Susan Rui Ling
2425170
BIOEN537 Software Project
12/09/24

Component Specifications

Overview:

The ProteinStructureVisualizer is a Python-based tool that provides 3D visualization of monomeric protein structures from PDB files. It provides a GUI for visualizing protein structures as point clouds and backbone graphs.

Components:

File and Data Processing

1. PDB Parser (parse_pdb): loads and parses a PDB file to extract protein data, organizing info on ATOM and HETATM records into a Pandas DataFrame.
    - Input: PDB file path
    - Output: Pandas DataFrame containing atomic coordinates and optional header.

2. Alternative Location Processor (remove_altloc): PDB files tend to have atoms that are assigned alternative locations. In order to visualize them, the DataFrame from above is processed to remove these and assign them to a single location.
    - Input: Parsed PDB DataFrame
    - Output: Cleaned DataFrame with single alt_loc assigned to each atom.

3. Graphein Object Creator (create_graphein_object): In order to create 3D backbone visualizations, this tool utilizes the Graphein package. To use this package, a Graphein object must first be initialized.
    - Input: Cleaned DataFrame with alt_loc processed, Raw PDB DataFrame, PDB Code, Granularity (defaults to alpha carbon)
    - Output: Graphein object

Visualization Components

1. Atom Point Cloud Visualizer (plot_atom_point_cloud): uses processed DataFrame to produce a 3D atom point cloud visualization of the protein. Utilizes the Plotly package for visualization. Allows for atom-based colouring, 3D coordinate plotting and interactive zoom and rotation.
    - Input: Processed DataFrame
    - Output: Interactive 3D HTML plot

2. Backbone Graph Visualizer (plot_backbone_graph): uses Graphein object to produce a 3D backbone graph. Allows for visualization of backbone connectivity, residue position colouring and interactive structure manipulation.
    - Input: Processed DataFrame
    - Output: Interactive 3D HTML plot

GUI Interface

1. Main Window (create_main_window): establishes and configures the main GUI application window.

Components:

- File selection button
- Visualization control buttons
- Error handling dialogs

Features:

- Clean exit handling
- Responsive layout

2. GUI Process PDB File (process_pdb_file): processes the input PDB file in the GUI and allows for visualization of the atom point cloud or backbone graph.

Technical Specifications

Dependencies

- pandas
- plotly
- networkx
- biopandas
- prody
- graphein

GUI Libraries:

- **tkinter**

File Outputs

HTML files:

- **point_cloud.html: 3D atomic visualization**
- **backbone_graph.html: 3D backbone structure**

Susan Rui Ling
2425170
BIOEN537 Software Project
12/09/24

Interactions to accomplish use cases:

a) The user would load a PDB file which will then parse the protein data into a data object.

To accomplish this use case, the user will go to RCSB Protein Data Bank (RCSB PDB), search and click into their protein of interest. The user will then download the PDB Format file. Using the "Load Protein Structure" function, the user will load in their PDB file from the "Select PDB File" button on the GUI.

b) The user would choose to view the protein's atom point structure.

To accomplish this use case, the user would click the "View Point Cloud" button on the GUI. This will then display the graph on the user's default browser.

c) The user would choose to view the protein's 3D backbone.

To accomplish this use case, the user would click the "View 3D Backbone Graph" button on the GUI. This will then display the graph on the user's default browser.

Preliminary Plan:

1. Project Initialization: set up a Git repository and define a clear project structure

including: /docs, /src, /tests.

2. For the File and Data Processing component, implement a function to load and parse PDB files and extract primary and secondary structure information. Build a dataframe for accessing protein data including chains, residues, and atomic coordinates.

3. For the Visualization component, implement a function that generates a 3D representation of the atoms as a point cloud. Add function for basic colouring based on residue type. Create another function to represent the protein as a backbone graph. Add function for highlighting specific residues. Add basic interactive capabilities and colouring by residues.

5. For the GUI component, configure a simple GUI that allows the user to visualize their desired PDB file by clicking a button and opening the graphs in their default browser. Use tkinter for the GUI development.

6. Write unit tests for each function and test the package with various PDB files. Debug any errors.

7. Finalize documentation, pyproject.toml, __init__.py, README and .gitignore files and upload to PyPI for user installation. Test using *pip* install.