This file contains example data that was analyzed from the full trajectory described in (Liu et al., 2023).

The data here only represents the first few ns of a 200 ns run. The goal of this example is to allow users to run the code that was generated to analyze the files on a much smaller and user-friendly set of data. The data will not look the same as the manuscript as a result.

This folder should have the following contents:

1. Read\_me.docx
   1. This document contains instructions on how to run the code
2. code
   1. Chain\_distances.py
      1. File used to analyze the trajectory in UCSF Chimera
   2. Distance\_Analysis.py
      1. File used to analyze the outputs from the trajectory analysis
3. MD\_files
   1. run\_input.tpr
      1. topology and run parameters for the MD simulation
   2. trajectory.xtc
      1. trajectory file for the MD simulation. This is a significantly shorter version than what was used in (Liu et al., 2023).
4. MDP
   1. equilibration.txt
      1. specific conditions for the equilibration step, generated from CHARMM-GUI
   2. minimization\_1.txt
      1. specific conditions for the first minimization step, generated from CHARMM-GUI
   3. minimization\_2.txt
      1. specific conditions for the second minimization step, generated from CHARMM-GUI
   4. production\_short\_step.txt
      1. specific conditions for the first production step, generated from CHARMM-GUI
   5. production\_long\_steps.txt
      1. specific conditions for subsequent production steps, generated from CHARMM-GUI with the number of steps increased for longer simulation times
5. output
   1. example.txt
      1. example output file generated from Chain\_distances.py file. The file is analyzed using the Distance\_Analaysis.py file
   2. norm\_example.png
      1. an example image that will be generated from the code

Formatting notes

>>> is the symbol for a line of code to run. This could be a command line prompt or Python code.

> Button to press in software. For example, to turn your computer on it would be written:

>power

To use this file:

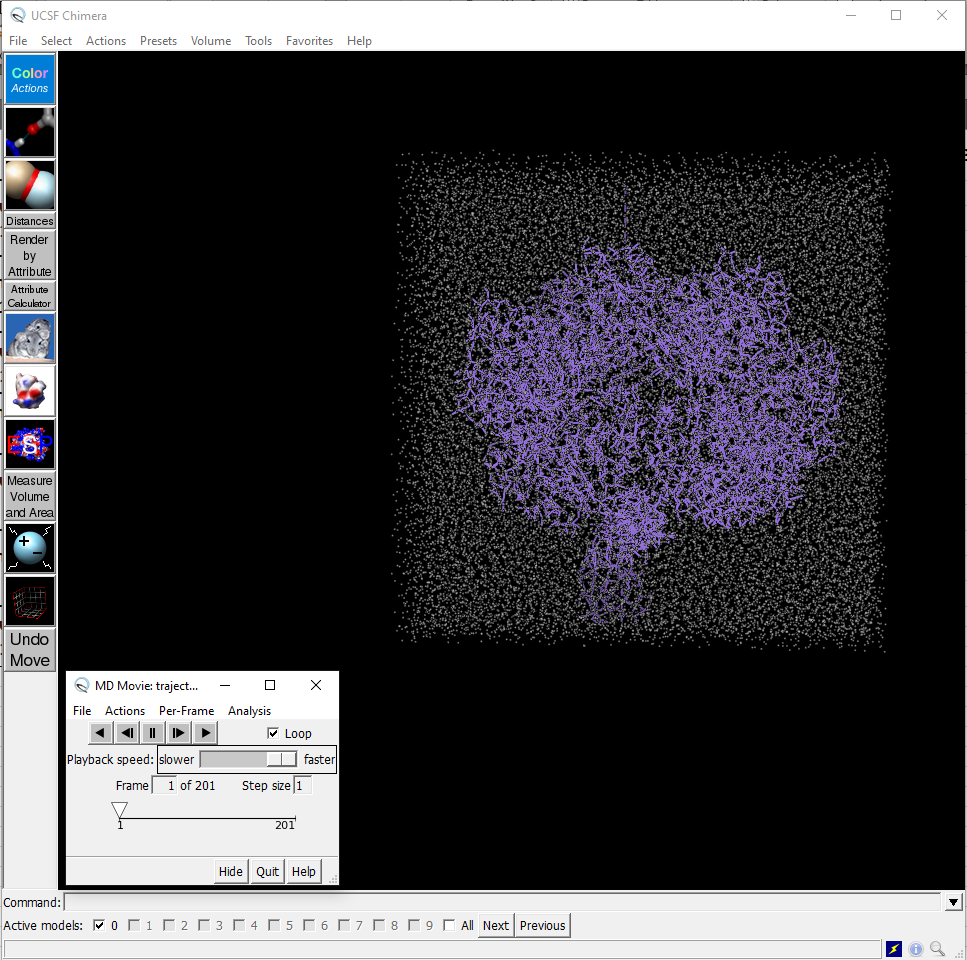
1. Install required software
   1. Chimera
      1. UCSF Chimera, version 1.15 was used in this project. This version is no longer supported by UCSF Chimera, which may lead to the analysis breaking if run on newer versions. This is especially true of analyzing files using Chimera X.
      2. <https://www.cgl.ucsf.edu/chimera/>
   2. Python
      1. Install Python. This was written in Python 3.8.12
      2. <https://www.python.org/>
   3. Anaconda
      1. Install Anaconda. This was written in Anaconda 22.9.0.
      2. <https://anaconda.org/>
   4. Spyder
      1. Install Spyder through the Anaconda platform. This was written using Spyder 5.2.1.
   5. Libraries
      1. It is recommended that you create a virtual environment in Anaconda before conducting any analysis. In the new environment, install pandas and Matplotlib.
      2. Pandas version 1.3.5
      3. Install pandas

>>> conda install pandas

* + 1. Install matplotlib 3.5.1

>>> conda install matplotlib

1. Analyze molecular dynamics trajectory
   1. Files required
      1. run\_input.tpr
      2. trajectory.xtc
      3. Chain\_distances.py
   2. Files generated
      1. Example.txt
   3. Place the required files in a location on your computer where you can access them.
      1. It may be convenient to make a .txt file with frequently used file paths
      2. In all cases below where PATH\TO\file\_name.type is used, it indicates you should replace that with the file path on your own device.
   4. Change the file path in the Chain\_distance.py file
      1. Change the 14th line of the Chain\_distance.py file to a folder where you want your MD simulation results to be saved. This can be done in Spyder or any other IDE that you prefer.
      2. If you are analyzing the MD files contained above, this program will give you the LAP residues that are within 5Å of internalin. If converting to a different trajectory, the code can be modified by changing the residues that are selected on lines 26 and 36 and renaming selections to more intuitive names based on your project in lines 28 and 32.
   5. Open the simulation in UCSF Chimera.
      1. Open UCSF Chimera
      2. Open the simulation
         1. >Tools > MD / Ensemble Analysis > MD Movie
      3. In the window that pops up use the following inputs:
         1. Trajectory format = GROMACS
         2. Run input (.tpr) = PATH\TO\run\_input.tpr
         3. Trajectory (.trr or .xtc) = PATH\TO\trajectory.xtc
         4. Use frames = first through =last
      4. This should open the simulation:



* + - 1. Your Chimera display may look different depending on what you have in the quick access toolbar on the left-hand side. The snip also shows the movie player in the window, which can be moved to any convenient location.
    1. Play the full movie. The file that you have opened is 201 frames (just the first few ns of the simulation from (Liu et al., 2023)). Playing the movie loads all of the frames for analysis.
  1. In the command line type “open”
     1. Use the browser window to select the Chain\_distances.py file
     2. When you click open in the browser on this file, it will start to run on the program. This may take a few minutes depending on the computer and the size of your simulation. It will appear that Chimera is frozen while the program runs. It is still running though, so be patient.
  2. If successful, a new file will be generated in the path with your output file named “example.txt” or another name if you choose to change the file name. This is the file that is analyzed with the next Python code in this example.

1. Analyze amino acids results from Chimera Distance Analysis
   1. Files required
      1. Distance\_Analysis.py
         1. This file is used to analyze individual amino acid counts from the trajectory and group them based on charge and hydrophobicity, and it does it for the interaction of B and C chains separately. It was written to count each amino acid type individually instead of in groupings by default since many combinations of amino acid groupings were considered during the original data exploration. This could be updated and made significantly better, but its current format is the result of the data exploration process.
   2. Files generated
      1. norm\_exmple.png
   3. Open Anaconda
   4. Open Spyder
   5. From Spyder, open Distance\_Analysis.py
   6. In this file, each cell is indicated by #In[CELL NAME]. A summary of cell functions follows. These cells can be run individually, or the whole program can be run to visualize different results from the analysis. What follows is a brief description of each cell.
   7. #In[IMPORT LIBRARIES]
      1. Read in the dependencies of this project including Pandas and Matplotlib
   8. #In[READ AND CLEAN THE DATA]
      1. This section reads in the file generated from the Chain\_distances.py analysis file
      2. It strips unwanted punctuation that is an artifact of the program that writes the data and assigns a time value to each step.
      3. The data format is now one row for each frame in the MD simulation and a series of columns that show which amino acids were within 5Å of internalin in the columns.
   9. # In[MAKING A DF OF A PARTICULAR CHAIN IF NEEDED]
      1. This looks at the interacting B and C chains separately. It may not be necessary for your project depending on the interactions you are interested in investigating.
   10. # In[MASKING RESIDUES TO PLOT SPECIFIC CATEGORIES]
       1. This section finds the counts of interacting residues for each time, by type, for each frame in the MD simulation.
       2. Amino acids are grouped into different categories including positive, negative, and hydrophobic (Kyte-Doolittle hydrophobicity > 0).
       3. Depending on the size of your trajectory, this may take some time to complete, as the process is conducted for each amino acid individually. It could be sped up by putting parts of this into functions and pre-grouping amino acid categories.
   11. # In[APPLY A ROLLING MEAN]
       1. Applies a rolling mean to the different interaction counts calculated above. Since this example data is fairly small, a rolling mean of 10 points was selected, which is different than what was originally published, where the rolling mean was 300 points.
   12. # In[GENERATE THE NORMALIZED VALUES]
       1. This section normalizes that counts to their frequencies in the B and C chains. The chain length and amino acid / amino acid type counts are the sum of those amino acids from the B and C chains in the LAP model.
   13. # In[PLOTTING INDIVIDUAL INTEACTIONS- COUNTS]
       1. This section shows how the individual groupings of amino acids can be plotted. These can be updated to fit your project needs.
   14. # In[PLOTTING ALL INTERACTIONS- COUNTS]
       1. This section plots all of the counts of amino acids on top of one another. It can be seen that there is not much distinction between interactions.
   15. # In[PLOTTING ALL INTERACTIONS- NORMALIZED]
       1. This section plots the normalized data. This shows that there is a distinction between interacting amino acids when they are normalized to their frequencies within the chains.
       2. This figure is also saved as an example.