GoMol: A 3D Protein Analysis & Visualization Tool

Tyler Katz, Shivank Sadasivan, Darin Boyes, Minhyek Jeon

Carnegie Mellon University

02-601: Programming for Scientists

Dr. Phillip Compeau

29 November 2023

GoMol: A 3D Protein Analysis & Visualization tool

Introduction

In 1957, Francis Crick described the central dogma of biology as a framework in which biological information flows from DNA to RNA to proteins. Proteins are large, complex molecules that are responsible for much of the complexity of life. 20 types of amino acids, encoded in DNA and RNA, serve as the building blocks of proteins, connecting in long chains and folding in a specific manner that gives rise to its function. The primary structure of a protein includes the sequence of amino acids that comprise the protein and the secondary structure confers the local folding within a protein due to backbone atom interactions, with the most common example being α-helices and β-sheets. The tertiary structure thus confers the overall three-dimensional structure of the protein and the quaternary structure involves the conglomeration of multiple protein subunits. Proteins play important roles in the structure, function, and regulation of biological processes. Disease can thus be thought of as a disruption in the structure or function of these biological processes on a molecular or cellular level. Understanding the mechanisms behind protein assembly and function is a crucial step in controlling biological processes and developing treatments for disease. With an ever increasing amount of biological data being generated from advancing experimental technologies, the field of computational biology has aimed to apply computational techniques to extract insight from these vast data sets. The ability to predict protein structures and efficiently analyze and visualize them is therefore an important area of research, as the time consuming and expensive process of designing small molecule drugs for a protein target can be made more efficient with computational tools. The comparison of proteins in meaningful ways is an interesting problem facing this field, as insight can be extracted and applied from one protein to another similar protein. Additionally, recent applications of artificial intelligence and generative technologies have created a need to compare computationally predicted protein structures to experimentally determined structural data. The RCSB Protein Data Bank includes structural data of proteins for over 200,000 experimentally determined structures and over 1,000,000 computed structure models from Alpha Fold DB and Model Archive. To address this need for computational tools to compare and visualize proteins, our group has developed GoMol, a software tool, written in Go, that can aid scientists in aligning primary structures of proteins, aligning three-dimensional structures of similar proteins, and visualizing the three-dimensional structure of proteins from publicly available PDB data.

Project Components

Sequence Alignment

To compare and to understand the nuances of two protein sequences which are evolutionarily related, we decided to implement the Needleman Wunsch algorithm in Go. Needleman Wunsch algorithm is a global sequence alignment algorithm which uses dynamic programming to align two protein sequences which have a common ancestor and are similar in length. It runs with a time complexity of O(nm), where m and n are the length of the sequences. BLOSUM62 scoring matrix was used to assign the matches and mismatches score along with gap penalty. By this approach we get to identify the conserved sequences which are crucial for understanding molecular function, and also help us identify potential drug targets by comparing the sequences of proteins from different variants. The objective here is to align two sequences to get insights on their similarity, functions and evolutionary relationship. Based on their alignment scores, we move ahead to understand their 3D structure through visualization and 3D structure alignment.

3-D Structural Alignment

To accomplish the goal of comparing the three-dimensional structures of two similar proteins, our group opted to implement the Kabsch algorithm. This algorithm leverages linear algebra to mathematically calculate the optimal rotation matrix that minimizes the root mean squared deviation (RMSD) from one set of points to another. This algorithm is useful because given the three-dimensional coordinates of two similar proteins, as found in their PDB files, the rotation matrix found by the Kabsch algorithm will rotate the points of the first protein in a manner that best aligns them with the points of the second protein. To implement this algorithm in Go, the gonum mat package was used for matrix math. The first step is to calculate the covariance matrix, H, by multiplying one set of coordinates by the transpose of the other. After this step, the singular value decomposition of this matrix is then calculated, yielding results in the form $H = U\Sigma V^T$. The optimal rotation matrix can easily be computed from this point by multiplying the matrices U and V^T. Now that we can mathematically rotate one protein to be three-dimensionally aligned in a manner that minimizes the RMSD, we can visualize them in a way that highlights or emphasizes the differences in structure that may not have been intuitive when looking at the two proteins from different vantage points.

Visualization

Our group has opted for a simple, but robust approach to visualizing proteins in three dimensions. Understanding how to go about modeling objects in three dimensions has been a long-standing issue in computer graphics, and here we make use of some of the most common strategies that have been used in the rendering of 3D objects for decades. The main rendering technique that we have implemented in GoMol is ray tracing, where we try to simulate the way that light interacts with objects in a virtual environment. It works by first specifying a camera, which will serve as the point of origin which all rays will be cast from. The direction of each ray is specified by spaces representing individual pixels in a viewport, where all of the area within the viewport is visible to the user. We can imagine this as a camera being located behind a viewport, and being pointed at the center of mass of our protein of interest. We can also think of each pixel on our screen as having a ray casted through it from the camera and interacting with objects in the scene. Then, we define a light source as a point light, which emits light uniformly in all directions. This piece will be important when rendering the proteins, because the color of individual atoms in the protein will be affected by factors such as distance from the light source, the light intensity, and various attenuation coefficients. Finally, we specify our object, which will be all atoms of a protein loaded from a PDB file. Each atom in a PDB file has a significant amount of information associated with it including its index, element, amino acid, amino acid index, and x, y, and z coordinates. We make use of this information, in addition to information about the Pauling radii of individual elements to produce an accurate representation of the protein in 3-dimensional space. To accomplish the rendering of the protein, we use the previously described ray tracing technique to cast rays into the scene and check for collisions with any of the atoms in the protein. This can be done using very simple vector math. Once we determine that there has been a collision, we take the point of intersection between the ray and the atom, and compute a color for that point based on a well-defined shading algorithm. The shading algorithm that we had opted for is called Phong shading, in which each atom is shaded based on some initial color, and its diffuse, ambient, and specular properties. Diffuse lighting essentially colors an object based on its orientation to the light, meaning that certain points that are more distant from the light source will be shaded darker, while closer points will be shaded lighter. Ambient light is just a uniform light factor that is assigned to the entire scene. This does not change with respect to the object's orientation or surface normals. Finally, specular lighting serves to provide specular highlights, which highlights points on the object with respect to the direction of the camera. All three of these factors combined serve to provide a realistic model of how light interacts with objects in the real world. Finally, we have implemented some quality of life improvements, including the ability to rotate the protein by clicking and dragging, zooming the camera in and out using the scroll wheel, moving the camera using the WASD keys, and options for coloring the protein based on the atom elements, side chains, and similar regions determined from sequence alignment or 3D structure alignment.

Graphical User Interface

For our Graphical user interface, we are considering 3 parts. First, UI elements, options for customization, interactive features and lastly integration of sequence alignment and 3D structural alignment results.

Firstly, for Buttons, Menus, or Other UI Elements, these elements serve as user triggers to initiate specific actions within the GoMol application. Sequence Alignment function starts the sequence alignment process, where users can input or select two protein sequences for comparison. Secondly 3D Structural Alignment function Initiates the 3D structural alignment process, allowing users to input or select two protein structures for comparison. Lastly, Customization Menus Dropdown menus or options for users to customize the visualization, such as choosing color schemes for atoms or sequence alignment results. Other UI Elements we can include are Additional buttons or controls for specific actions, like loading protein structures, saving results, or accessing help/documentation.

Options for Customization provides users with the ability to customize the visual representation of proteins based on their preferences or analysis requirements. Options can include color schemes, rendering styles and highlighting features.

For Color Schemes, users can choose different color schemes for atoms based on their elements (e.g., oxygen, nitrogen) or customize colors based on sequence alignment results. For Rendering Styles, it is options to change the rendering style of proteins, such as wireframe, solid, or ribbon representation. Lastly, for Highlighting Features, it enables to highlight specific regions of interest, such as conserved sequences or areas identified through sequence alignment.

Interactive Features enhances user experience by providing interactive controls for manipulating the visualization. Functionality we can include is rotation controls, zooming controls, camera movement. For Rotation Controls, users can rotate the protein structure by clicking and dragging with the mouse. For Zooming Controls, users can Zoom in and out using the scroll wheel on the mouse. Lastly for Camera Movement user can control Movement using keyboard shortcuts (e.g., WASD keys) for changing the viewpoint and exploring different angles.

Integration of Sequence Alignment and 3D Structural Alignment Results ensures a cohesive user experience by seamlessly integrating the results of sequence alignment and 3D structural alignment. Option for results can include overlay visualization, linked views and result information. For Overlay or Side-by-Side Visualization, it can display both the aligned sequences and the aligned 3D structures simultaneously for easy comparison. For the linked Views, changes in one view (sequence or 3D structure) are reflected in the other, allowing users to understand the relationship between sequence similarities and structural alignments. Lastly for the result Information it can Display alignment scores, conserved sequences, or any relevant information derived from the alignment processes.

By incorporating these key elements, the GUI of GoMol aims to provide a user-friendly and comprehensive environment for scientists to analyze and visualize protein structures.

Results

Discussion

References

Potential/Eventual sources to site

- https://en.wikipedia.org/wiki/Central_dogma_of_molecular_biology
- https://medlineplus.gov/genetics/understanding/howgeneswork/protein/
- https://biology.indiana.edu/undergraduate/biology/biology-bs/areas-of-concentration/disease.html#:~:text=In%20simple%20terms%20a%20disease,a%20molecular%20and%20cellular%20level.
-