Kinetically-aware Conformational Distances in Molecular Dynamics

Chen Gu* Xiaoye Jiang† Leonidas Guibas‡

Abstract

In this paper, we present a novel approach for discovering kinetically metastable states of biomolecular conformations. Several kinetically-aware metrics which encode both geometric and kinetic information about biomolecules are proposed. We embed the new metrics into k-center clustering and r-cover clustering algorithms to estimate the metastable states. Those clustering algorithms using kinetically-aware metrics are tested on a large scale biomolecule conformation dataset. It turns out that our algorithms are able to identify the kinetic meaningful clusters.

1 Introduction

Conformational changes are of fundamental importance in a wide range of biological processes including protein folding [4], RNA folding [1] and the operation of key cellular machinery [7]. Extensive genetic, biochemical, biophysical and structural experiments can help to understand these conformational changes. However, probing the mechanisms of conformational changes at atomic resolution is very difficult experimentally and without these details it is impossible to understand the crucial chemistry they perform. On the other hand, computer simulations may complement such experiments by providing dynamic information at an atomic level. With powerful individual processors, or large distributed clusters of processors, one can routinely generate large quantities of simulation data for a given phenomenon of interest. As a result, a growing challenge is how to mine such massive data sets so as to gain insight into the interesting biochemical phenomena under study.

To meet such a challenge, a lot of recent effort has been devoted to constructing stochastic kinetic models, often in the form of discrete-state Markov models, from relatively short molecular dynamics simulations [2]. In order to construct useful mathematical models that can faithfully represent the molecular dynamics at the timescales of interest, it is often necessary to decompose the conformational space into a set of kinetically *metastable states*, or clusters.

In this paper, we present a new method for the discovery of kinetically metastable states that are generally applicable to solvated macromolecules. Given molecular dynamics trajectories consisting of thousands of molecular conformations, our algorithm can identify the long lived, kinetically metastable states by clustering with respect to the *kinetically-aware conformational distances*. Such distance functions encode both the geometry and kinetic information about molecular conformations, which allow robust partitioning of the conformational space into kinetically related regions.

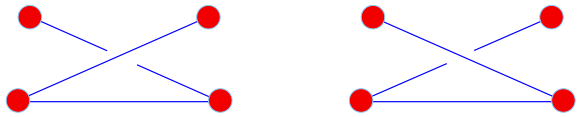


Figure 1: Two conformations which are geometrically close but kinetically far away. Red dots and blue lines denote atoms and bonds respectively.

2 Conformational distance measures

2.1 cRMS distance

In bioinformatics, a commonly used metric for estimating the distance between two molecules is the *coordinate root mean squared (cRMS) distance*. Such a distance can be evaluated as the root mean squared deviation (RMSD) distance[1] of the Cartesian coordinates of heavy atoms in the molecules, optimally aligned by a rigid body translation and rotation minimizing the RMSD [6]. The cRMS distance is a popular choice for biological computation because it possesses all the qualities of a proper distance metric [9], which takes account of both local similarities between molecule conformations and global ones. Moreover, the complexity of estimating the cRMS distance is proportional to the number of atoms, which makes it possible to compute distances between large molecules quickly.

However, a key disadvantage of the cRMS distance is that it ignores the kinetic deformation change from one conformation to another. As illustrated in Figure 1, each of the two conformations has two folded arms, yet the orders that the arms overlap are different. Thus, the two conformations are close geometrically, while they indeed differ greatly kinetically because the deformation change from one to the other has to follow a long trajectory without self-collision in the conformational space. Therefore, it would be more appropriate if we can incorporate such kinetic information from trajectories into the distance functions.

*Institute for Computational and Mathematical Engineering, Stanford University, guc@stanford.edu

†Institute for Computational and Mathematical Engineering, Stanford University, xiaoyej@stanford.edu

‡Department of Computer Science, Stanford University, guibas@cs.stanford.edu

[1]The RMSD distance between two vectors $x = (x_1, \ldots, x_N)^T$, $y = (y_1, \ldots, y_N)^T$ is $\sqrt{\sum_{i=1}^{N}(x_i - y_i)^2/N} = \|x - y\|_2/\sqrt{N}$.

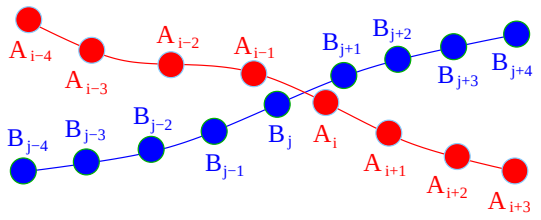


Figure 2: An illustration of the delayed coordinates distance.

2.2 Kinetically-aware conformational distances

In this section, we propose two different kinetically-aware conformational distance functions. The first distance function is defined using the delayed coordinates of conformations which incorporate information of conformational changes at nearby timesteps. The second distance function is given by the shortest path graph distance, while such a graph is constructed based on trajectory dynamics.

2.2.1 Delayed coordinates distance

To define the *delayed coordinates distance* between two conformations, we examine their path context – a set of conformations surrounding them in the trajectory where they come from. Here we assume that the sampling is at the same rate along all trajectories, and each conformation belongs to a unique trajectory in simulation. When we compare two conformations A_i and B_j, we take $2h+1$ samples around each conformation on their paths: $\{A_{i-h}, \ldots, A_i, \ldots, A_{i+h}\}$ and $\{B_{j-h}, \ldots, B_j, \ldots, B_{j+h}\}$ (h is a pre-given sample window size), and define the distance between A_i and B_j as a weighted average of the cRMS distances between the corresponding samples:

$$D(A_i, B_j) = \sum_{\ell=-h}^{h} w_\ell d(A_{i+\ell}, B_{j+\ell}) \qquad (1)$$

In (1), $d(A_{i+\ell}, B_{j+\ell})$ is the cRMS distance between $A_{i+\ell}$ and $B_{j+\ell}$, and all weights w_ℓ's are non-negative. It is easy to verify that the distances defined in (1) satisfy the triangle inequality:

$$\begin{aligned} & D(A_i, B_j) + D(B_j, C_k) \\ = & \sum_{\ell=-h}^{h} w_\ell d(A_{i+\ell}, B_{j+\ell}) + \sum_{\ell=-h}^{h} w_\ell d(B_{j+\ell}, C_{k+\ell}) \\ = & \sum_{\ell=-h}^{h} w_\ell \Big(d(A_{i+\ell}, B_{j+\ell}) + d(B_{j+\ell}, C_{k+\ell}) \Big) \\ \geq & \sum_{\ell=-h}^{h} w_\ell d(A_{i+\ell}, C_{k+\ell}) = D(A_i, C_k). \end{aligned}$$

Therefore, the above defined delayed coordinates distance is a valid metric.

As depicted in Figure 2, A_i and B_j are geometrically very close in the conformational space, but they occur on paths that pass through in very different ways. By considering their path neighbors, we can better characterize their distance because the nearby conformations can help address the kinetic difference between them.

Notice that in (1), we need to compute the best alignment for each conformation pair in the sample window. Alternatively, we can optimize one alignment jointly for all conformation pairs. Without loss of generality, we assume all conformations are centered at the origin (after optimal translation). We map A_i to $A'_i = [w_{-h}A_{i-h}, \ldots, w_0 A_i, \ldots, w_h A_{i+h}]^T$ and B_j to $B'_j = [w_{-h}B_{j-h}, \ldots, w_0 B_j, \ldots, w_h B_{j+h}]^T$, and define $D(A_i, B_j)$ as the cRMS distance between A'_i and B'_j, which is also a valid metric. The optimal alignment (rotation) f between A'_i and B'_j will minimize the following objective function:

$$\begin{aligned} & D^2(A_i, B_j) = d^2(A'_i, B'_j) = \|f(A'_i) - B'_j\|_2^2 \\ = & \sum_{\ell=-h}^{h} \|f(w_\ell A_{i+\ell}) - w_\ell B_{j+\ell}\|_2^2 \\ = & \sum_{\ell=-h}^{h} \|w_\ell f(A_{i+\ell}) - w_\ell B_{j+\ell}\|_2^2 \\ = & \sum_{\ell=-h}^{h} w_\ell^2 \|f(A_{i+\ell}) - B_{j+\ell}\|_2^2 \qquad (2) \end{aligned}$$

(ignoring the constant sacling factor $1/N$ in RMSD definition). So, f gives the best alignment jointly for all conformation pairs in the sample window.

2.2.2 Shortest path graph distance

Given a large number of relatively short conformational trajectories, we can adapt the cRMS distance to reflect the fact that successive conformations along a trajectory should in some sense be closer to each other than their cRMS distance represents, capturing the affinity between the conformations implied by the physical process generating the trajectory.

Since ultimately we deal with a discrete set of conformations, we can consider a large graph of all the conformations along the generated trajectories as nodes and add edges between all pairs with weights given by their corresponding cRMS. To incorporate kinetic information into our distance function, for conformation pairs that are neighbors along a trajectory, we reduce their cRMS distance by multiplying a certain factor $0 < c < 1$, so that conformations along a trajectory are closer to each other than their static distances.

However, after discounting the cRMS distance for certain edges that correspond to conformations along the same trajectory, these new weights may violate the triangle inequality. To retain the metric property, we define the new distances as the lengths of shortest paths in this graph connecting the two conformations in question, which clearly define a metric.

The factor c controls the tradeoff between the static cRMS distances and the kinetic information from trajectories. When $c = 1$, the distance function is purely static; On the other hand, when $c \to 0$, all conformations along a trajectory are arbitrarily close to each

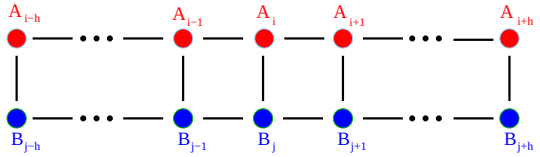


Figure 3: Relation between two distance functions.

other. As a result, each trajectory becomes a cluster itself. Thus, by varying the factor c, we can control the relative amount of geometry and kinetic information that are used in the new distance function.

2.2.3 Relation between two distance functions

Intuitively, these two distance functions represent two different ways to incorporate kinetic information from trajectoties: either penalize conformation pairs from different trajectories, or maintain conformation pairs along a same trajectory close to one another. In fact, both of them can be viewed as graph distances (see Figure 3). In the delayed coordinates distance, we consider a set of $2h+1$ paths from A_i to B_j: $\{A_i \to \ldots \to A_{i+\ell} \to B_{j+\ell} \to \ldots \to B_j\}|_{\ell=-h}^{h}$. Notice that $d(A_{i+\ell}, B_{j+\ell})$ can be seen as the length of the path $\{A_i \to \ldots \to A_{i+\ell} \to B_{j+\ell} \to \ldots \to B_j\}$ with a discount factor $c = 0$, so the delayed coordinates distance $D^{(1)}_{w,h}(A_i, B_j)$ is equal to the weighted average of these $2h+1$ path lengths. In contrast, the shortest path graph distance $D^{(2)}_{c}(A_i, B_j)$ is defined as the minimum path length among all possible paths from A_i to B_j in the complete graph. Therefore, $D^{(1)}_{w,h}(A_i, B_j) \geq \sum_{\ell=-h}^{h} w_\ell \cdot D^{(2)}_{c=0}(A_i, B_j)$.

3 Clustering massive data sets

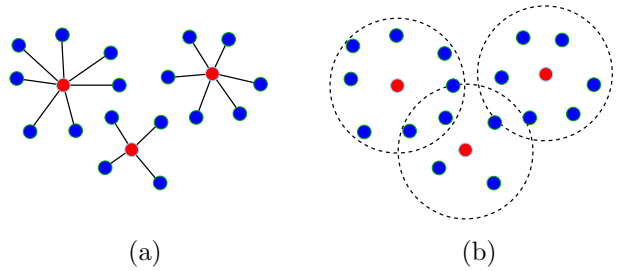
3.1 k-center clustering

Due to the heterogeneous nature of many biological processes at the molecular scale, we usually need a large quantity of simulation data to mine in order to gain insight into the fundamental biochemical phenomena under study. To reach an understanding into the data scientifically, one often needs to shrink the data sets by applying a clustering algorithm to yield a family of clusters (metastable states) of much smaller size than the original data set. Since it is common for simulations conducted on supercomupters to generate data sets that contain $10^5 - 10^7$ conformations in up to 10^4 trajectories, we would prefer a clustering algorithm with computational complexity linear in the number of conformations. In non-Euclidean space, a good candidate for clustering such massive data sets is the k-center clustering.

The k-center problem originates from the *facility location problem*, whose goal is to open k facilities centers among n points such that every point is near some facility center. The problem is formulated as follows:

k-center problem: Given n demand points $\mathcal{D}$ in a metric space, find k supply points $\mathcal{S} \subseteq \mathcal{D}$, such that the maximum distance between a demand point $p \in \mathcal{D}$ and its nearest supply point $q \in \mathcal{S}$ is minimized.

In the k-center problem, the goal is to find the optimal value $r = \min_{\mathcal{S}} \max_{p \in \mathcal{D}} \min_{q \in \mathcal{S}} |p - q|$, and to specify which points should be chosen as centers to satisfy the constraints with that value of r. Notice that if we draw k balls centered at these supply points with radius r, they will cover all n demand points (see Figure 4). Therefore,



(a) (b)

Figure 4: k-center problem (a) and its equivalent formulation (b).

the k-center problem can be equivalently formulated as follows: Given n points $\mathcal{D}$ in a metric space, find k balls of smallest radius centered at $\mathcal{S} \subseteq \mathcal{D}$ which altogether cover every point in $\mathcal{D}$.

A basic fact about the k-center problem is that it is NP-hard. Thus there is no efficient algorithm that always returns the optimal solution. However, there is a simple greedy algorithm called *farthest-first traversal* [3] that works fairly well in practice. The algorithm iteratively picks a new center farthest from the ones chosen so far, and it returns a 2-approximation solution for the k-center problem. In fact, it is not possible to achieve a better approximation ratio for arbitrary metric spaces: even getting a factor $2 - \epsilon$ for any $\epsilon > 0$ is NP-hard [8].

Assuming we can fetch the distance between two points in $\mathcal{O}(1)$ time, farthest-first traversal takes $\mathcal{O}(kn)$ running time and $\mathcal{O}(n)$ working space. So, this algorithm is good for clustering using delayed coordinates distance. However, in the case of shortest path graph distance, we cannot get the pairwise distance from the graph in constant time. As a result, the running time grows to $\mathcal{O}(kn^2)$ since we need to run Dijkstra's algorithm to update the distances from every point to its nearest center in each iteration. In scenarios when k is also large (e.g., clustering all conformations into hundreds of microstates), farthest-first traversal becomes too slow. In the next section, we propose a new clustering algorithm using shortest path distances by considering a related variant problem of the k-center problem, namely, the problem of computing covering numbers.

3.2 r-cover clustering

When we use the k-center clustering, a natural question is how many clusters should we choose (especially for the case when k is large)? As we have seen before, when we cluster data, we implicitly compute the radius r. If we choose a large number for k, then r should be small. In contrast, when k is small, the returned number r should be large. Therefore, it is equivalent to ask how large is the radius we want for clustering? From this observation, we transfer the original k-center problem into a variant problem of computing covering numbers, by swapping the input k and the output r .

r-covering number: Given n demand points $\mathcal{D}$ in a metric space, an r-cover of $\mathcal{D}$ is a set of supply points $\mathcal{S} \subseteq \mathcal{D}$ such that every demand point $p \in \mathcal{D}$ is at most distance r away from its nearest supply point $q \in \mathcal{S}$. The r-covering number of $\mathcal{D}$ is the size of its smallest

r-cover, i.e., $\mathcal{N}(\mathcal{D},r) = \min_{\mathcal{S}}\{|\mathcal{S}| : \max_{p\in\mathcal{D}} \min_{q\in\mathcal{S}} |p-q| \le r\}$.

In the farthest-first traversal algorithm, we repeatedly choose a new center that is farthest from all previous centers, which costs $\mathcal{O}(n^2)$ per iteration for shortest path distances. The main problem here is that we spend a lot of time to compute the real shortest path distances between nodes that are very far from each other. However, by transforming the k-center problem into the r-cover model, it is possible for us to combine Dijkstra's shortest path algorithm and clustering together (see Algorithm 1).

In this r-cover clustering algorithm, we randomly choose an uncovered node as a new center, and run Dijkstra's algorithm to cover all nodes that are at most r away from this new center. Recall that Dijkstra's algorithm finds the real shortest path distances for all nodes in an increasing order. Once we find a node whose real shortest path distance is greater than r from the source, we can stop Dijkstra's algorithm, because all the remaining nodes are outside this cluster and we do not care about their real shortest path distances. Finally, if a node is covered by multiple clusters, it will be assigned to its nearest center at the end of this algorithm.

Let $\mathcal{S}$ be the r-cover returned by Algorithm 1. Then, $\mathcal{N}(\mathcal{D},r) \le |\mathcal{S}| \le \mathcal{N}(\mathcal{D},r/2)$ because all centers in $\mathcal{S}$ are more than r away from each other. Theoretically, this may not be a good approximation for $\mathcal{N}(\mathcal{D},r)$, and the design of a better approximation algorithm for covering numbers is still an open problem [3]. However, our goal here is not to compute covering numbers but use the r-cover model for clustering. Moreover, we can adjust the returned size $|\mathcal{S}|$ by varying the input radius r to approximate the number of clusters we want. We will discuss the running time of Algorithm 1 in Section 4.3.

Algorithm 1 r-cover clustering

Input: A complete graph $G =< V, E >$ and a radius r.
Output: An r-cover of V, according to shortest path distances.
Procedure:
1) Initialize r-cover $S = \phi$.
2) Assign to every node a label $\ell(v) = \infty$ (distance to its nearest center).
3) Randomly pick an uncovered node s as a new center, $S = S \cup \{s\}$.
4) Assign to every node a distance label: $d(s) = 0$ and $d(v) = \infty$ for all other nodes.
5) Mark all nodes as unvisited.
6) Extract node u with smallest $d(u)$ among all unvisited nodes (if all nodes are visited, go to step 12).
7) If $d(u) > r$, go to step 12.
8) If $d(u) \ge \ell(u)$, go to step 11.
9) Update $\ell(u) = d(u)$ (assign node u into this new cluster).
10) Update $d(v) = \min\{d(v), d(u) + w(u,v)\}$ for all unvisited node v.
11) Mark node u as visited, go to step 6.
12) If all nodes are covered, return S; otherwise, go to step 3.

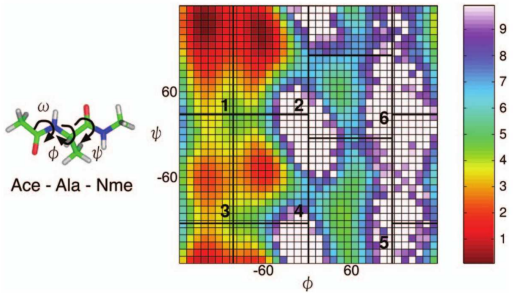


Figure 5: The terminally blocked alanine dipeptide with ϕ, ψ, ω backbone torsions are labeled on the left. Potential of mean force and state decompositions for alanine dipeptide are labeled manually on the right. This picture is taken from [2].

4 Experiments

4.1 Test model - alanine dipeptide

We test our clustering algorithms using kinetically-aware conformational distances on a simple model system, terminally blocked alanine peptide (sequence Ace-Ala-Nme) in explicit solvent. This data set covers both thermodynamic simulations and kinetic simulations useful for testing algorithms analyzing the biomolecular systems, and has already been used in several research papers before [2].

The trajectories were obtained from the 400K replica of a 20ns/replica parallel tempering simulation, and consisted of an equilibrium pool of $1,000$ constant-energy, constant-volume trajectory segments 20ps in length with conformations stored every 0.1ps. A small population of the trajectories contained an ω peptide bond in the *cis* state, rather than the typical *trans* state, were removed from the set of trajectories used for analysis, leaving 975 trajectories with a total of $195,000$ conformations. The minimum cRMS distance between conformation pairs is 3.54×10^{-2}, and the maximum cRMS distance between conformation pairs is 1.87.

In the protein backbone geometry, although there are many degrees of freedom, many of these are not important and what really matters are only a few local angles: the torsion angles ϕ and ψ (see Figure 5) are the primary degrees of freedom on the backbone. Since the slow degrees of freedom (ϕ and ψ) are known a priori, it is relatively straightforward to manually identify metastable states from examination of the potential of mean force, making it a popular choice for the study of biomolecular dynamics. Previously, a master equation model constructed using six manually identified states was shown to reproduce dynamics over long times. We therefore determine whether our algorithms can recover a model of equivalent utility to this manually constructed six-state decomposition for this system. Because the algorithm uses the solute Cartesian coordinates, rather than the (ϕ, ψ) torsions, this is a good test of whether good approximations to the true metastable states can be discovered without prior knowledge of the slow degrees of freedom.

For ease of visualization, we still project the state assignments onto the (ϕ, ψ) torsion map for comparison with the manually constructed states. As depicted in Figure 5, a two-dimensional potential of mean force at 400K in the (ϕ, ψ) backbone torsions was estimated from the parallel tempering simulation using the weighted histogram analysis method by discretizing

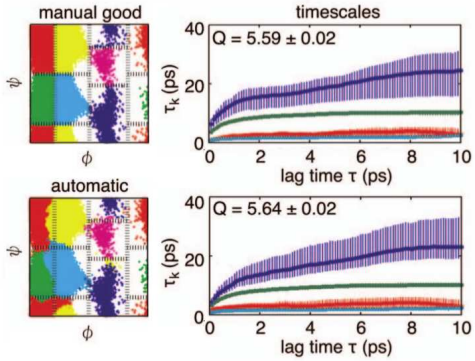


Figure 6: Good manual state decompositions and automatic state decompositions with their implied timescales plots. This picture is taken from [2].

each degree of freedom into 10° bins. The six such states identified in the previous study can be seen adequately separate the free energy basins observed at 400K. In [2], the authors designed an automatic state decomposition algorithm using the method of splitting and lumping to get a good clustering result (see Figure 6). We will take these decompositions as our references of groundtruth decomposition and compare the results from our algorithms with them.

4.2 Clustering results

Among the six states in the manual state decompositions (see Figure 5), states 1 and 2 are the two densest clusters. It is usually difficult to distinguish these two states using the original cRMS distance, because they are more kinetically distinct rather than structurally distinct. States 3 and 4 are two large clusters that are also difficult to be distinguished, but the internal kinetic barrier separating them is smaller than the barrier separating states 1 and 2. The remaining two states 5 and 6 have much smaller sizes than states 1-4.

The clustering result using the cRMS distance is shown in Figure 7-(1). It turns out that directly applying the cRMS metric will cluster states 1 and 2 together. Thus, such a clustering result is a poor decomposition because its states include internal kinetic barriers.

We first test the clustering quality of kinetically-aware conformational distances using delayed coordinates. We set the weights $w_\ell = \exp(-\lambda|\ell|)$ which decay exponentially around the center. When the window size $h = 0$, the metric is simply cRMS. Figure 7-(2-4) shows the clustering results with decay rate $\lambda = 1$ and window sizes $h = 2, 5, 10$ respectively. We can see that as we increase the window size, the conformations in states 1 and 2 become separated. For $h = 10$, the returned six clusters are almost in the same locations as the groundtruth (Figure 7-(4)). If we further increase the window size h, the clustering result will not change too much, because conformations that are far from the center have small weights w_ℓ in the distance function (1).

Figure 7-(5,6) shows two more clustering results that are close to the groundtruth decomposition with different decay rates λ. In Figure 7-(5), the decay rate $\lambda = 0$ and sample window size $h = 12$, so all conformations in the sample window are equally weighted. As a result, the boundaries of the clustering result become ambiguous as there are many outliers in the (ϕ, ψ) torsion map. In Figure 7-(6), the decay rate $\lambda = 0.5$ and the sample window size $h = 12$. By letting $\lambda > 0$, we can reduce the number of outliers significantly.

We have also implemented the alternative approach where we find only one transformation that jointly align two series of conformations in the sample window. As depicted in Figure 7-(7,8), we use decay rates and window sizes $(\lambda = 0.5, h = 5)$ and $(\lambda = 1, h = 5)$ respectively. The clustering quality is also very close to the groundtruth. Notice that the clustering results converge faster in this case because the weights w_ℓ are squared in the objective function (2).

We finally test the kinetically-aware conformational distances using shortest paths (see Figure 7-(9-12)), which use discount factors and radii $(c = 0.9, r = 1.1)$, $(c = 0.8, r = 1.0)$, $(c = 0.7, r = 1.0)$ and $(c = 0.5, r = 0.9)$ respectively. We can see that as we decrease the discount factor c, more kinetic information is incorporated into the distance function, and thus the conformations in states 1 and 2 become separated. For $c = 0.7$, the clustering quality is closest to the groundtruth (Figure 7-(11)). When the discount factor c is too small, conformations from the same trajectory are more likely to be clustered together, while in this case we will observe that the conformations in states 3 and 4 are merged into a single cluster (Figure 7-(12)).

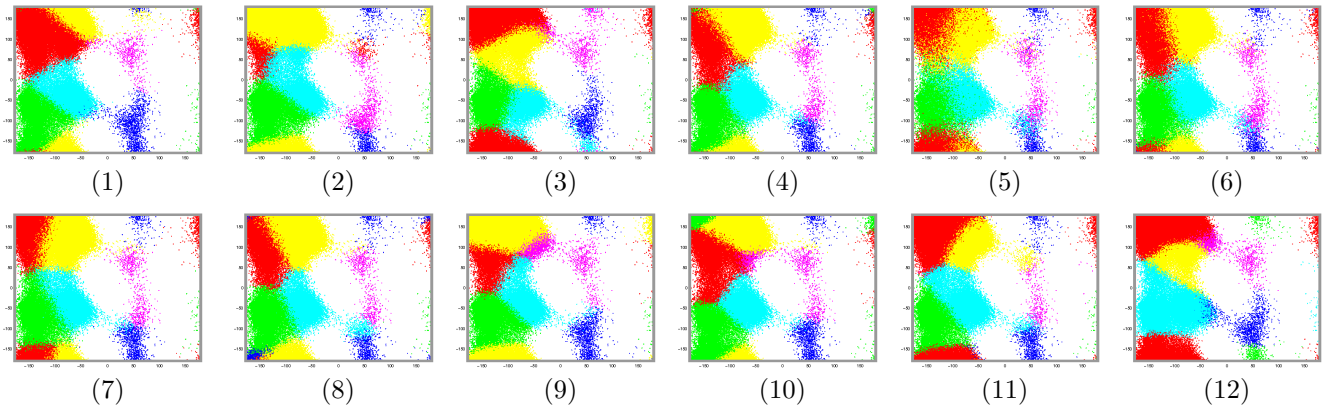
For validation, we examine the implied timescales as a function of lag time (τ), as computed from the eigenvalues of the transition matrix [5]. Theoretically, if the model is Markovian, then the implied timescales will be independent of the lag time for large τ. Figure 8 shows the estimated implied timescales (in picoseconds) as a function of lag time for good decompotitions in Figure 7-(4), (6), (7), (8) and (11) respectively, indicating that they can reproduce dynamics over long times.

4.3 Running time analysis

In this section, we investigate the running time of our clustering algorithms. For k-center clustering, the farthest-first traversal algorithm takes $\mathcal{O}(kn)$ time, which is fairly efficient. For r-cover clustering, we set the discount factor $c = 0.8$ and generate clusters of different sizes by varying the input radius r. The empirical runtime of Algorithm 1 is shown in Table 1. Such an experiment is performed on a computer cluster with AMD Opteron(tm) Processor 250 and 16GB Memory. When $r \to \infty$, the r-cover contains only one node, so the run-

Radius r	0	0.10	0.15	0.20	0.25
Size of r-cover	195,000	42,479	4,826	1,042	377
Runtime (hour)	24.9	63.3	142.3	81.7	109.4
Radius r	0.30	0.40	0.50	1.00	2.00
Size of r-cover	180	67	33	6	1
Runtime (hour)	88.1	65.6	67.9	53.3	24.5

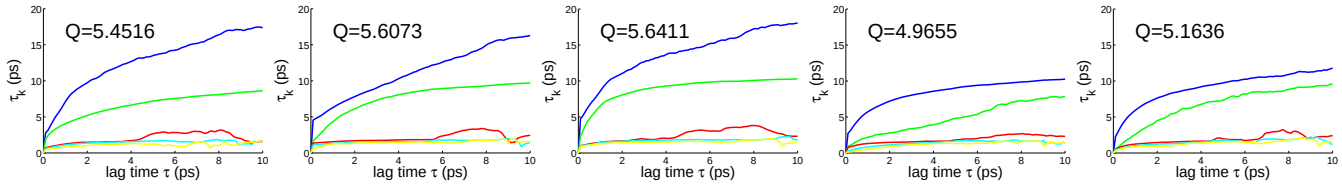
Table 1: Running time of r-cover clustering algorithm.



(1) (2) (3) (4) (5) (6)

(7) (8) (9) (10) (11) (12)

Figure 7: Clustering results with kinetically-aware conformational distances.



(a) Fig 7-(4) (b) Fig 7-(6) (c) Fig 7-(7) (d) Fig 7-(8) (e) Fig 7-(11)

Figure 8: Implied timescales as a function of lag time. The metastability Q is the sum of the self-transition probabilities of the Markov transition matrix.

ning time is $\Theta(n^2)$ by running Dijkstra's algorithm once in a complete graph. As we decrease the radius r, the size of r-cover increases, but we can save more running time from Dijkstra's algorithm because we will never compute the real shortest path distances between nodes that are greater than r. Finally, when $r = 0$, the r-cover contains all nodes in the graph, so we only relax one node (the source) in each Dijkstra's computation, and the total running time is also $\Theta(n^2)$.

For $0 < r < \infty$, the running time is $\Omega(n^2)$ since all nodes in the graph are covered, and it would be larger than those two extreme cases because there exists overlap between different clusters. However, the experimental results in Table 1 show that Algorithm 1 usually runs in $\mathcal{O}(n^2)$ time in practice, which is significantly faster than farthest-first traversal for large k.

5 Conclusions and future work

In this paper, we designed and tested algorithms that use kinetically-aware distances to cluster biomolecular conformations. The proposed approach outperforms the existing methods that only use geometric information within biomolecules to build distance functions. The shortest path graph distance is of particular interest for constructing metric spaces on a discrete point set: Once we have a distance funciton, we need to check whether it satisfes the triangle inequality. If not, we can always form a new metric by using shortest path graph distances. Therefore, it would be interesting to derive a theoretical upper bound on the expected running time of Algorithm 1, or develop other efficient algorithms for clustering using shortest path distances. This would be a topic for our future research.

Acknowledgments

This research was supported by NSF grant IIS 0914833, NSF/NIH grant 0900700, as well as ARO grant W911NF-10-1-0037. The authors wish to thank Xuhui Huang and Lutz Maibaum for their helpful comments and suggestions.

References

[1] P. Brion and E. Westhof. Hierarchy and dynamics of RNA folding. *Annual review of biophysics and biomolecular structure*, 26, 1997.

[2] J. Chodera, N. Singhal, V. Pande, K. Dill, and W. Swope. Automatic discovery of metastable states for the construction of Markov models of macromolecular conformational dynamics. *Journal of Chemical Physics*, 126(15), 2007.

[3] S. Dasgupta. Lecture notes on unsupervised learning. http://cseweb.ucsd.edu/∼dasgupta/291/, 2008.

[4] C. Dobson. Protein folding and misfolding. *Nature*, 426(6968), 2003.

[5] X. Huang, Y. Yao, G. Bowman, J. Sun, L. Guibas, G. Carlsson, and V. Pande. Constructing multi-resolution Markov state models to elucidate RNA hairpin folding mechanisms. *Pacific Symposium on Biocomputing*, 2010.

[6] L. Kavraki. Molecular distance measures. *Connexions*, http://cnx.org/content/m11608/1.23/, 2007.

[7] R. Marshall, C. Aitken, M. Dorywalska, and J. Puglisi. Translation at the single-molecule level. *Annual Review of Biochemistry*, 77, 2008.

[8] S. Plotkin. Lecture notes on advanced algorithms. http://cs361b.stanford.edu/, 2010.

[9] B. Steipe. A revised proof of the metric properties of optimally superimposed vector sets. *Acta Crystallographica Section A*, 58(5), 2002.