Geometric Approaches for Protein Structural Manipulation

1 Introduction

In Chapter **??** we discussed several ways to represent a protein structure in-silico using cartesian coordinates and internal coordinates, using the values of bond lengths, planar angles and dihedral angles. In order to simulate structural and dynamic processes such as protein folding, protein conformational changes and protein binding, one can generate different three dimensional structures of the same protein by varying the values of the dihedral angles, since in most cases the bond lengths and planar angles change very little and can be considered fixed by many methods.

In this chapter we will discuss methods to manipulate the degrees of freedom of a protein to sample processes such as folding and binding using geometric methods.

2 Robotics-Inspired Approach

Robotics-inspired approach to protein flexibility exploits the similarity between proteins and robots. In particular, there have been algorithms designed in robotics to explore the motion of robot-like objects in a physically constrained environment. The similarity can be exploited to sample protein conformations with constraints that come from the protein's energy in a complex, high-dimensional space spanned by the possible conformations of the protein.

Robotics based algorithms model robotic objects as *articulated manipulators*, which means – a set of links possibly connected by articulated joints with rotational and/or translational degrees of freedom at the joints. Figure 1 (a) shows an illustration of a robotic arm fixed to a base, with three links connected by flexible joints. This is a 2D example, so every flexible joint has one rotational degree of freedom. Some robots are not fixed to a base and can move around the room (think about a vacuum cleaning robot). In this case the robot also has translational degrees of freedom. Similarly, we can think of a molecule as a set of links (atoms bonded to other atoms) connected by flexible joints (in the case of a protein, the dihedral angles are rotatable bonds, see Figure 1 (b)).

2.1 Motion Planning

Motion planning algorithms have been developed as part of robotics to try to plan to motion of a robot-like object in a constrained environment. Imagine a robotic arm performing some work in an industrial setting, for example. The robot should grab objects and move them from one place to another using its translational and/or rotational degrees of freedom, all the while avoiding collisions with known obstacles. Usually, the robot has a start configuration and a goal configuration which denote the start and end of the robot's task. The robotic path planning problem is, given a robot, a physical work space, and starting and goal configurations for the robot, find a collision-free path for the robot from the starting configuration to the goal, if one exists. Otherwise determine that no such path exists.

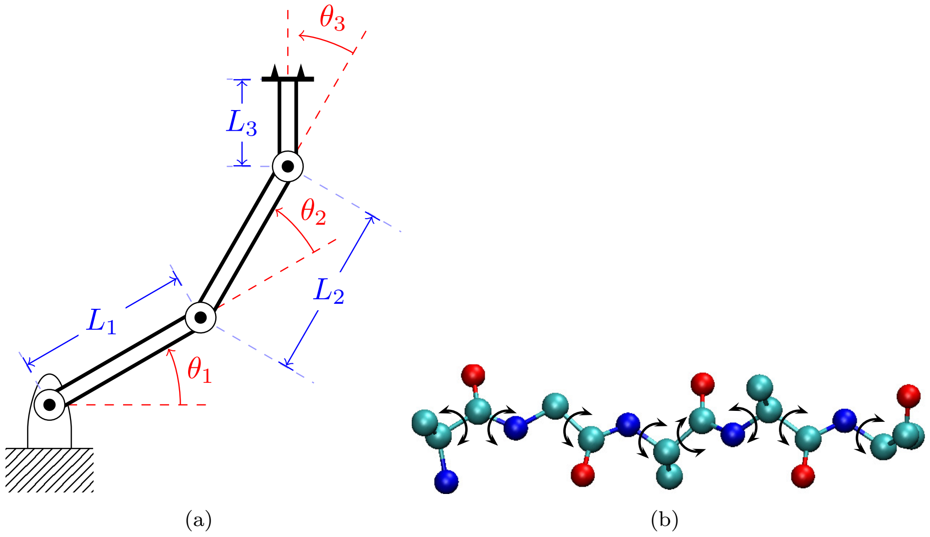


Figure 1: (a) A robotic arm with rotational degrees of freedom at the joints. (b) A protein molecule with the backbone dihedral degrees of freedom.

2.1.1 The Workspace and the Configuration Space

To demonstrate the way motion planning algorithms compute the motion of a robot, let us look at the simple example in Figure 2. Our robotic arm has two links and two joints, hence two rotational degrees of freedom. It is attached to a base and resides on a 2-D plane with several physical obstacles (geometric objects in different colors) and empty space that can be occupied by the robot. This is our physical workspace. Figure 2 (a) shows the start configuration of the robot, and figure 2 (b) shows the goal configuration. Our task is to move the robotic arm from the start to the goal configuration using the arm's rotational degrees of freedom while avoiding the obstacles. Solving the problem on the workspace is not simple, until we realize that if we know the specifications of the robot – the exact point it is attached to the base and the lengths of the two links, all we have to know in order to unambiguously specify its configuration are the values of the two angles, denoted q_1 and q_2. Therefore, we can transform the workspace into an alternative space called the *configuration space* or C-space. This is a mathematical space spanned by the degrees of freedom of the robot. In this case, it's a 2-dimensional space as well (by coincidence), but the dimensionality depends on the number of the degrees of freedom of the robot, not the dimensionality of the workspace. A three-link, three-joint robot embedded in the same space would have a 3-dimensional configuration space. Since the C-space represents the configurations of the robot as a function of its degrees of freedom, every configuration is a mathematical point in the C-space. The obstacles are transformed into "forbidden" areas, which are combination of degrees of freedom that make the robot collide with an object. Figure 3 describe the C-space for the workspace in Figure 2. Remember that the x and y axis represent rotational angles. The colored "blobs" represent forbidden regions, which are combinations of rotational angles which make the robot collide with an object (the colors match the obstacles in Figure 2).

The two black dots in Figure 3 represent the start configuration (figure 2 (a), marked in "1")

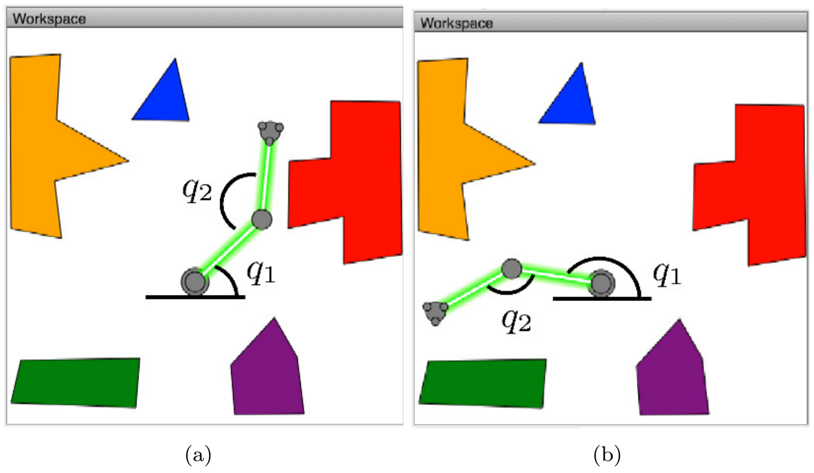


Figure 2: A workspace with a two-link, two-joint robotic arm. The colored shapes represent physical obstacles. The figure was created by: `https://www.cs.unc.edu/~jeffi/c-space/robot.xhtml`.

and the goal configuration (figure 2 (b), marked in ”2”). The path planning problem is to find a mapping $c : [0, 1] \to Q$ where 0 means no collision and 1 means collision, and Q represents the set of points in the configuration space. In other words, we have to find a collision-free path between the two dots. As can be seen in the figure, such path exists (as a matter of fact, an infinite set of paths). An example is marked in blue. Algorithms that try to solve the motion planning problem can sample the collision free space (hence – free space) and find a path by creating a graph that connect dots to their neighbors. From that point on any shortest path algorithm will work. More on this below.

When dealing with molecules, the definitions are rather similar, even though the systems we work with look so different. Just like with physical robots, the molecule is an object embedded in a three dimensional physical space. The work space is defined by the atomic coordinates of the molecule. The 3D arrangement of the molecule can be defined by its set of x, y, z atomic coordinates. However, these coordinates are not independent. The locations of atoms is severely constrained by physico-chemical interactions between the atoms, that determine the inter-atomic distances and angles. The dimensionality of configuration space is determined by the *real* degrees of freedom of the molecule. These are the rotatable bonds we are allowed to modify when changing the configuration of the molecule. In most sampling based algorithms we assume that the covalent bonds and planar angles are fixed, so we are left with the backbone (and sometimes side chain) dihedral angles. Figure 4 shows a dialanine (ALA-ALA) peptide. A peptide with N amino acids has 2N-2 backbone dihedral DOFs, therefore the dialanine peptide has two backbone DOF – one ϕ and one ψ, marked in the Figure. Therefore, the configuration space is two-dimensional in this case. But what are the obstacles? They are not other physical objects, but rather *high-energy configurations*. So, instead of a binary collision function we have a continuous function that depends on the inter-atomic interactions inside the molecule and with other molecules. The energy value of a conformation is proportional to the probability to find this conformation in nature. A high-energy configuration is unlikely to appear in nature and therefore should be avoided. The Figure shows the

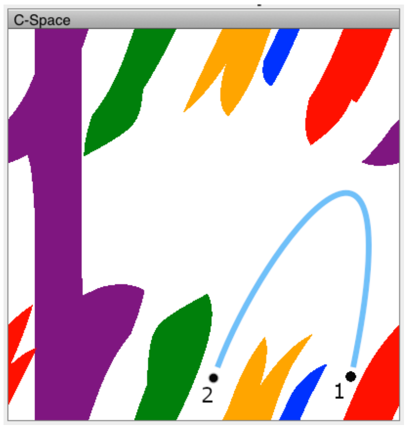


Figure 3: The configuration space for the workspace described in Figure 2. The figure was created by: `https://www.cs.unc.edu/~jeffi/c-space/robot.xhtml`.

energy landscape of the peptide as a function of the two dihedral angles. This is the C-space. To sample the conformations of the peptide, we have to plan a path in the configuration space while avoiding high-energy configurations. Notice that the C-space looks like a mountainous landscape, where the mountains should be avoided. Since this is not a binary function, the user has to define a cutoff above which the energy is considered high. The cutoff depends on the application and the molecule in question. Several approaches will be discussed below.

To recap, here are some important terms:

- **Work space:** The work space is the geometric space in which a robot operates. It consists of obstacles and empty space that may be occupied by the robot.
- **Configuration:** A full description of the robot's state, including its position, orientation, and the states of any internal degrees of freedom (such as revolute joint angles).
- **Collision:** A configuration is said to be in collision if any part of the robot overlaps with either another part of the robot or with a work space obstacle.
- **Free:** A configuration is said to be free if it is not in collision.
- **Configuration space (C-space):** The space of all configurations of the robot, annotated by whether the robot is in collision or free at each configuration.
- **Free space:** The space of all free configurations.

Address: Plan motions in the configuration space but compute in workspace (protein structure, surface or cavity)!

3 kinematics

Kinematics is a branch of classical mechanics that describes the motion of points, bodies (objects), and systems of bodies (groups of objects) without considering the forces that caused the motion.

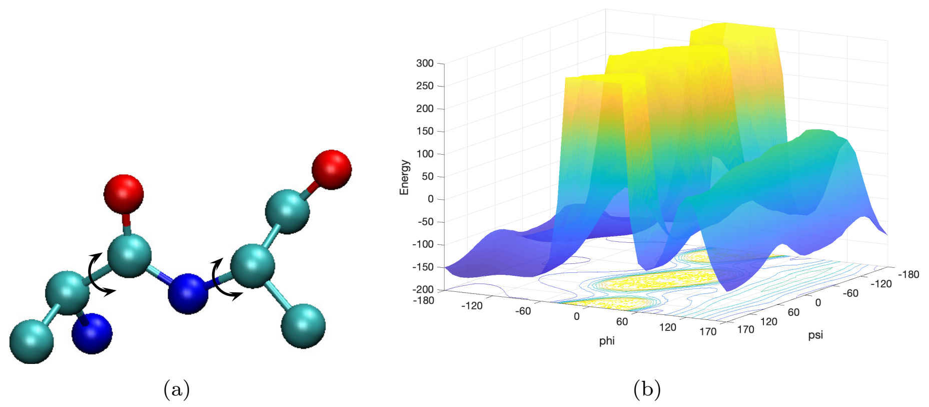

(a) (b)

Figure 4: An example of the energy landscape of the ALA-ALA (dialanine) peptide. It has two degrees of freedom, the backbone dihedral angles ϕ and ψ. High energy values are capped at 300 for clarity.

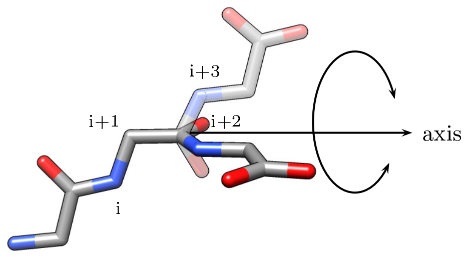


Figure 5: An example of rotating a molecule around a dihedral angle, where the axis is defined by the two middle atoms, $i + 1$ and $i + 2$.

A kinematics problem begins by describing the geometry of a system and the initial conditions of any known values of position, velocity and/or acceleration of points in the system. Then, using geometric methods, the position, velocity and acceleration of any unknown parts of the system can be determined.

Chapter **??** describes how to represent a protein structure and manipulate its structure using matrices or quaternions, as well as internal coordinate representation. When a rotation of θ degrees around a bond is performed, the bond itself is the axis of rotation and hence does not move. As a matter of fact, the only atoms that change their positions are the ones located after the rotation axis. In other words, if the axis is defined by atoms $i+1$ and $i+2$, all the atoms until $i+2$ inclusive remain stationary. See Figure 5.

4 Forward Kinematics

Forward kinematics refers to the use of the kinematic equations of a robot to compute the position of the end-effector (the device at the end of the robotic arm, or the arm's endpoint) from specified values for the joint parameters. In protein motion, the problem becomes computing the new locations of the atoms given a set of dihedral rotations. In the case of protein structures, these dihedral rotations are usually the backbone dihedral angles ϕ and ψ. Even though the rotation is performed in the dihedral angle space, we have to find a way to modify the cartesian coordinates of the rotating atoms ($i+3$ and up). If we apply a rotation by and angle θ around an axis starting at atom a_i on a vector $v = (v_x, v_y, v_z, 1)$ (in homogenous coordinate) the transformation can be written as:

$$R(i, \theta) = T(a_i) \cdot R(axis, \theta) T(-a_i)$$

Since atomic coordinates are generally represented using an arbitrary axis system, the above equation corresponds to translating the atom to its position if the rotation axis passes through the origin, applying the rotation and translating the atom back to its position in the arbitrary axis system. It is also possible to combine several rotations by sorting by position along the protein chain (bond number) and composing the matrices. So for example, applying rotation by θ around an axis starting at atom a_i followed by a rotation by some ϕ around an axis starting at atom $a_j, j > i$ can be defined as:

$$R(\mathit{combined}) = R(j, \phi) \cdot R(i, \theta)$$

Remember that the convention used for matrix-vector multiplication is to multiply column vectors by matrices on the left, so the rightmost transformation is applied first.

Probabilistic Roadmap Motion Planning (PRM)

While not guaranteed to find a path, PRM is *probabilistically complete.* As the number of samples increases, the probability of the planner finding a path if one exists approaches 1. For many real-world path-planning problems, the method is very fast and reliable in practice.

Application of PRM to Protein-Ligand Docking

The A fixed coordinate system P is attached to the protein The ligand is a small flexible molecule A moving coordinate system L is defined using three bonded atoms in the ligand A conformation of the ligand is defined by the position and orientation of L relative to P and the torsional angles of the ligand

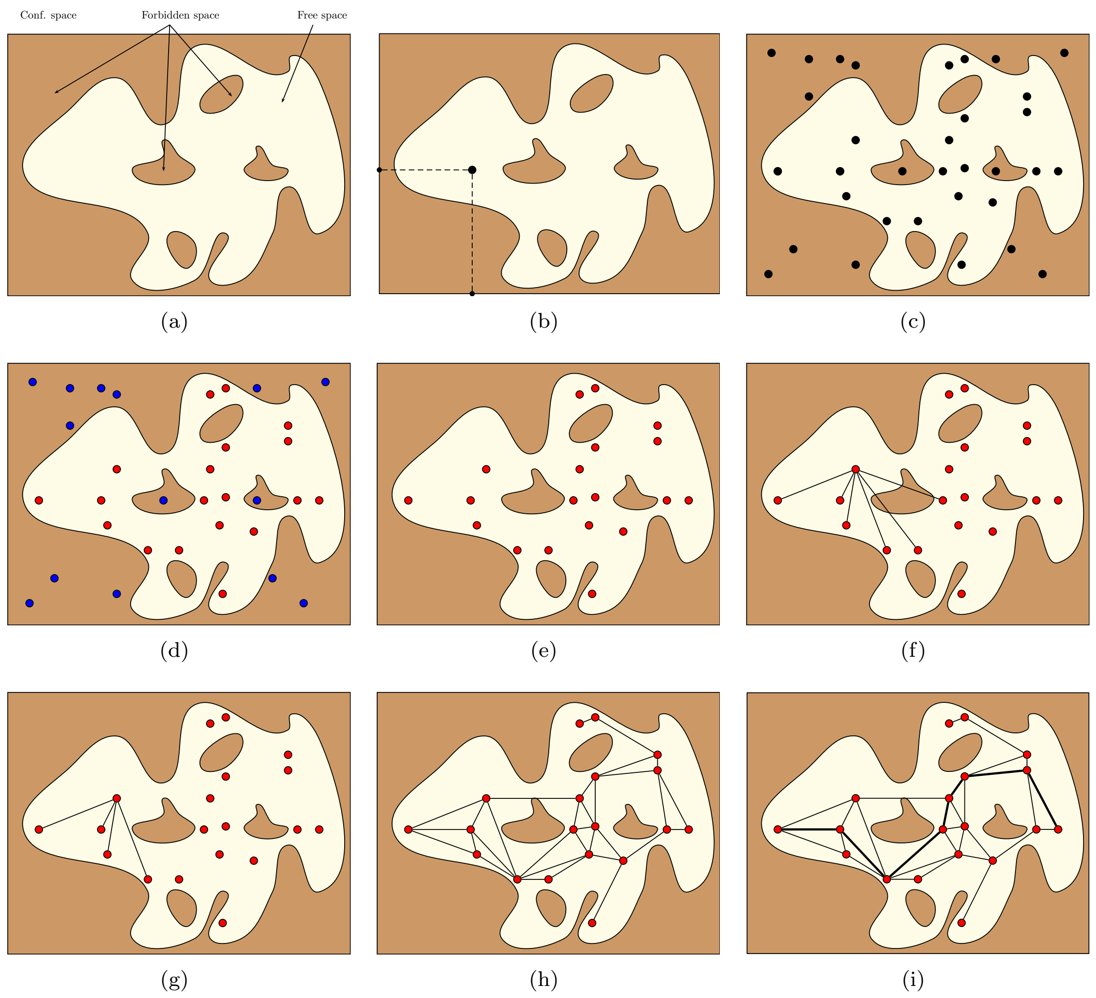
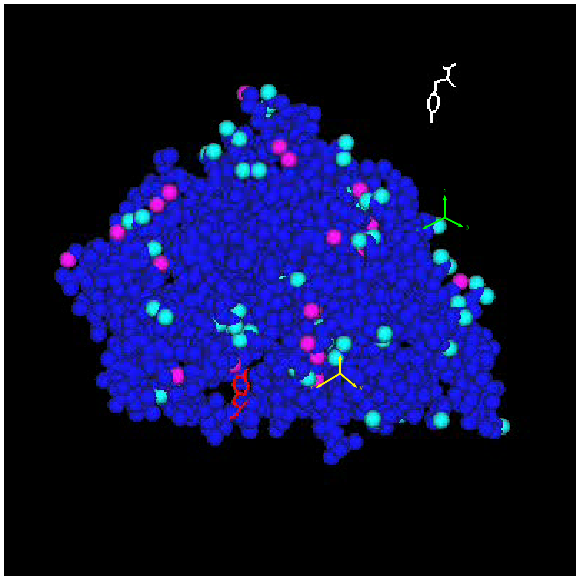
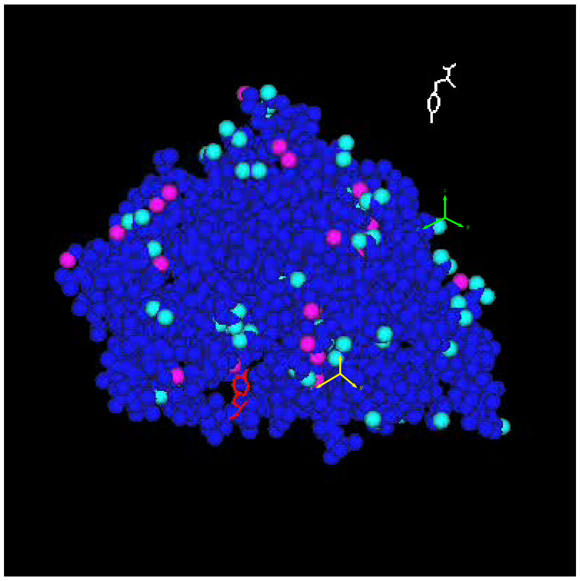


Figure 6: PRM configuration sampling: (a-b) Configurations are sampled by picking coordinates at random and representing them in the configuration space. (c) Sampled configurations are tested for collision (in the workspace!) (d) The collision-free configurations are retained as "milestones" (e) Each milestone is linked by straight paths to its k-nearest neighbors (f) Each milestone is linked by straight paths to its k-nearest neighbors (g) The collision-free links are retained to form the PRM (h) Querying the map to find paths



Roadmap Construction: Node Generation The nodes of the roadmap are generated by sampling conformations of the ligand uniformly at random in the parameter space (around the protein) The energy of each sampled conformation is $E = E_{interaction}$ (electrostatic) + $E_{internal}$ (vdw) A sampled conformation is retained with probability:

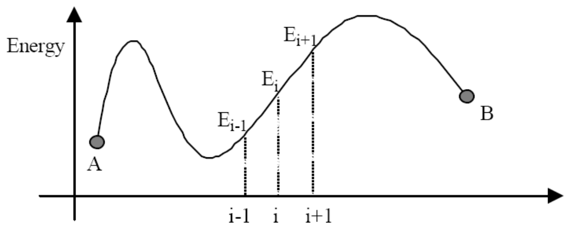
$$p = \begin{cases} 0 & \text{if } E > E_{max} \\ \frac{E_{max}-E}{E_{max}-E_{min}} & \text{if } E_{min} \leq E \leq E_{max} \\ 1 & \text{if } E < E_{min} \end{cases}$$

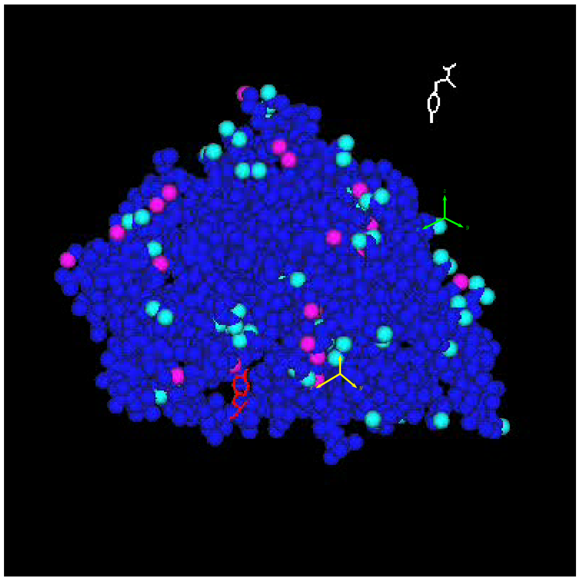


Results in denser distribution of nodes in low-energy regions of conformational space

Edge Generation

Each node is connected to its closest neighbors by straight edges Each edge is discretized so that between q_i and q_{i+1} no atom moves by more than some $\varepsilon = 1\text{Å}$.



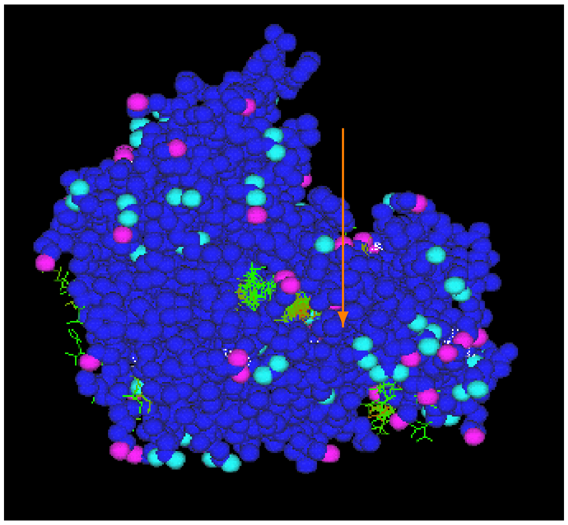


Results in denser distribution of nodes in low-energy regions of conformational space

Querying the Roadmap: For a given goal node q_g (e.g., binding conformation), the Dijkstras single-source shortest-path algorithm computes the lowest-weight paths from q_g to each node (in either direction) in $O(N \log N)$ time, where N = number of nodes Various quantities can then be easily computed in $O(N)$ time, e.g., average weights of all paths entering qg and of all paths leaving q_g (binding and dissociation rates K_{on} and K_{off})

Computing Binding Conformations Sample many (several 1000s) ligand's conformations at random around protein Then repeat several times:

- Select lowest-energy conformations that are close to protein surface
- Re-sample around them
- Retain k (approx. 10) lowest-energy conformations whose centers of mass are at least 5Å apart



lactate dehydrogenase

Related: Finding Folding Pathways Using RPM The degrees of freedom are modeled as the number of rotatable backbone dihedral angles (approx. 2N, number of amino acids) Nodes are generated in a similar manner as the docking scheme above. Sampling cannot be done at random due to high dimensionality – sampling is done from a set of distributions around the native state. Edges connect neighboring nodes in a similar manner to the one described above. This method can be used to discover folding pathways, intermediate structures and other folding events.

From Flexible Ligand to Flexible Receptor?

The target receptor is big. It may have many DOFs (in the thousands) Most methods that try to model receptor flexibility try to somehow find and focus only on relevant motions. In order for this process to become efficient, we must find a representation for protein flexibility that avoids the direct search of a solution space comprised of thousands of degrees of freedom. There are several methods available, and the accuracy of the results is usually directly proportional to the computational complexity of the representation. There are several types of methods to model receptor flexibility:

Soft Receptor Soft receptors can be easily generated by relaxing the high VdW energy penalty The rationale is that the receptor structure has some inherent flexibility which allows it to adapt to slightly differently shaped ligands. If the change in the receptor conformation is small enough, it is assumed that the receptor is capable of such a conformational change. It is also assumed that the change in protein conformation does not incur a sufficiently high energetic penalty to offset the improved interaction energy between the ligand and the receptor. It is also quite easy to implement (relax the collision component).

Selecting Specific DOFs It is it possible to select only a few degrees of freedom to model explicitly. These degrees of freedom usually correspond to rotations around single bonds These

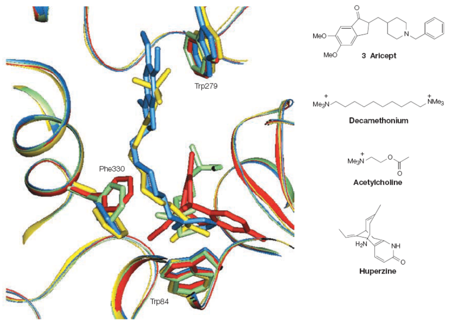


Figure 5 | **Multiple conformations of a single residue.** Overlay of native and three *Torpedo californica* acetylcholinesterase–ligand complexes using the protein C-α atoms (Protein Data Bank codes 2ACE, 1EVE, 1VOT and 1ACL). Key protein side chains are indicated by thick lines, as are the inhibitors. The colour codes are: donepezil (Aricept, blue), decamethonium (yellow), native (green), huperzine (red). The flexibility of Phe330, in comparison with the rigidity of the rest of the gorge, is highlighted.

Figure 7: Acetylcholinesterase: Phe330 is flexible and acts as swinging gate

degrees of freedom are usually considered the natural degrees of freedom in molecules. Rotations around bonds lead to deviations from ideal geometry that result in a small energy penalty when compared to deviations from ideality in bond lengths and bond angles. Selection of which torsional degrees of freedom to model is usually the most difficult part of this method because it requires a considerable amount of a priori knowledge regarding the binding site. The torsions chosen are usually rotations of side chains in the binding site of the receptor protein. It is also common to further reduce the search space by using rotamer libraries.

Ensemble Docking One possible way to represent a flexible receptor for drug design applications is the use of multiple static receptor structures The best description for a protein structure is that of a conformational ensemble of slightly different protein structures coexisting in a low energy region of the potential energy surface. The structures can be determined experimentally either from X-ray crystallography or NMR, or generated via computational methods such as Monte Carlo or MD simulations.

Receptor Flexibility – Collective DOF Finally, collective DOF allows the representation of full protein flexibility without a dramatic increase in computational cost. One method is the calculation of normal modes for the receptor. See Figure 8 for an example of the normal mode analysis of HIV-protease. Alternatively, we can use dimensionality reduction methods. The most commonly used method for the study of protein motions is principal component analysis (PCA).

5 Inverse Kinematics

Inverse kinematics (IK) is the problem of finding the right values for the underlying degrees of freedom of a chain. In the case of a protein chain these degrees of freedom of the dihedral angles, so that the chain satisfies certain spatial constraints. For example, in some applications, it is necessary to find rotations that can steer certain atoms to desired locations in space. To achieve a particular function, protein regions sometimes have to undergo concerted motion where atoms move together in order to locate themselves near another protein or molecule. The motion of atoms is spatially constrained because they have to assume specific target locations in space. However, since

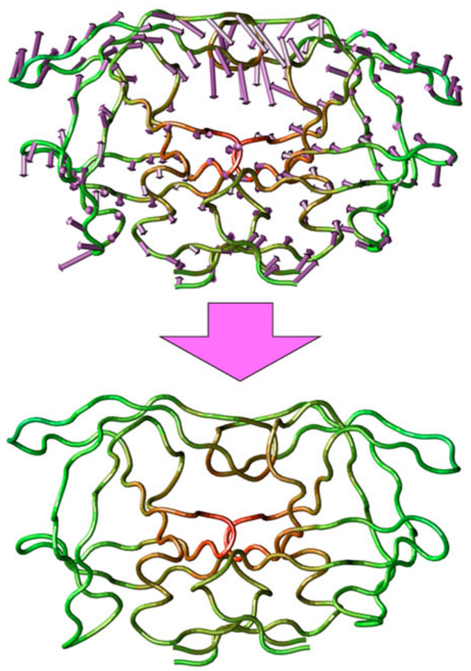


Figure 8: Collective DOF analysis of HIV-protease

atoms must move together in order not to break bonds by their motion, it is easier to model their motion in dihedral angle space, where bond lengths and bond angles are fixed. The applications of inverse kinematics to protein structure include mainly loop modeling and generating ensembles of structures.

Modeling Loops Using Inverse Kinematics (IK)

Goal: Model the ensemble of conformations of a protein. It is known that proteins are not rigid but fluctuate about an ensemble of structures under equilibrium conditions. Focus mostly on loop regions, as they are the most flexible ones.

As mentioned in the introduction, Inverse kinematics is used to manipulate the degrees of freedom of an articulated chain to satisfy some end-constraints. In this case - manipulate the rotational degrees of freedom of a loop region to find possible loop conformations that attach to the rest of the protein.

Cyclic Coordinate Descent (CCD)

We solve for and rotate one dihedral at a time. CCD tries to find an optimal angle by which to rotate a single bond at a time, so as to steer the desired atoms towards its target position. When finding dihedral angles to rotate so that the ends of the fragment connect properly with the rest of the chain, it is important to steer not just one atom of the end of the fragment, but the three backbone atoms of the end simultaneously. Finding values to the dihedral angles that steer the

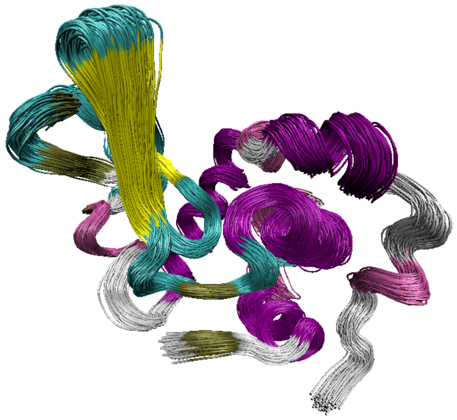


Figure 9: An ensemble of conformations modeled using IK

three backbone atoms of the end of the fragment simultaneously to their target positions guarantees that the end of the fragment will assume both its target position and orientation in space. We will explain how to find optimal values to simultaneously steer the three backbone atoms of the end of the fragment to their target positions. Let us first define the current positions before rotation as M_0, after rotation M and the target positions F, as shown in Figure 10. The goal is to minimize the Euclidean distance between the current and the target positions for all three atoms simultaneously. Figure 11 illustrates the process on a protein chain. In order to find the optimal angle by which to rotate a particular bond, let us define an objective function S that we wish to minimize. We propose a value for S that sums the square of the deviations between the final position of the atoms after the rotation M and the desired positions F. Looking at Figure 10 Since S is defined as the sum of squared distances between current positions and target positions, steering these three atoms to their target positions requires minimizing S. Therefore, the optimal dihedral rotation can be found by minimizing S:

$$S = |\vec{F}_1 M_1|^2 + |\vec{F}_2 M_2|^2 + |\vec{F}_3 M_3|^2$$

Where

$$\vec{F}_1 M_1 = \vec{O}_1 M_1 - \vec{O}_1 F_1$$

Explanation: The FM vectors can be defined relative to an origin O located along the axis of rotation, which will simplify the math, since the rotation is in 2D when working on the plane perpendicular to the axis of rotation. O can be computed by projecting the current position of the atom, M, onto the rotation axis. It is convenient to decompose OM for each feature atom into two components (along the $\hat{r}$ and $\hat{s}$ local axes). The squared norm of the vector $M - F$ (denoted FM) has precisely this value for each of the three atoms, so we can sum the three contributions to S in order to allow its expression in terms of the angle being rotated (using cosine and sine). This way, the distance between the atoms and their target positions will be only a function of the fixed (rotatable bond) atoms and the angle to rotate, which remains the only variable and the problem can be solved. We can express the rotation with respect to the $\hat{r}$ and $\hat{s}$ plane as:

$$\vec{O}_1 M_1 = r_1 \cos\theta \hat{r}_1 + r_1 \sin\theta \hat{s}_1$$

Where r_1 is the vector between O and M_{01}, which we want to rotate by θ. From the two equations above it follows that:

$$\vec{F}_1 M_1 = r_1 \cos\theta \hat{r}_1 + r_1 \sin\theta \hat{s}_1 - \vec{f}_1 \equiv \vec{d}_1$$

with similar equations for the two other atoms. Calculating the squared distances between the moving atoms and the fixed target atoms, we obtain:

$$\begin{aligned}
|\vec{d}_1|^2 &= r_1^2 + f_1^2 - 2r_1 \cos\theta(\vec{f}_1 \cdot \hat{r}_1) - 2r_1 \sin\theta(\vec{f}_1 \cdot \hat{s}_1) \\
|\vec{d}_2|^2 &= r_2^2 + f_2^2 - 2r_1 \cos\theta(\vec{f}_2 \cdot \hat{r}_2) - 2r_1 \sin\theta(\vec{f}_2 \cdot \hat{s}_2) \\
|\vec{d}_3|^2 &= r_3^2 + f_3^2 - 2r_1 \cos\theta(\vec{f}_3 \cdot \hat{r}_3) - 2r_1 \sin\theta(\vec{f}_3 \cdot \hat{s}_3)
\end{aligned}$$

Putting it all together, we can express S as the sum of the squared distances above. Differentiating with respect to θ gives us:

$$\frac{dS}{d\theta} = \frac{d|\vec{d}_1|^2}{d\theta} + \frac{d|\vec{d}_2|^2}{d\theta} + \frac{d|\vec{d}_3|^2}{d\theta}$$

where

$$\frac{d|\vec{d}_i|^2}{d\theta} = 2r_i \sin\theta(\vec{f}_i \cdot \hat{r}_i) - 2r_i \cos\theta(\vec{f}_i \cdot \hat{s}_i)$$

where $i = 1, 2, 3$. The $r_i^2 + f_i^2$ part of the expression are not a function of θ and therefore their derivative is 0.

This is just a first order differential equation that we can equate to 0 to find α, the value of the rotation angle that yields an extreme value for the equation above:

$$\tan\alpha = \frac{(\vec{f}_1 \cdot \hat{s}_1)r_1 + (\vec{f}_2 \cdot \hat{s}_2)r_2 + (\vec{f}_3 \cdot \hat{s}_3)r_3}{(\vec{f}_1 \cdot \hat{r}_1)r_1 + (\vec{f}_2 \cdot \hat{r}_2)r_2 + (\vec{f}_3 \cdot \hat{r}_3)r_3}$$

Inverting the tangent will produce two values for α that are 180° apart. One of them is a minimum and one a maximum. The right one is that which produces a positive value of the second derivative of S. In practice this is a bit cumbersome, but α is obtained in a different way. Notice that S is of the form:

$$S = a - b\cos\theta - c\sin\theta$$

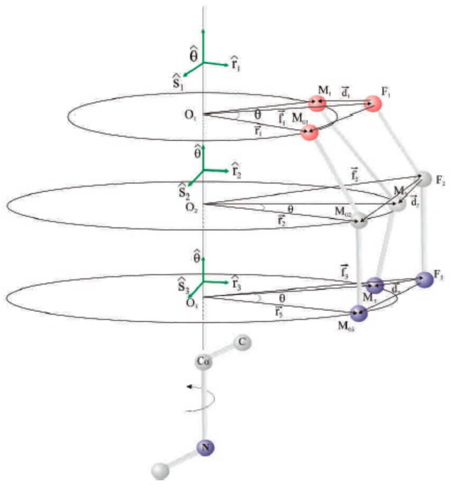


Figure 10: An illustration of the CCD algorithm: Find optimal dihedral rotation for the current bond so that all three desired atoms reach their target positions.

Multiplying the last two terms by

$$\sqrt{b^2+c^2}/\sqrt{b^2+c^2}$$

we get:

$$\cos\alpha = \frac{b}{\sqrt{b^2+c^2}}$$
$$\sin\alpha = \frac{c}{\sqrt{b^2+c^2}}$$

then S can be written as:

$$S = a - \sqrt{b^2+c^2}\cos(\theta-\alpha)$$

S is minimum when $\theta = \alpha$. Now we have explicit values for sine and cosine. Notice that the Time complexity is linear time on the number of DOFs to solve for all dihedrals of a chain.

Figure 12 shows an illustration of the process for Chymotrypsin inhibitor 2.

Equilibrium Fluctuations

Since there is redundancy (more degrees of freedom than constraints), there is a continuum of possible solutions. In order to find physically feasible conformations. We can combine the conformational sampling with energy minimization to obtain an ensemble of physical structures. This can be exploited to generate fragment fluctuations. We can generate the Boltzmann ensemble average based on the energy with the following formula:

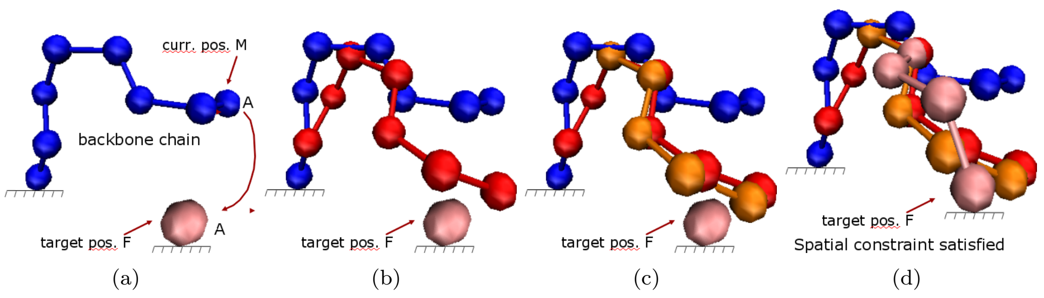


Figure 11: Demonstration of the CCD algorithm to close a loop.

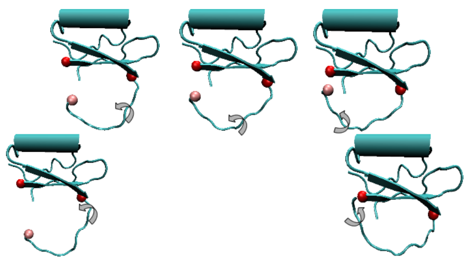


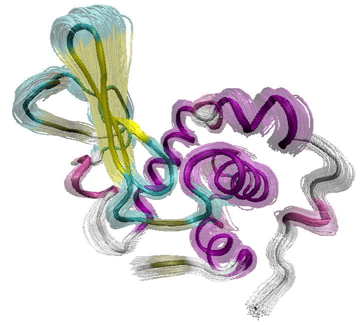
Figure 12: Example: Chymotrypsin inhibitor 2

$$RMSD_x = \sum_{Confs} \frac{RMSD(C, C_{native})e^{-\beta\Delta E_c}}{Q}$$
$$\Delta E_c = E_c - E_{native}$$
$$Q = \sum_{Confs} e^{-\beta\Delta E_c}$$

Figure 13 shows two examples: α-Lactalbumin, 123 residues and Ubiquitin, 76 residues.

Additional Reading

- A.P. Singh, J.C. Latombe, and D.L. Brutlag. A Motion Planning Approach to Flexible Ligand Binding. Proc. 7th ISMB, pp. 252-261, 1999
- Cyclic Coordinate Descent: Canutescu A. A., and Dunbrack R. L. Protein Science 12, 2003


(a)

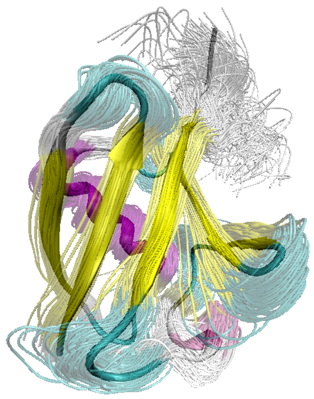

(b)

Figure 13: a) α-Lactalbumin (α-Lac) b) Ubiquitin