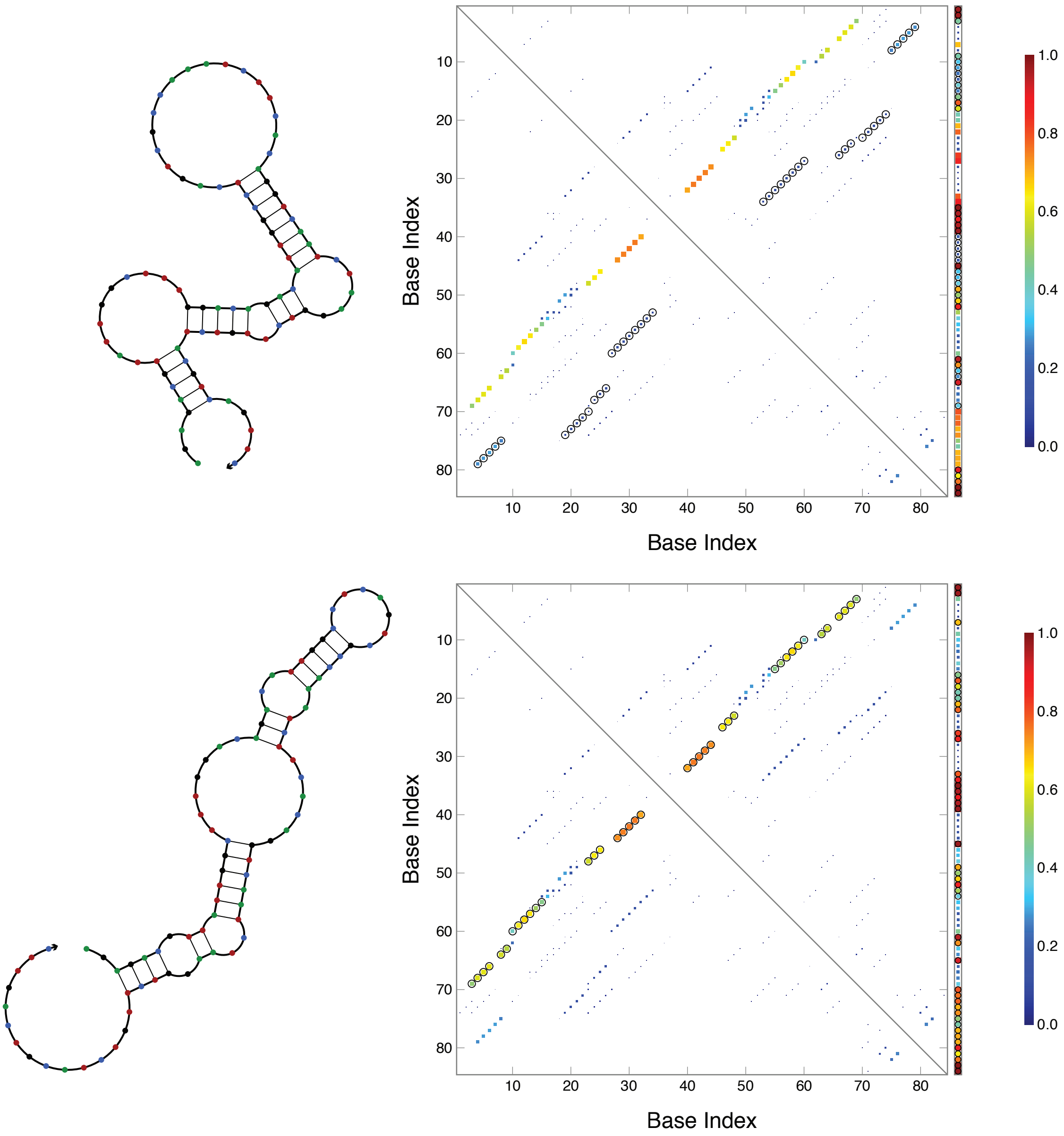


Figure 5.1: MFE (top) and centroid (bottom) structures for the sequence `AGAGACGUUAUUGGCUUUGGACAGACAUUGGCCUCAGUCGCCAAAUCUUCACAGGUCAAUCUAAGGUCUUGUCUACGUCAGUUC`. The color and size of the boxes in the pair probability matrix represent the probability of the base pairs in the ensemble; the column to the right shows the probability that the bases are unpaired. The pair probability matrices shown here are identical, apart from having the base pairs that make up the MFE (top) and centroid (bottom) structures circled. $n(s^{\mathrm{MFE}}) = 25.3$ and $n(s^{\mathrm{cent}}) = 24.4$; $\Delta G(s^{\mathrm{MFE}}) = -16.4$ kcal/mol and $\Delta G(s^{\mathrm{cent}}) = -14.2$ kcal/mol.

for random sequences because neither does well.

5.1.2 Robustness to uncertainty in the energy parameters

One benefit of the centroid might be that its dependence on the entire ensemble via the pair probability matrix makes it more robust to uncertainty in the energy model. To test this hypothesis we generated parameter sets where each parameter was independently perturbed by a Gaussian random variable with standard deviation 10, 20, 50, or 80 percent of the original parameter. At each level of perturbation, we generated 100 parameter files and computed the MFE and centroid structures for 300 designed and 300 random sequences each of lengths 100 and 200. Finally, for each class of sequence we computed the average over all sequences of the average distance between all perturbed MFE (or centroid) structures. If the centroid were more robust then the perturbed centroid structures would be more tightly clustered. Figure 5.4 shows these results. The top panel shows the average distances for each sequence class and perturbation level. The MFE structures appear to be less tightly clustered on average than the centroid structures. To test this hypothesis we performed a Wilcoxon signed rank test with null hypothesis that the median of

$$\mathcal{S} = \mathrm{avgdist}(s^{\mathrm{MFE}}) - \mathrm{avgdist}(s^{\mathrm{cent}})$$

is zero. This gives us 95% confidence intervals and an estimate for the median of the distribution $\mathcal{S}$, which are plotted in the lower pane. For all but the designed sequences at small perturbations the centroid structures are much more tightly clustered.

However, considering only $\mathcal{S}$ masks the fact that for large perturbations neither is very tightly clustered. The random results at low perturbation levels support our intuition regarding the difference between the MFE and centroid. Since the MFE optimization looks for the deepest minimum in the free energy landscape, small perturbations might shift the relative depth of minima that are quite distant from each other. Since the centroid considers information about all structures in the ensemble, it is more robust to small changes in the energies of structures in the ensemble. The results for designed sequences at small perturbations support the use of $n(s)$ as a design criteria and amplify the similar results in [20]. When a sequence is well designed with a single dominant basin, the MFE and centroid will be close, and small perturbations to the parameters do not change the location of the

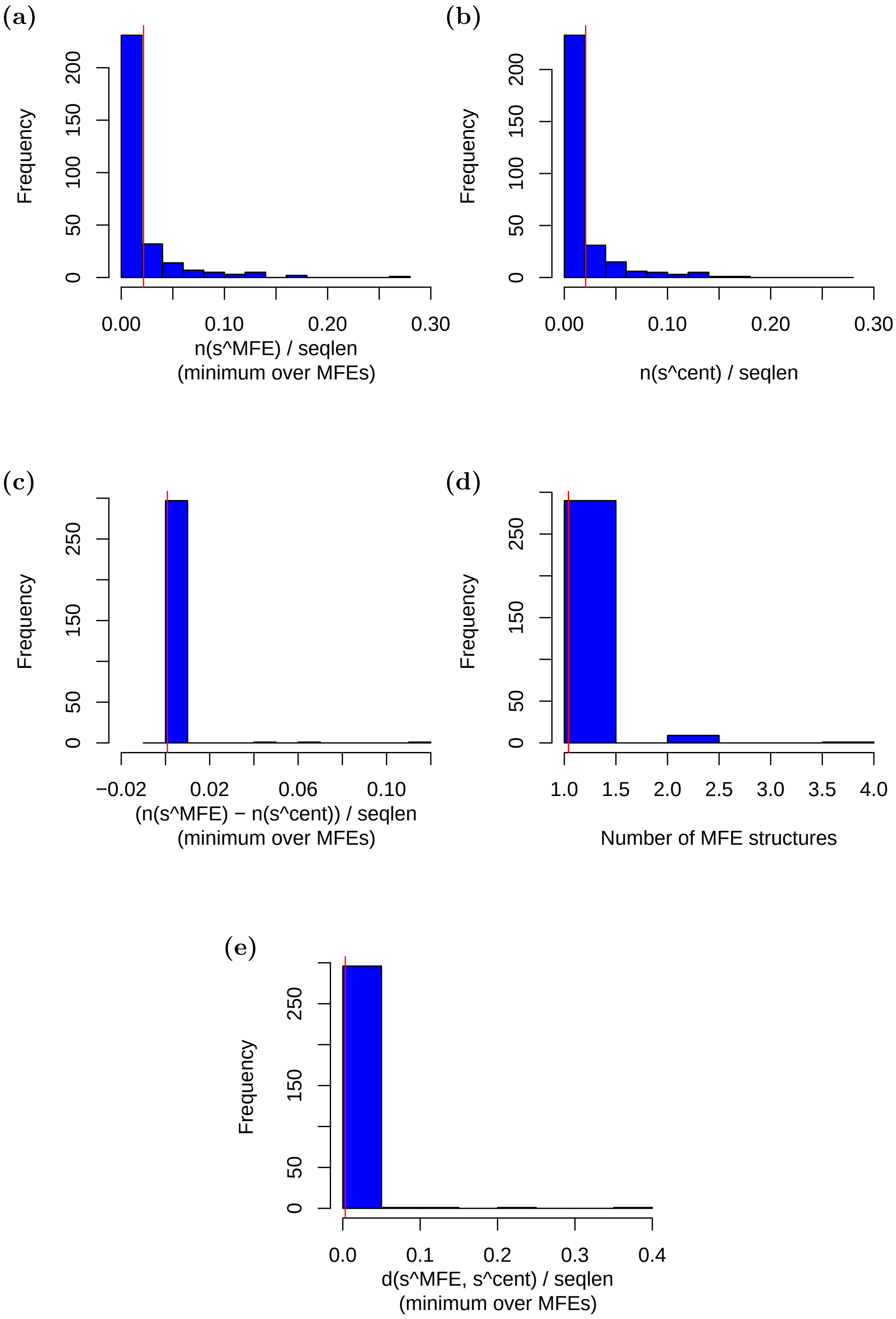


Figure 5.2: Comparison between the MFE and centroid structures for 300 designed sequences 200 nucleotides long. The red line indicates the mean.

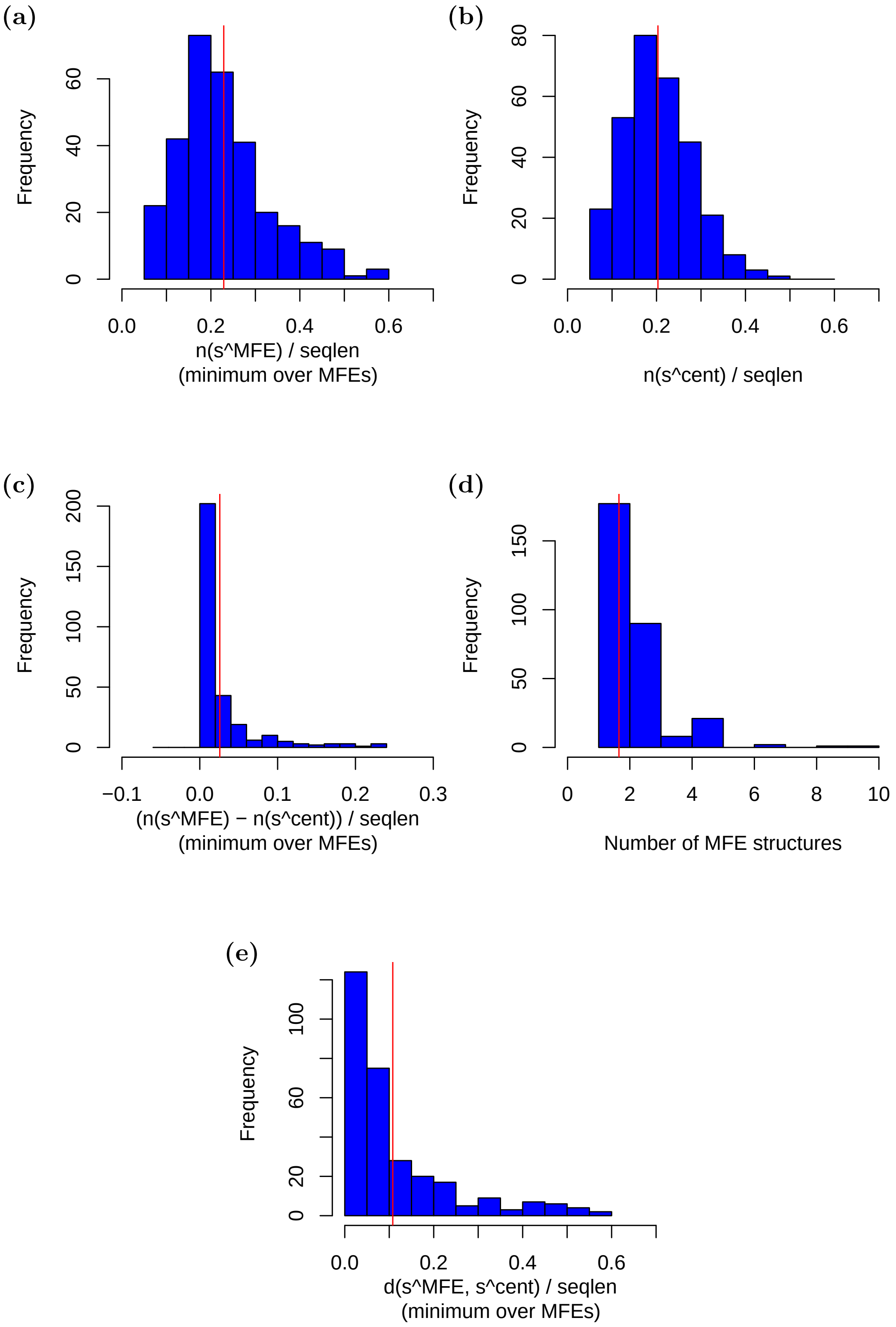


Figure 5.3: Comparison between the MFE and centroid structures for 300 random sequences 200 nucleotides long. The red line indicates the mean.

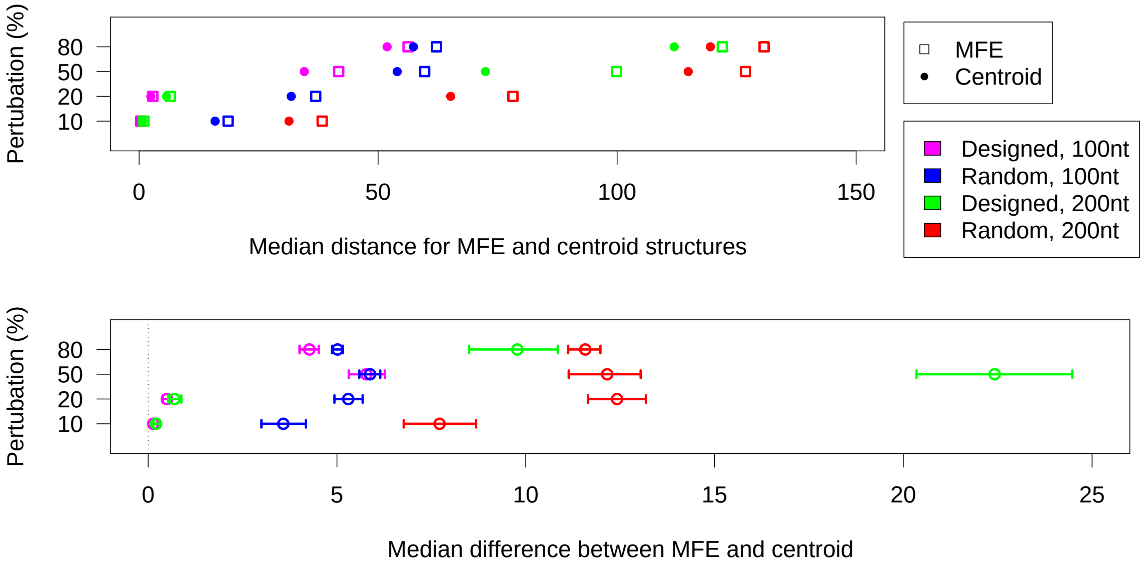


Figure 5.4: Average distance among perturbed MFE (centroid) structures with parameters perturbed by normal distribution with standard deviation shown on the ordinate (averaged over both structures and perturbations). Top pane shows average distances. Bottom shows estimated median and 95% confidence intervals for the distance between the MFE and centroid points in the top pane.

basin very much.

5.1.3 Comparing two centroids

A centroid is defined with respect to a particular metric and Ding et al. [16] present a centroid with respect to the base pair distance instead of the nucleotide distance. Rather than considering the entire ensemble, Ding draws a sample of structures from the equilibrium ensemble, looks for distinct clusters of structures, and computes the centroid for each of the clusters. For purposes of comparison, we compute the centroid with respect to the base-pair distance for the entire ensemble. We refer to base-pair distance centroid as the Ding centroid or s^{Ding}. While we use a dynamic program to find the centroid for a variety of ensembles (Section 5.2), Ding simply chooses all base-pairs in the pair probability matrix with entries strictly larger than 0.5. Since no pair with probability greater than 0.5 can have a competing pair of higher probability (the probabilities sum to 1 for each base) and a pair with probability exactly 0.5 can be omitted without increasing the average distance, this procedure is guaranteed to give the unique centroid structure with the minimum number of base-pairs. This approach cannot, however, be extended to a complex of interacting strands since the structure formed by only base pairs of probability greater than 50% may not be connected.

Fortunately, we can evaluate Ding's notion of closeness to the ensemble, the base-pair distance from a structure to the ensemble, without sampling structures. Evaluating (1.6) and noting as Ding did, that the base-pair distance is equivalent to the squared Euclidean distance between the structure matrices, we find that

$$\begin{aligned}
n_{BP}(s) &= \sum_{\sigma\in\Omega} p(\sigma)\|S(\sigma)-S(s)\|_{BP} \\
&= \frac{1}{2}\sum_{\sigma\in\Omega} p(\sigma)\sum_{i=1}^{N}\sum_{j=1}^{N}\left(S(\sigma)_{ij}-S(s)_{ij}\right)^2 \\
&= \frac{1}{2}\sum_{\sigma\in\Omega} p(\sigma)\sum_{i=1}^{N}\sum_{j=1}^{N}\left[S(\sigma)_{ij}^2-2S(\sigma)_{ij}S(s)_{ij}+S(s)_{ij}^2\right] \\
&= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}P_{ij}+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}S(s)_{ij}-\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{\sigma\in\Omega}p(\sigma)S(\sigma)_{ij}S(s)_{ij} \\
&= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}P_{ij}+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}S(s)_{ij}-\sum_{i=1}^{N}\sum_{j=1}^{N}S(s)_{ij}\sum_{\sigma\in\Omega}p(\sigma)S(\sigma)_{ij} \\
&= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}P_{ij}+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}S(s)_{ij}-\sum_{i=1}^{N}\sum_{j=1}^{N}S(s)_{ij}P_{ij}
\end{aligned}$$

where we have used the fact that $S_{ij}\in\{0,1\}$, so $S_{ij}^2=S_{ij}$. The interpretation is clear: The first term is the average number of base pairs in the ensemble; the second is the number of base pairs in the target structure; and the third is twice the average number of base pairs in common—twice because any pair in common is penalized once in each of the first two terms.

Figures 5.5 and 5.6 compare the two centroids for designed and random sequences. There is very little difference between the MFE, centroid, and Ding centroid for designed sequences, suggesting that they all reside in a single dominant basin in the free energy landscape. For random sequences (Figure 5.6), the Ding centroid is not much worse than the centroid at characterizing the ensemble by $n(s)$ (d). Even so, the two centroids can be quite different, greater than 5%, on average (b). This is closer than the MFE and Ding centroid (a) or the MFE and centroid (Figure 5.3 (e)). It is interesting that while the centroid nearly always improves on the MFE in characterizing the ensemble with respect to the base-pair distance, the Ding centroid is frequently worse than the MFE with respect to $n(s)$ (e). This is not surprising since $n(s)$ considers the whole pair probability matrix, including the unpaired bases, while the base-pair distance considers only paired bases.

5.2 Algorithms

In this section we present algorithms to find centroid structures for a single strand and an ordered complex of interacting strands. In computing a centroid structure for multiple strands additional difficulties arise if multiple strands of the same sequence are considered to be indistinguishable.

5.2.1 A single strand

Like finding the MFE structure and computing the partition function, the problem of finding the centroid structure is solved by dynamic programming. This is because the contribution from each base pair to the sum in (5.2) is additive. To make the approach clearer, we will first review the dynamic program that computes the minimum free energy structure for a given sequence. We use notation similar to that used to describe the partition function algorithm [21].[2] The main recursion is

$$F_{i,j} = \min \left\{ 0, \min_{i \le d < e \le j} \left\{ F_{i,d-1} + F^b_{d,e} \right\} \right\}. \tag{5.3}$$

The term $F_{i,j}$ refers to the minimum free energy structure on the subsequence from i to j. This structure has either no base pairs and thus zero energy, or it has some rightmost base pair between indices d and e. Once the base pair $d \cdot e$ is fixed, we search for the minimum free energy structure on the independent segments $(i, d-1)$ and (d, e). The MFE of the structure on $i, d-1$ is stored in $F_{i,d-1}$, while $F^b_{d,e}$ stores the MFE of the structure on the

[2] In particular, the algorithm to compute the MFE can be obtained from the partition function algorithm by replacing the Boltzmann factors, $e^{-\Delta G/k_B T}$, by free energies, ΔG, adding instead of multiplying terms for independent subsequences, and minimizing instead of summing over alternate structures. The reverse is not true since a MFE algorithm may have redundancies, but a partition function algorithm must recurse over each structure exactly once.

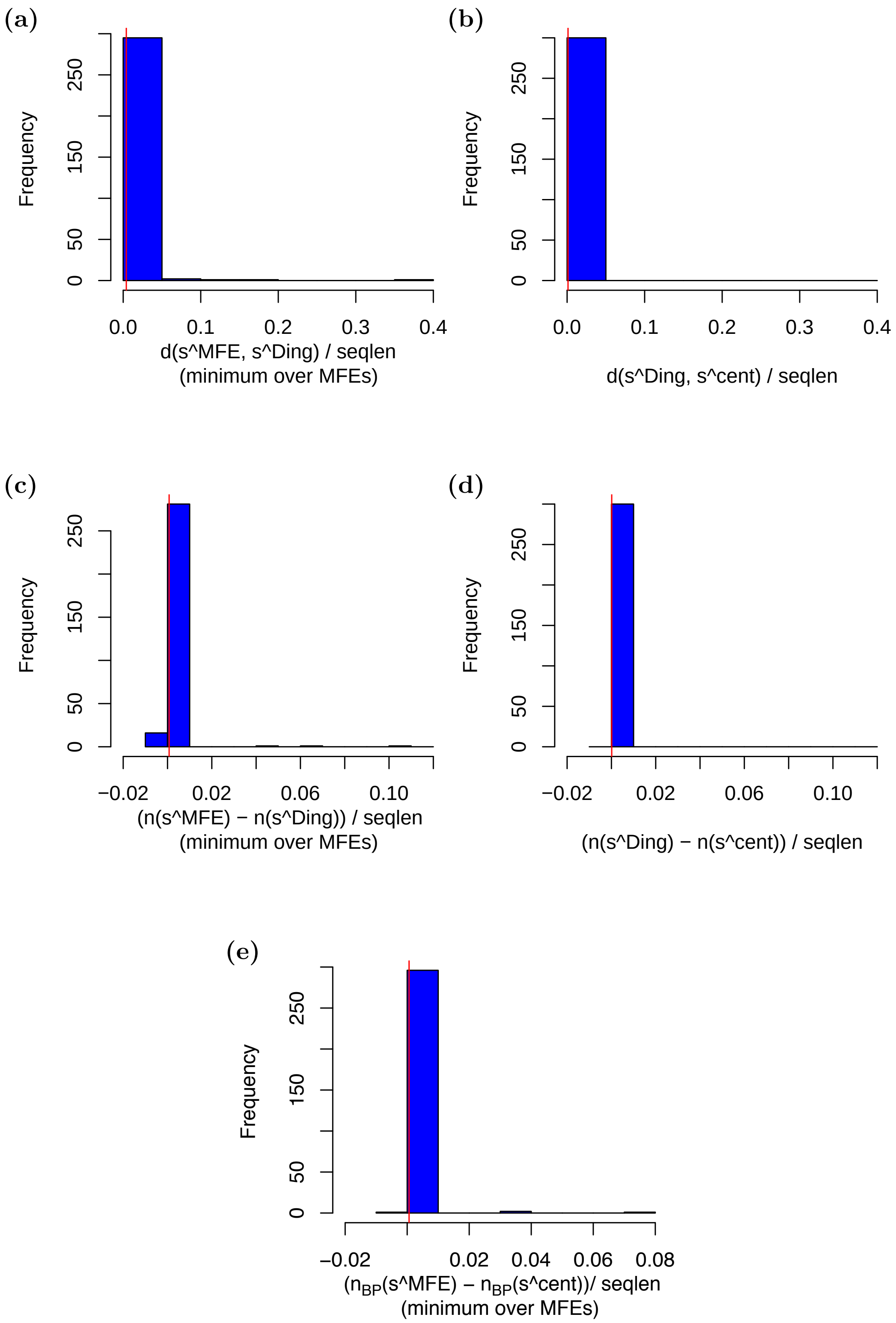


Figure 5.5: Comparing the centroid with the Ding centroid for 300 designed sequences 200 nucleotides long. The red line indicates the mean.

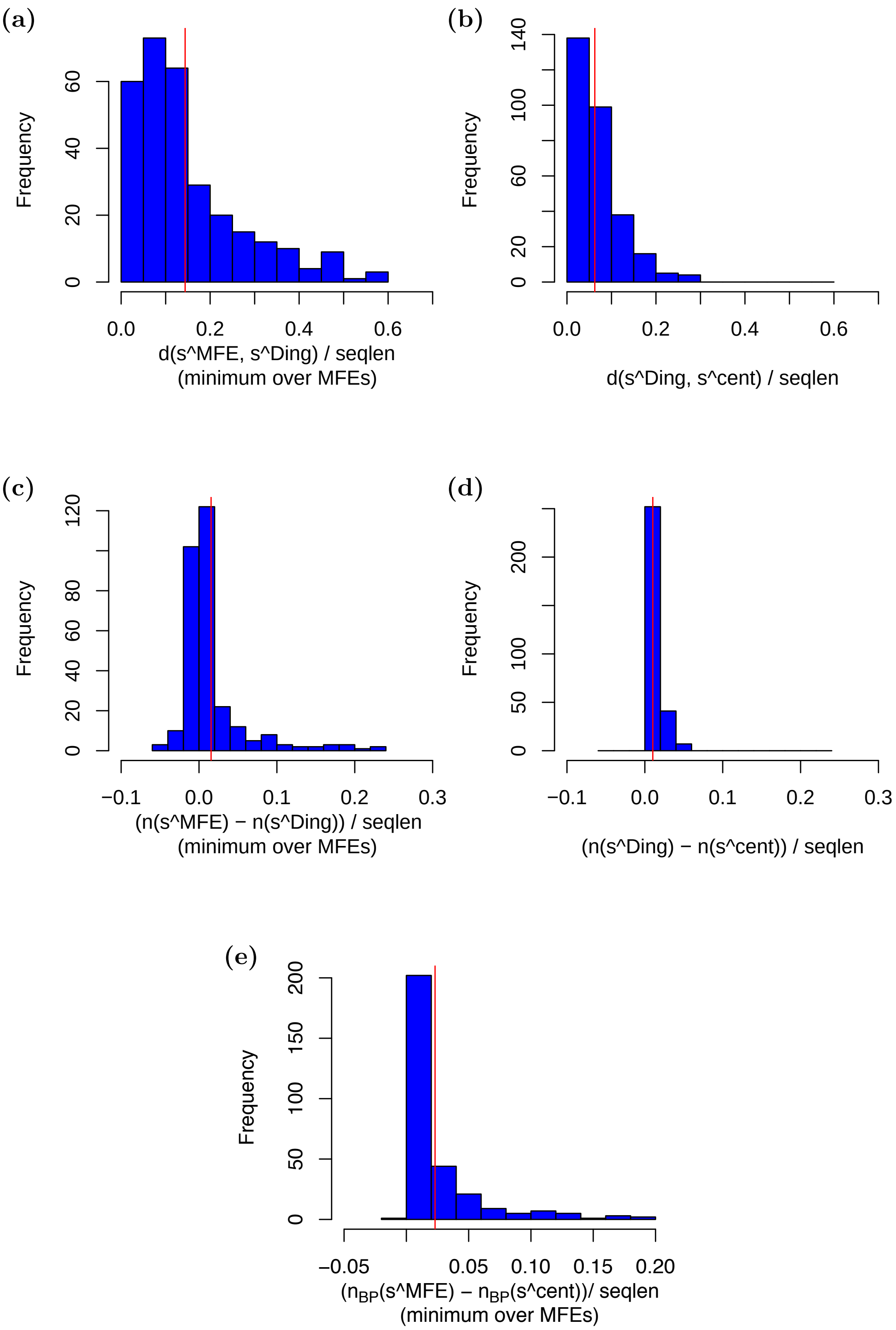


Figure 5.6: Comparing the centroid with the Ding centroid for 300 random sequences 200 nucleotides long. The red line indicates the mean.

interval from d to e conditional on d base-pairing with e. This is computed via the recursion

$$F^b_{i,j} = \min\left\{F^{\text{hairpin}}_{i,j}, \min_{i\le d<e\le j}\left\{F^{\text{interior}}_{i,d,e,j} + F^b_{d,e}\right\}, \min_{i\le d<e\le j}\left\{F^m_{i,d-1} + F^b_{d,e} + F^{\text{multi}}\right\}\right\}. \quad (5.4)$$

The structure within the pair $d \cdot e$ can be empty and have a hairpin energy. Alternatively, it can contain a single nested pair with an interior loop energy (the base stack that makes up a helix is a special case of this) or multiple pairs with a multiloop energy. A third recursion, F^m, contains the minimum free energy of the structure within the multiloop:

$$F^m_{i,j} = \min\left\{\min_{i\le d<e\le j}\left\{F^b_{d,e} + F^{\text{multi}}\right\}, \min_{i\le d<e\le j}\left\{F^m_{i,d-1} + F^b_{d,e} + F^{\text{multi}}\right\}\right\}. \quad (5.5)$$

The multiloop can terminate with a single base pair or can have a pair and an additional multiloop. The quantity $F_{1,N}$ is the minimum free energy for the entire sequence. Though these recursive quantities are best understood going from larger to smaller subsequences, in dynamic programming these recursions are computed from shortest to longest subsequences so that no quantity need be computed more than once and each value is available when it is needed to compute a longer subsequence. Algorithm 5.1 shows pseudocode for the computation of these recursions. The four levels of nested **for** loops correspond to a time complexity of $O(N^4)$. Once the MFE is known, the structure having that energy can be found by backtracking through the recursions. That is, starting from $F_{1,N}$, we note at each level which recursion (or possibly recursions) gives the optimal energy.

The recursions to compute the centroid structure are similar, but in many ways simpler than those used to compute the MFE. The F^m and F^b recursion are needed to find the MFE because of the details of the loop-based energy model. In computing the centroid structure, all information regarding the structures' energies is contained in the pair probability matrix. This allows for simpler recursions. However, unlike the MFE where the empty structure has zero energy, there is a score associated with an empty substructure to account for the $(N+1)^{st}$ column of the pair probability matrix. Thus, we have the recursion C^e for empty substructures. Then, as in the MFE case, the optimal centroid score for a subsequence i, j is either the empty substructure or a rightmost base pair between d and e. The full set of recursions is

$$\begin{aligned} C^e_{i,j} &= C^e_{i+1,j} - P_{i,N+1} && (5.6)\\ C^b_{i,j} &= C_{i+1,j-1} - 2P_{i,j} && (5.7)\\ C_{i,j} &= \min\left\{C^e_{i,j}, \min_{i\le d<e\le j}\left\{C_{i,d-1} + C^b_{d,e} + C^e_{e+1,j}\right\}\right\}. && (5.8)\end{aligned}$$

The information from the pair probability matrix is captured in the C^e and C^b recursions, which account for unpaired bases and base pairs, respectively. Algorithm 5.2 shows pseudocode for this algorithm. The minimization over indices i, d, e, and j in (5.8) corresponds to $O(N^4)$ time complexity.

The time complexity for the centroid calculation can be reduced to $O(N^3)$ by calculating C using the supplementary recursion $C^s_{d,j}$, which considers all structures with a base pair between an index d and some other index between d and j. That is, by performing the loop over the right side of the base pair ahead of time, we remove one level of nested **for** loops. The revised set of recursions is

$$\begin{aligned} C^s_{i,j} &= \min_{i<d\le j}\left\{C^b_{i,d} + C^e_{d+1,j}\right\} && (5.9)\\ C_{i,j} &= \min\left\{C^e_{i,j}, \min_{i\le d<j}\left\{C_{i,d-1} + C^s_{d,j}\right\}\right\}, && (5.10)\end{aligned}$$

where C^b and C^e are as above. Algorithm 5.3 shows pseudocode for this reduced-complexity approach. The optimal centroid score is $n(s^{\text{cent}}) = N + C_{1,N}$ and the structure itself is found by backtracking through the recursions.

5.2.2 A complex of interacting strands

Dirks et al. [19] present algorithms to compute the partition function for complexes of multiple interacting strands. Following their lead, we present recursions to find the centroid for an ordered complex of strands and for a box with a finite number of strands.

For a multistranded complex the strands are ordered and drawn from $5'$ to $3'$ with a *nick* at each strand break. By convention, a nick between bases i and $i+1$ is given the index $i+\frac{1}{2}$. The function $\eta[i+\frac{1}{2}, j-\frac{1}{2}]$ returns the number of nicks in the interval between $i+\frac{1}{2}$ and $j-\frac{1}{2}$, and $\eta[i+\frac{1}{2}]$ returns one if there is a nick at $i+\frac{1}{2}$. For a complex of L strands there are $(L-1)!$ circular permutations that correspond to different orderings

```
Initialize $(F, F^b, F^m)$ with all entries set to 0   // $O(N^2)$ space
for $l = 1, N$
  for $i = 1, N-l+1$
    $j = i+l-1$
    // $F^b$ recursion
    if $l \geq 5$
      $F^b_{i,j} = F^{\text{hairpin}}_{i,j}$
    for $d = i, j-4$
      for $e = d+4, j$
        $F^b_{i,j} = \min\{F^b_{i,j}, F^{\text{interior}}_{i,d,e,j} + F^b_{d,e}\}$
        $F^b_{i,j} = \min\{F^b_{i,j}, F^m_{i+1,d-1} + F^b_{d,e} + F^{\text{multi}}\}$
    // $F, F^m$ recursion
    for $d = i, j-4$
      for $e = d+4, j$
        $F_{i,j} = \min\{F_{i,j}, F_{i,d-1} + F^b_{d,e}\}$
        $F^m_{i,j} = \min\{F^m_{i,j}, F^b_{d,e} + F^{\text{multi}}\}$
        $F^m_{i,j} = \min\{F^m_{i,j}, F^m_{i,d-1} + F^b_{d,e} + F^{\text{multi}}\}$
// $n(s^{\text{MFE}}) = F_{1,N}$; $s^{\text{MFE}}$ is found by back-tracking
```

Algorithm 5.1: Pseudocode to find the minimum free energy (MFE) structure. This algorithm has time complexity $O(N^4)$.

```
Initialize $(C, C^b, C^e)$ with all entries set to 0   // $O(N^2)$ space
Load base-pairing probability matrix $P$   // $O(N^2)$ space
for $l = 1, N$
  for $i = 1, N-l+1$
    $j = i+l-1$
    // $C^e$ recursion
    $C^e_{i,j} = C^e_{i+1,j} - P_{i,N+1}$
    // $C^b$ recursion
    if $l \geq 5$
      $C^b_{i,j} = C_{i+1,j-1} - 2P_{i,j}$
    // $C$ recursion
    $C_{i,j} = C^e_{i,j}$
    for $d = i, j-4$
      for $e = d+4, j$
        $C_{i,j} = \min\{C_{i,j}, C_{i,d-1} + C^b_{d,e} + C^e_{e+1,j}\}$
// $n(s^{\text{cent}}) = N + C_{1,N}$; $s^{\text{cent}}$ is found by back-tracking
```

Algorithm 5.2: Pseudocode to find the centroid structure, $\operatorname{argmin} n(s)$. This algorithm has time complexity $O(N^4)$. The computation proceeds from shortest to longest subsequences.

of the strands. For example, three strands have two circular permutations, 123 and 213. The L cyclic permutations, the orderings 123, 231, and 312, are equivalent, since whether a secondary structure is non-pseudoknotted is invariant under rotation.[3] Let $\bar{\Omega}$ be the set of non-pseudoknotted structures for a complex, and let $\bar{\Omega}(\bar{\psi})$ be those restricted to a particular circular permutation. Here we consider an *ordered complex*, that is, a complex with a particular choice of circular permutation.

The recursions are similar to those presented in the previous section, but the centroid structure for an ordered complex must be connected. To ensure this, $C^e_{i,j}$ is infinite unless $\eta[i + \frac{1}{2}, j - \frac{1}{2}]$ is zero. Algorithm 5.4 shows pseudocode that incorporates these nick checks in an algorithm that requires $O(N^4)$ time. As before, we can reduce the time complexity to $O(N^3)$ by storing intermediate results in the matrix C^s (Algorithm 5.5).

5.2.2.1 A box containing a finite number of strands

With the ability to find centroid structures for single strands and multi-stranded complexes, we can find the centroid structure for a box containing a finite number of distinctly labeled strands. This is the situation when one performs a kinetic simulation with multiple strands, as in Chapter 3.

Assume we have the set of strands $A = \{a_1, \ldots, a_k\}$. Begin by generating all possible complexes from A, that is, the power set of A minus the empty set. The centroid for each complex is found via minimization over the circular permutations, that is, over the ordered complexes. The centroid for the box is found by choosing the partition of A (into complexes) that minimizes the sum of $n(s^{\text{complex}})$ over the complexes. Since the number of complexes is $2^k - 1$ and the number of circular permutations for a complex of size L is $(L-1)!$, this procedure is only practical for boxes containing a small number of species.

5.2.2.2 Distinguishability issues

The algorithms for computing the centroid of an ordered complex or for a box assume that all strands can be distinguished even if they share the same sequence. Experimentally, strands of the same sequence are indistinguishable, and we would like the centroid structure to reflect that fact. Dirks et al. [19] present a distinguishability correction to account for

[3] An alternative way to draw structures is to draw the nucleic acid backbones around the outside of a circle and base pairs as chords. Then a secondary structure is non-pseudoknotted if there are no crossing chords.

```
Initialize $(C, C^s, C^b, C^e)$ with all entries set to 0   // $O(N^2)$ space
Load base-pairing probability matrix $P$   // $O(N^2)$ space
for $l = 1, N$
  for $i = 1, N{-}l + 1$
    $j = i{+}l{-}1$
    // $C^e$ recursion
    $C^e_{i,j} = C^e_{i+1,j} - P_{i,N+1}$
    // $C^b$ recursion
    if $l \geq 5$
      $C^b_{i,j} = C_{i+1,j-1} - 2P_{i,j}$
    // $C^s$ recursion
    for $d = i + 4, j$
      $C^s_{i,j} = \min\left\{C^s_{i,j}, C^b_{i,d} + C^e_{d+1,j}\right\}$
    // $C$ recursion
    $C_{i,j} = C^e_{i,j}$
    for $d = i, j - 4$
      $C_{i,j} = \min\left\{C_{i,j}, C_{i,d-1} + C^s_{d,j}\right\}$
// $n(s^{\text{cent}}) = N + C_{1,N}$; $s^{\text{cent}}$ is found by back-tracking
```

Algorithm 5.3: Pseudocode to find the centroid structure, $\operatorname{argmin} n(s)$, that operates in time $O(N^3)$

```
Initialize $(C, C^b, C^e)$   // $O(N^2)$ space
Set entries of $C^b$ and $C^e$ to $\infty$ and entries of $C$ to 0
Set $C^e_{i+1,i}$ to 0
Load base-pairing probability matrix $P$   // $O(N^2)$ space
for $l = 1, N$
  for $i = 1, N{-}l + 1$
    $j = i{+}l{-}1$
    // $C^e$ recursion
    if $\eta[i + \frac{1}{2}, j - \frac{1}{2}] == 0$
      $C^e_{i,j} = C^e_{i+1,j} - P_{i,N+1}$
    // $C^b$ recursion
    if $l \geq 2$
      if ($l \geq 5$ or $\eta[i + \frac{1}{2}, j - \frac{1}{2}] == 0$) and ($\eta[i + \frac{1}{2}] == 0$ or $\eta[j - \frac{1}{2}] == 0$ or $j == i + 1$)
        $C^b_{i,j} = C_{i+1,j-1} - 2P_{i,j}$
      for $c \in \{i, \ldots, j-1\}$ s.t. $\eta[c + \frac{1}{2}] == 1$
        if ($\eta[i + \frac{1}{2}] == 0$ and $\eta[j - \frac{1}{2}] == 0$) or ($c == i + 1$ and $\eta[j - \frac{1}{2}] == 0$) or ($c == j - 1$ and $\eta[i + \frac{1}{2}] == 0$)
          $C^b_{i,j} = \min\{C^b_{i,j}, C_{i+1,c} + C_{c+1,j-1} - 2P_{i,j}\}$
    // $C$ recursion
    $C_{i,j} = C^e_{i,j}$
    for $d = i, j - 1$
      for $e = d + 1, j$
        if $\eta[e + \frac{1}{2}, j - \frac{1}{2}] == 0$ and ($\eta[d - \frac{1}{2}] == 0$ or $d == i$)
            $C_{i,j} = \min\{C_{i,j}, C_{i,d-1} + C^b_{d,e} + C^e_{e+1,j}\}$
// $n(s^{\text{cent}}) = N + C_{1,N}$; $s^{\text{cent}}$ is found by back-tracking
```

Algorithm 5.4: Pseudocode to find the centroid structure of a multistranded complex in time $O(N^4)$

```
Initialize $(C, C^s, C^b, C^e)$   // $O(N^2)$ space
Set entries of $C^b$, $C^e$, and $C^s$ to $\infty$ and entries of $C$ to 0
Set $C^e_{i+1,i}$ to 0
Load base-pairing probability matrix $P$   // $O(N^2)$ space
for $l = 1, N$
  for $i = 1, N{-}l+1$
    $j = i{+}l{-}1$
    // $C^e$ recursion
    if $\eta[i+\frac{1}{2}, j-\frac{1}{2}] == 0$
      $C^e_{i,j} = C^e_{i+1,j} - P_{i,N+1}$
    // $C^b$ recursion
    if $l \geq 2$
      if ($l \geq 5$ or $\eta[i+\frac{1}{2}, j-\frac{1}{2}] == 0$) and ($\eta[i+\frac{1}{2}] == 0$ or $\eta[j-\frac{1}{2}] == 0$ or $j == i+1$)
        $C^b_{i,j} = C_{i+1,j-1} - 2P_{i,j}$
      for $c \in \{i, \ldots, j-1\}$ s.t. $\eta[c+\frac{1}{2}] == 1$
        if ($\eta[i+\frac{1}{2}] == 0$ and $\eta[j-\frac{1}{2}] == 0$) or ($c == i+1$ and $\eta[j-\frac{1}{2}] == 0$) or ($c == j-1$ and $\eta[i+\frac{1}{2}] == 0$)
          $C^b_{i,j} = \min\{C^b_{i,j}, C_{i+1,c} + C_{c+1,j-1} - 2P_{i,j}\}$
    // $C^s$ recursion
    for $d = i+1, j$
      if $\eta[d+\frac{1}{2}, j-\frac{1}{2}] == 0$
        $C^s_{i,j} = \min\{C^s_{i,j}, C^b_{i,d} + C^e_{d+1,j}\}$
    // $C$ recursion
    $C_{i,j} = C^e_{i,j}$
    for $d = i, j-1$
      if $\eta[d-\frac{1}{2}] == 0$ or $d == i$
        $C_{i,j} = \min\{C_{i,j}, C_{i,d-1} + C^s_{d,j}\}$
// $n(s^{\mathrm{cent}}) = N + C_{1,N}$; $s^{\mathrm{cent}}$ is found by back-tracking
```

Algorithm 5.5: Pseudocode to find the centroid structure of a multistranded complex in time $O(N^3)$

the indistinguishabiliy of strands of the same sequence in partition function calculations. This correction is straightforward for partition function calculations, but to find the MFE structure for indistinguishable strands, one must potentially enumerate an exponentially large number of structures.

The first issue that we face is that the notion of closeness to the ensemble, $n(s)$, needs redefinition in the context of indistinguishable strands. For this section only, we denote the ensemble where strands of like sequence are distinguishable with an over bar. Thus $\bar{\pi}(\bar{s}, \bar{\psi})$ is the equilibrium measure where strands of like sequence can be distinguished and $\pi(s, \psi)$ is the measure when strands of like sequence are indistinguishable. In both cases, we have fixed a particular circular permutation $\bar{\psi} \in \bar{\Psi}$ or $\psi \in \Psi$.

We can view the state space, $\Omega(\pi)$, as a partitioning of $\bar{\Omega}(\bar{\psi})$, where $s \in \Omega(\psi)$ is the set of equivalent structures (with distinguishable strands) that are encountered when recursing over the set $\bar{\Omega}(\bar{\psi})$. Then,

$$\pi(s, \psi) = \sum_{\bar{s} \in s} \bar{\pi}(\bar{s}, \bar{\psi}). \tag{5.11}$$

For distinguishable strands, $n(\bar{s}, \bar{\psi})$ is defined by

$$n(\bar{s}, \bar{\psi}) = \sum_{\bar{\sigma} \in \bar{\Omega}} \bar{\pi}(\bar{\sigma}, \bar{\psi}) |S(\bar{s}) - S(\bar{\sigma})|, \tag{5.12}$$

and we can compute a centroid structure as shown in the previous section.

In defining $n(s)$ for a structure where strands of the same sequence are indistinguishable, the first hurdle is defining an appropriate distance. Since structures $s \in \Omega$ represent disjoint sets of structures $\bar{s} \in \bar{\Omega}$ we can think of the distance as the distance between two sets of structures. The average distance between all pairs of elements in the sets is not a metric because the distance between a non-singleton set and itself under averaging is greater than zero. However, the minimum distance between pairs of elements, one from each set, is a metric, and it makes physical sense: By taking the minimum distance we, in essence, line up the structures so as to change as few base pairs as possible before counting the number of differing nucleotides.

Unfortunately, the calculation of equation (5.12) when applied to a structure $s \in \Omega$ computes the average distance between structures $\bar{s} \in \bar{\Omega}$. What we would like to compute is

$$n(\theta, s, \pi) = \sum_{\sigma \in \Omega} p(\theta, \sigma, \pi) \|s - \sigma\|_1 = \sum_{\sigma \in \Omega} p(\theta, \sigma, \pi) \min_{\bar{\sigma} \in \sigma,\, \bar{s} \in s} \|\bar{s} - \bar{\sigma}\|_1. \tag{5.13}$$

Since we must take a minimum for each σ—and there is no reason that the choice for $\bar{\sigma}$ would in any way be consistent over all σ—it seems that we cannot avoid enumerating all structures to compute $n(s)$.

These difficulties in evaluating the objective function $n(s)$ make it unlikely that a polynomial algorithm will be found to compute the centroid for a complex where strands of like sequence are indistinguishable. In the case of computing the MFE for a complex of indistinguishable strands, the correction is always positive [19]. One can compute ΔG^{MFE} assuming distinguishability. By enumerating all secondary structures with energies between ΔG^{MFE} and $(\Delta G^{\mathrm{MFE}} + \Delta G^{\mathrm{correction}})$ one can find the true MFE. For the centroid, the correction to $n(s)$ will be negative. Then, unless one could devise a bound on the size of the correction, we would potentially have to enumerate all structures in the ensemble to find the centroid for the complex with indistinguishable strands.

5.3 Summary

The centroid structure of a single-strand or an ordered complex can be efficiently computed by a dynamic program in time $O(N^3)$, where N is the sequence length. In contrast to similar algorithms for computing the MFE or partition function, all information regarding the structures' energies and probabilities is contained in the pair probability matrix. This makes the recursions themselves simpler. Moreover, we can compute a centroid for any ensemble for which we can construct a pair probability matrix without any information about particular secondary structures in the ensemble.

The centroid is the optimal representative of an ensemble of secondary structures in that it minimizes the average distance to all members of the ensemble. The MFE is universally used as a representative structure, and though the centroid is optimal, it is not much better at characterizing the ensemble than the MFE. Thus, its utility will be greatest for non-equilibrium ensembles, such as in a kinetic simulation, where the MFE cannot be computed by dynamic programming.

Chapter 6

Conclusions and Outlook

The three methods presented in this thesis all attempt to devise a concise, easily interpreted, and physically meaningful representation of a complex nucleic acid system.

The centroid structure is the optimal characterization of an ensemble of secondary structures in the sense that it minimizes the distance to every structure in the ensemble. Though optimal, the centroid does not characterize the ensemble much better than the universally used minimum free energy structure. Ding et al. [16, 17] seek to characterize the ensemble through the combination of an alternate centroid and clustering of sampled structures. They report impressive gains from their approach. In light of the results of Section 5.1.3 it would be interesting to tease apart the gains from clustering and from using the centroid in place of the MFE. Because we see few gains from the centroid as compared with the MFE, we hypothesize that clustering with the MFE structure would perform as well as clustering with the centroid, though it is easier to compute the centroid than the MFE from the pair probability matrices that characterize each cluster.

The ability to compute the centroid structure for a box of labeled strands was critical for the trajectory-based method of Chapter 3. It would be enormously useful if we could compute the centroid for a complex of indistinguishable strands efficiently. This could replace the brute-force method for computing the MFE that is currently used. Given the difficulties even in evaluating $n(s)$ when strands of like sequence are indistinguishable, constructing an efficient algorithm will take new insight into the problem.

The trajectory-based method presented in Chapter 3 sought to mimic the landscape-partitioning approach of Chapter 2 or the eigenvector-based method of Deuflhard et al. [14] without having to write down the exponentially large rate matrix. Our clustering is distinct from that of Ding et al. [16] because we are interested in not only the equilibrium ensemble, but also any macrostates that are kinetically important as on- or off-pathway intermediates given a particular starting configuration.

The method is a combination of many elements, several of which deserve mention. The first is our stopping criterion for clustering, which departs from prior criteria by acknowledging that the objects being clustered are themselves distributions of other objects, and basing our stopping criterion on how well clustered the underlying objects are.

Though the maximum likelihood estimate and confidence interval procedures that we use in computing rate constants are well established, we add a key new element: By running many short simulations, we compute the partition function for each macrostate without enumerating secondary structures. This allows us to only estimate "downhill" transition rates and compute the reverse rates by enforcing detailed balance with respect to the macrostate equilibrium measure. This is particularly important when the forward rates are essentially irreversible, as is the case for most NA systems studied in our laboratory.

The transition identification procedure is key to the success of the method, and, to our knowledge, without precedent in the simulation and model reduction literatures. Voter [51] addressed the need for a method to detect transitions while running molecular dynamics simulations. He addresses this issue by periodically halting the simulation and doing an energy minimization to find which basin the molecule is in. Our identification procedure would be of great use in such a situation since the search for transitions could be done as the simulation proceeds.

Two issues would need to be addressed in order to apply the transition identification procedure to new problems. First, we must be able to simulate the system for a long enough time to observe some transitions. The distributed approach presented in [47] may not yield simulations long enough to explicitly average over τ, not to mention observe transitions. The second issue is developing an appropriate measure on the state space with which to compute the distance in variation. Secondary structures are a natural choice for nucleic acids and have proved their usefulness in thermodynamic calculations. For simulations in continuous space, an approach might be to discretize the trajectory then apply ideas from diffusion maps [8], where the diffusion timescale would be much shorter than τ, or set-oriented methods [9]. Alternatively, if one had a reaction coordinate or set of coarse variables for the system, the transition identification procedure would locate the divisions between macrostates in those variables.

Bibliography

[1] Agresti, A. and B. A. Coull (1998, May). Approximate is better than "exact" for interval estimation of binomial proportions. *The American Statistician 52*(2), 119–126.

[2] Akutsu, T. (2000). Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics 104*, 45–62.

[3] Alberts, B., A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter (2001). *Molecular Biology of the Cell* (4th ed.). New York: Garland Science.

[4] Aldous, D. (1989). *Probability Approximations via the Poisson Clumping Heuristic.* New York: Springer-Verlag.

[5] Aldous, D. and J. Fill (1999). Reversible Markov chains and random walks on graphs. Monograph in preparation.

[6] Ben Arous, G., A. Bovier, and V. Gayrard (2002). Aging in the random energy model. *Physical Review Letters 88*, 087201–087204.

[7] Brémaud, P. (1999). *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues.* New York: Springer-Verlag.

[8] Coifman, R. R., S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker (2005). Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. *PNAS 102*(21), 7426–7431.

[9] Dellnitz, M. and O. Junge (2002). Set oriented numerical methods for dynamical systems. In G. I. B. Fiedler and N. Kopell (Eds.), *Handbook of Dynamical Systems II: Towards Applications*, pp. 221–264. World Scientific.

[10] Dellnitz, M. and R. Preis (2003). Congestion and almost invariant sets in dynamical systems. In F. Winkler (Ed.), *Proceedings of SNSC 2001*, New York, pp. 183–209. Springer-Verlag.

[11] Dembo, A. and O. Zeitouni (1998). *Large Deviations Techniques and Applications.* New York: Springer-Verlag.

[12] Derrida, B. (1980). Random-energy model: Limit of a family of disordered models. *Physical Review Letters 45*(2), 79–82.

[13] Derrida, B. (1981). Random-energy model: An exactly solvable model of disordered systems. *Physical Review B 24*, 2613–2626.

[14] Deuflhard, P., W. Huisinga, A. Fischer, and C. Schütte (2000). Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear Algebra and its Applications 315*, 39–59.

[15] Dill, K. A. and S. Bromberg (2003). *Molecular Driving Forces: Statistical Thermodynamics in Chemistry and Biology.* New York: Garland Science.

[16] Ding, Y., C. Y. Chan, and C. E. Lawrence (2005). RNA secondary structure prediction by centroids in a boltzmann weighted ensemble. *RNA 11*, 1157–1166.

[17] Ding, Y., C. Y. Chan, and C. E. Lawrence (2006). Clustering of RNA secondary structures with application to messenger RNAs. *Journal of Molecular Biology 359*, 554–571.

[18] Ding, Y. and C. E. Lawrence (2003). A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Research 31*(24), 7280–7301.

[19] Dirks, R. M., J. S. Bois, J. M. Schaeffer, E. Winfree, and N. A. Pierce (2007). Thermodynamic analysis of interacting nucleic acid strands. *SIAM Review 49*(1), 65–88.

[20] Dirks, R. M., M. Lin, E. Winfree, and N. A. Pierce (2004). Paradigms for computational nucleic acid design. *Nucleic Acids Research 32*, 1392–1403.

[21] Dirks, R. M. and N. A. Pierce (2003). A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of Computational Chemistry 24*, 1664–1677.

[22] Dirks, R. M. and N. A. Pierce (2004a). An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. *Journal of Computational Chemistry 25*, 1295–1304.

[23] Dirks, R. M. and N. A. Pierce (2004b). Triggered amplification by hybridization chain reaction. *PNAS 101*, 15275–15278.

[24] Fitzpatrick, S. and A. Scott (1987). Quick simultaneous confidence intervals for multinomial proportions. *Journal of the American Statistical Association 82*(399), 875–878.

[25] Flamm, C., W. Fontana, I. L. Hofacker, and P. Schuster (2000). RNA folding at elementary step resolution. *RNA 6*, 325–338.

[26] Flamm, C., I. L. Hofacker, P. F. Stadler, and M. T. Wolfinger (2002). Barrier trees of degenerate landscapes. *Z. Phys. Chem. 216*, 155–173.

[27] Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association 58*(301), 13–30.

[28] Isambert, H. and E. D. Siggia (2000). Modeling RNA folding paths with pseudoknots: Application to hepatitis delta virus ribozyme. *PNAS 97*, 6515–6520.

[29] Jain, A. K., M. N. Murty, and P. J. Flynn (1999). Data clustering: A review. *ACM Computing SUrveys 31*(3), 264–323.

[30] Kawasaki, K. (1966, May). Diffusion constants near the critical point for time-dependent Ising models. I. *Physical Review 145*(1), 224–230.

[31] Lance, G. N. and W. T. Williams (1967). A general theory of classificatory sorting strategies: 1. hierarchical systems. *Computer Journal 9*(4), 373–380.

[32] León, C. A. and F. Perron (2004). Optimal Hoeffding bounds for discrete reversible Markov chains. *The Annals of Applied Probability 14*(2), 958–970.

[33] Lyngsø, R. B. and C. N. S. Pedersen (2000). RNA pseudoknot prediction in energy-based models. *Journal of Computational Biology 7*(3/4), 409–427.

[34] Lyngsø, R. B., M. Zuker, and C. N. S. Pedersen (1999). Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics 15*(6), 440–445.

[35] Mathews, D. H., J. Sabina, M. Zuker, and D. H. Turner (1999). Expanded sequence dependence of thermodynamics parameters improves prediction of RNA secondary structure. *Journal of Molecular Biology 288*(5), 911–940.

[36] Maulik, U. and S. Bandyopadhyay (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*(12), 1650–1654.

[37] McCaskill, J. S. (1990). The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers 29*, 1105–1119.

[38] Miklós, I., I. M. Meyer, and B. Nagy (2005). Moments of the Boltzmann distribution for RNA secondary structures. *Bulletin of Mathematical Biology 67*, 1031–1047.

[39] Monthus, C. and J.-P. Bouchaud (1996). Models of traps and glass phenomenology. *Journal of Physics A 29*, 3847–3869.

[40] Nussinov, R., G. Pieczenik, J. R. Griggs, and D. J. Kleitman (1978). Algorithms for loop matchings. *SIAM Journal on Applied Mathematics 35*(1), 68–82.

[41] Rivas, E. and S. R. Eddy (1999). A dynamic programming algorithm for RNA structure prediction including pseudoknots. *Journal of Molecular Biology 285*, 2053–2068.

[42] Ross, S. M. (1987). *Introduction to Probability and Statistics for Engineers and Scientists*. New York: Wiley.

[43] SantaLucia, J., H. Allawi, and P. Seneviratne (1996). Improved nearest-neighbor parameters for predicting DNA duplex stability. *Biochemistry 35*(11), 3555–3562.

[44] Schaeffer, J. and E. Winfree (2008). Personal communication.

[45] Seeman, N. C. (2005). From genes to machines: DNA nanomechanical devices. *Trends in Biochemical Sciences 30*(3), 119–125.

[46] Simmel, F. C. and W. U. Dittmer (2005). DNA nanodevices. *Small 1*(3), 284–299.

[47] Singhal, N., C. D. Snow, and V. S. Pande (2004). Using path sampling to build better Markovian state models: Predicting the folding rate and mechanism of a tryptophan zipper beta hairpin. *Journal of Chemical Physics 121*(1), 415–425.

[48] Sundberg, R. (2001). Comparison of confidence procedures for Type I censored exponential lifetimes. *Lifetime Data Analysis 7*, 393–413.

[49] van Batenburg, F. H. D., A. P. Gultyaev, and C. W. A. Pleij (2001). Pseudobase: structural information on RNA pseudoknots. *Nucleic Acids Research 29*(1), 194–195.

[50] van Kampen, N. G. (1992). *Stochastic Processes in Physics and Chemistry.* Amsterdam: North-Holland.

[51] Voter, A. F. (1998). Parallel replica method for dynamics of infrequent events. *Physical Review B 57*(22), R13985–R13988.

[52] Waterman, M. S. and T. F. Smith (1978). RNA secondary structure: a complete mathematical analysis. *Mathematical Biosciences 42*, 257–266.

[53] Widom, B. (1961). Deviations from thermal equilibrium among reactant molecules. *Journal of Chemical Physics 34*(6), 2050–2056.

[54] Widom, B. (1965). Molecular transitions and chemical reaction rates—stochastic model relates rate of a chemical reaction to underlying transition probabilities. *Science 148*, 1555–1560.

[55] Winfree, E., F. Liu, L. A. Wenzler, and N. C. Seeman (1998). Design and self-assembly of two-dimensional DNA crystals. *Nature 394*, 539–544.

[56] Wolfinger, M. T., W. A. Svrcek-Seiler, C. Flamm, I. L. Hofacker, and P. F. Stadler (2004). Efficient computation of RNA folding dynamics. *Journal of Physics A: Mathematical and General 37*, 4731–4741.

[57] Yin, P., H. M. T. Choi, C. R. Calvert, and N. A. Pierce (2008). Programming biomolecular self-assembly pathways. *Nature 451*, 318–322.

[58] Zhang, W. and S.-J. Chen (2002). RNA hairpin-folding kinetics. *PNAS 99*(4), 1931–1936.

[59] Zhang, W. and S.-J. Chen (2003). Analyzing the biopolymer folding rates and pathways using kinetic cluster method. *Journal of Chemical Physics 119*(16), 8716–8729.

[60] Zuker, M. and D. Sankoff (1984). RNA secondary structures and their prediction. *Bulletin of Mathematical Biology 46*, 591–621.

[61] Zuker, M. and P. Stiegler (1981). Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research 9*(1), 133–148.