

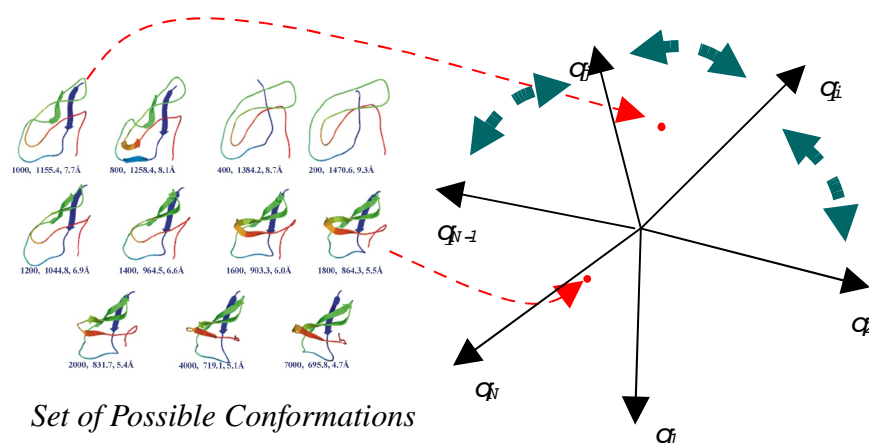
Name: Jiayi Chong

SUID: jyichong

CS273 Scribing Notes for Lecture #3

The conformation of a molecule is a specification of the relative positions of all its atoms in 3D-space. We can describe the conformation using different types of parameterizations. Typically, we pick the atom centers(x, y, z) or the torsional angles as parameterizations. The conformational space is thus the enumeration of all possible conformations.

Every protein has a set of physically-possible conformations. Each of these conformations map to a point in the conformational space. In computational biology, we often want to be able to determine the similarity between different protein conformations as well as the conformations that are of lowest energy.



In order to investigate the motion of proteins as they evolve through these different conformations, we can trace a curve through the different conformations in the conformational space, retrieving the conformation at each point on the curve. The conformational space in proteins is analogous to the configuration space in computer graphics and robotics. In the latter, we often utilize the concept to describe for example, the motion of the body parts of a digitized model.

Simulations in molecular dynamics often produce millions of different conformations. Hence, there is a need for a metric to measure similarities between conformations. We also often need to determine the conformations that are closest to the folded protein. Such a metric is also used in clustering of conformations.

The definition of the metric in conformational space is:

$d: c, c' \in C \rightarrow d(c, c') \in \mathbb{R}^+ \cup \{0\}$ where the 3 conditions below are satisfied

$$1) d(c, c') = 0 \Leftrightarrow c = c' \quad (\text{non-degeneracy})$$

$$2) d(c, c') = 0 \Leftrightarrow c = c' \quad (\text{non-degeneracy})$$

$$3) d(c, c') + d(c', c'') \geq d(c, c'') \quad (\text{triangle inequality})$$

The definition above opens itself up to many different metrics that agree with the above definition. However, not every metric is a good measure of protein similarity. For example, the euclidean metric:

$$d(c, c') = \sum_{i=1, \dots, n} (|\phi_i - \phi_{i'}|^2 + |\psi_i - \psi_{i'}|^2)$$

functions poorly. This is because it gives us incorrect measurements for protein alignment. What we typically use as metrics are cRMSD and dRMSD.

cRMSD

Assuming we have a protein molecule with n atoms and two conformations c and c', then cRMSD(c, c') is defined as:

$$\min_T [(1/n) \sum_{i=1, \dots, n} |a_i(c) - T(a_i(c'))|^2]^{1/2}$$

where $a_i(c)$ is position of a_i when the molecule is at c. T represents a rigid-body alignment transform. cRMSD finds the minimization over all possible rigid-body transforms. The process of finding this minimum T can be done using computer vision methods. Some of these methods employ an iterative scheme to compute the minimum transformation matrix. cRMSD is of $O(n)$ complexity. It is often restricted to a subset of atoms like the protein's backbone.

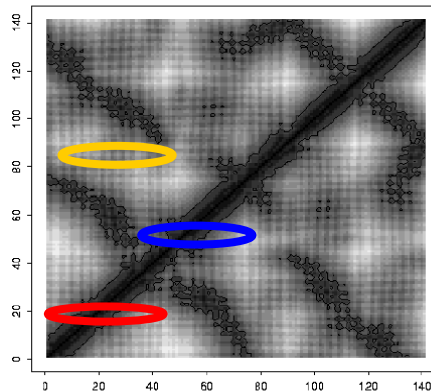
dRMSD

Assuming we have a protein molecule with n atoms and two conformations c and c', then dRMSD(c, c') is defined as:

$$[(1/n(n-1)) \sum_{i=1, \dots, n-1} \sum_{j=i+1, \dots, n} (d_{ij}(c) - d_{ij}(c'))^2]^{1/2}$$

where $\{d_{ij}(c)\}$: n X n symmetrical intra-molecular distance matrix in the molecule at c. Just like cRMSD, dRMSD is also often restricted to a subset of atoms of the molecule.

A visualization of the intra-molecular distance matrix looks like this:



Here, the darker the color, the shorter the distance represented. We can often tell the structure of the protein by examining this matrix. For example, the highlighted red oval represents an alpha-helix while the blue oval represents a protein loop. The advantage of dRMSD over cRMSD is that we do not need to find the alignment transform for different protein conformations. However, dRMSD is of $O(n^2)$ complexity. An important point to note is that dRMSD represents the conformation itself and its mirror image.

K-Nearest-Neighbors

Both cRMSD and dRMSD can be used to find the k-Nearest-Neighbors for a set of given conformations. The total time needed to compute the k-Nearest-Neighbors of every conformation in S is $O(N^2(\log k + L))$. The N represents the number of conformations we have, L the time to calculate the metric and $\log k$ the time needed to keep track of the k nearest neighbors during the computation. We can reduce the running time of this algorithm by either picking a more efficient algorithm or reducing the cost of calculating L .

KD-Tree

One data structure that is useful is the KD-Tree. The KD-Tree adaptively subdivides the current space so as to try to obtain a well-balanced tree structure. In a d -dimensional space, where $d > 2$, range searching for a point in the KD-Tree takes $O(dn^{1-1/d})$. Because of this, we can see that the performance payoff you get as one goes into higher and higher dimensions decreases with a KD-Tree. Hence, such a data structure is only useful at lower dimensions.

m-Averaging, Simplifying the Protein

In order to tackle the computational complexity due to the large number of potential conformations and atoms in protein molecules, we can choose to simplify the protein description by doing m-averaging. First, we assume that each conformation is described by the coordinates of the $C\alpha$ atoms. We know that in the real physical world, atoms cannot bunch up together. Hence, atoms that are very far apart along the chain are on average spatially distant from each other. Instead of considering every single atom in the backbone, we represent sets of atoms in the backbone as fragments

and then replace each fragment by the centroid of the atoms which the fragment is made out of. This cuts down the number of elements we have to evaluate when doing the metric calculations.

Further optimizing dRMSD

We can further optimize dRMSD by doing PC(Principal Component) dimensionality reduction. We do this by first stacking m-averaged distance matrices as vectors over a matrix A. Each vector in this matrix represents a given conformation of the protein. We then compute the SVD(Singular Value Decomposition) of the matrix A:

$$\begin{array}{c}
 \begin{array}{|c|c|} \hline N \\ \hline \begin{array}{|c|} \hline r \\ \hline \end{array} \begin{array}{|c|} \hline A (r \times N) \\ \hline \end{array} \\ \hline \end{array} = \begin{array}{|c|} \hline U (r \times r) \\ \hline \end{array} \begin{array}{|c|} \hline D (r \times r) \\ \hline \end{array} \begin{array}{|c|} \hline V^T (r \times N) \\ \hline \end{array}
 \end{array}$$

Vector a_j of elements of distance matrix of j^{th} conformation ($j = 1$ to N)

The intuition behind this idea is to find the space of greatest variant over all these vector conformations. From this, we can project these vectors into a hyperplane that is of one dimension lower. We do this projection process iteratively until we have reduced the vectors to a small enough dimension to make the metric computation tractable. We utilize SVD to find the axis of the hyperplane and do the projection. Projecting down into lower dimensions works with dRMSD because dRMSD does not require being in a certain coordinate frame to function. One point to note is that SVD itself is a costly algorithm. SVD has computational complexity $O(n^4N)$. Hence, SVD has to be utilized in conjunction with m-averaging in order for it to be effective.

In summary, we can optimize the dRMSD metric by employing m-averaging and PC dimensionality reduction as fast filters. Once the number of dimensions has been reduced to a lower dimensional space, we can use techniques that work well in lower dimensions like the KD-Tree to further speed up the process.