Integrating Industry Standard Rendering Techniques for Visualizing Solvation in Molecular Dynamics Simulations Using The MolecularNodes Software Plug-in

Joyce Kim[1]
External Project Supervisor: Dr. Jessica Nash[2]
Collaborator: Orion Cohen[3]

[1] Molecular Science and Software Engineering Master's Student, Department of Chemistry, University of California, Berkeley.
[2] The Molecular Science Software Institute
[3] Graduate Research Fellow, Materials Science and Engineering, Persson Group, Berkeley Lab

Abstract

MolecularNodes is a Blender plugin that offers advanced rendering for molecular dynamics (MD) simulations.[4] Unlike other visualization softwares, Blender employs industry-standard rendering techniques commonly used in film and video games. However, MolecularNodes is not able to natively display solvent interactions, which are often crucial in elucidating molecular structure-function relationships. To address this, this project integrated SolvationAnalysis, a plugin that provides data structures for examining solute-solvent interactions, to MolecularNodes, enabling users to easily specify and visualize solute and solvent structures. This expansion of MolecularNodes' functionality is expected to appeal to researchers, students, and non-specialists because it allows users to produce professional-quality visualization of solvent interactions without needing sophisticated programming expertise.

Introduction

Protein function defects are related to various diseases and pathologies. Understanding protein dynamics is, therefore, essential for comprehending their functions. Molecular dynamics (MD) simulations utilize computational methods to simulate the movement of proteins and other molecules in their native environments, and export the dynamics of these simulations to data files termed "trajectories."[5] These simulations can also be applied to various systems in organic and inorganic chemistry, as well as solid-state chemistry. One crucial challenge in MD simulation practice is visualizing and understanding the interactions between solutes and solvents. This is made particularly difficult by the large amount of solvent molecules in a typical MD simulation, and the complicated visuals this often produces. Solvation structures in MD simulations are complex, but can often be understood by analyzing the solvent molecules that are closest and most durably associated with particular solutes, a structure often termed a solvent "shell."[6]

This project aimed to develop a new functionality in the existing Blender[7] plugin MolecularNodes to enhance visualization of MD simulations' solute-solvent interactions. This functionality enabled production of professional-quality images that require no programming background to generate, making them accessible to students and non-specialists. The proposed functionality will enable robust and efficient depiction of solvent-solute interactions and help users better understand the coordination and localization of solvent molecules in MD simulations.

Existing visualization python package options have limited ability to depict solvent interactions and lack the ability to highlight and image specific solute-solvent interactions. Although visualization softwares such as NGLView Jupyter Widget[8] enable simple visual

[4] B. Johnston, D. Marson, and Y. Yao, "Bradyajohnston/molecularnodes: V2.3.0 for Blender 3.4.1+," Zenodo, 26-Jan-2023. [Online]. Available: https://doi.org/10.5281/zenodo.7572290. [Accessed: 06-Feb-2023].
[5] I. V. Likhachev, N. K. Balabaev, and O. V. Galzitskaya, "Available Instruments for Analyzing Molecular Dynamics Trajectories," *The Open Biochemistry Journal*, vol. 10, no. 1, pp. 1–11, Mar. 2016, doi: https://doi.org/10.2174/1874091x01610010001.
[6] S. Han, "Structure and dynamics in the lithium solvation shell of nonaqueous electrolytes," *Scientific Reports*, vol. 9, no. 1, Apr. 2019, doi: https://doi.org/10.1038/s41598-019-42050-y.
[7] B. Foundation, "Home of the blender project - free and open 3D creation software," *blender.org*. [Online]. Available: https://www.blender.org/. [Accessed: 21-Feb-2023].
[8] "nglview," Digital Object Identifier System. [Online]. Available: https://doi.org/10.1093/bioinformatics/btx789. [Accessed: 06-Feb-2023].

inspection, they often lack a robust rendering capacity. Blender[9], an open-source modeling program, is uniquely well-positioned to support MD solvation visualization, with the Blender[10] plugin MolecularNodes[11] facilitating advanced and computationally-efficient rendering techniques. In the past, Blender has proven itself to be a good platform for MD visualization, including in the case of Pyrite, a Blender plugin that enables users to import general MD trajectories in a compact and memory-efficient form. MolecularNodes provides a method for quick 1-button import of a range of formats, including MD trajectories, and supports every feature of Pyrite, but with an enhanced user-friendly aspect.[12]

The proposed work had two-pronged approach that entailed: development of a user-friendly implementation of SolvationAnalysis, allowing users to input solvent and shell information at load time and an alternative but, more flexible approach where the user loads an MD trajectory with solvent information and can dynamically select the shell and frame. The second approach was not pursued primarily because it would have entailed altering MolecularNodes, drastically increasing development time. Furthermore, altering the original open source code may have compromised the integrity and sustainability of the original codebase. If the original code is modified and not properly documented, it can become difficult to maintain, update, and contribute to the software. Additionally, if the modifications are not made in a way that is consistent with the original codebase[13], it can create compatibility issues and hinder the ability of other developers to use and contribute to the software.

Rather than altering the original MolecularNodes code, we created a separate module that calls on a key function that creates the Blender objects used in the MD Trajectory module. This approach allows for greater flexibility and adaptability while also preserving the original codebase, and is therefore the strategy that was implemented.

Approach

In order to create a user-friendly implementation, we designed an user-interface(UI) that allows users to input solute selections and solvent group selections, with the latter specifying the number of solvents involved in that particular interaction. In order to accomplish this in an efficient and robust manner, we took advantage of two pre-existing libraries for analyzing MD data in MolecularNodes. The first library is called MDAnalysis[14], which provides data structures for parsing and analyzing MD trajectories, while the second library is an offshoot called SolvationAnalysis, which extends these same data structures to solute and solvents. This implementation enables users to select an individual solvation shell from all the shells present for a given frame through a drop-down menu in the named attribute panel. Leveraging these

[9] B. Foundation, “Home of the blender project - free and open 3D creation software,” *blender.org*. [Online]. Available: https://www.blender.org/. [Accessed: 21-Feb-2023]
[10] B. Foundation, “Home of the blender project - free and open 3D creation software,” *blender.org*. [Online]. Available: https://www.blender.org/. [Accessed: 21-Feb-2023]
[11] B. Johnston, D. Marson, and Y. Yao, “Bradyajohnston/molecularnodes: V2.3.0 for Blender 3.4.1+,” Zenodo, 26-Jan-2023. [Online]. Available: https://doi.org/10.5281/zenodo.7572290. [Accessed: 06-Feb-2023].
[12] Durrant JD. BlendMol: advanced macromolecular visualization in Blender. Bioinformatics. 2019 Jul 1;35(13):2323-2325. doi: 10.1093/bioinformatics/bty968. PMID: 30481283; PMCID: PMC6596883.
[13] B. Johnston, D. Marson, and Y. Yao, “Bradyajohnston/molecularnodes: V2.3.0 for Blender 3.4.1+,” Zenodo, 26-Jan-2023. [Online]. Available: https://doi.org/10.5281/zenodo.7572290. [Accessed: 06-Feb-2023].
[14] “MDAnalysis: A python package for the rapid analysis of molecular ...” [Online]. Available:https://www.researchgate.net/publication/328774554_MDAnalysis_A_Python_Package_for_the_Rapid_Analysis_of_Molecular_Dynamics_Simulations. [Accessed: 13-Apr-2023].

libraries, we created a mechanism for visualizing solvation that does not require programming knowledge, and integrates well with existing modules.

Software dependencies

The software components that we built are primarily centered around the “MD Solvation Shell” tab in the user interface (Fig 1). The purpose of this tab is to provide users with an easy and intuitive way to analyze protein solvation in the context of MD simulations. The project's software library dependencies include Blender[15] and MDAnalysis[16], with SolvationAnalysis[17] built on the latter library to improve structural specifications for solute and solvent structures.

Components and organization

There are several input fields available for analyzing protein structures in this software. The "Structure Name" input field allows the user to specify the name of the protein structure being analyzed (Fig 1A). The "Structure Topology File Input" field is used to specify the topology file for the protein structure, which contains information about the connectivity of the protein atoms (Fig 1B). The "Trajectory File Input" field allows the user to specify the trajectory file for the molecular dynamics simulation being analyzed, which contains information about the position and velocity of each atom at each time step in the simulation (Fig 1C). The "Frame Number Input" field lets the user choose the specific frame they want to analyze from the trajectory file (Fig 1D). The "Solute Selection Input" field lets the user choose the solute they want to analyze and includes sub-inputs for the solute name and the atoms or residues to be included in the analysis (Fig 1Ea and 1Eb). The "Solvent Group Selection" input field lets the user specify the solvent they want to analyze and includes sub-inputs for the solvent name, the atoms or residues to be included, and the "Solvent Shell Count," which specifies the chemical composition of particular solvent shells in complex solvent mixtures (Fig 1Fa, 1Fb, and 1Fc). This allows users to analyze the different chemical structures and associations that comprise solvated atoms.

Designing and creating a software interface between MolecularNodes and SolvationAnalysis has been effective in creating a user-friendly and seamless solution for visualizing solvation shells. Unlike other projects such as Pyrite, it does not require a “molecular mesh” (a complex Blender work-around for parsing MD trajectories)[18] and instead leverages MDAnalysis to integrate topology and trajectory information into a “Universe” class for object creation and mesh construction without the need for user input.[19] This approach allows for easy visualization of multiple solvation shells belonging to a given frame, enhancing user-friendliness

---

[15] B. Foundation, “Home of the blender project - free and open 3D creation software,” *blender.org*. [Online]. Available: https://www.blender.org/. [Accessed: 21-Feb-2023].
[16] “MDAnalysis: A python package for the rapid analysis of molecular ...” [Online]. Available:https://www.researchgate.net/publication/328774554_MDAnalysis_A_Python_Package_for_the_Rapid_Analysis_of_Molecular_Dynamics_Simulations. [Accessed: 13-Apr-2023].
[17] O. Cohen, “MDAnalysis/Solvation-Analysis: A comprehensive suite of tools for analyzing liquid solvation structure.,” GitHub. [Online]. Available: https://github.com/MDAnalysis/solvation-analysis. [Accessed: 16-Feb-2023].
[18] Durrant JD. BlendMol: advanced macromolecular visualization in Blender. Bioinformatics. 2019 Jul 1;35(13):2323-2325. doi: 10.1093/bioinformatics/bty968. PMID: 30481283; PMCID: PMC6596883.
[19] “MDAnalysis: A python package for the rapid analysis of molecular ...” [Online]. Available:https://www.researchgate.net/publication/328774554_MDAnalysis_A_Python_Package_for_the_Rapid_Analysis_of_Molecular_Dynamics_Simulations. [Accessed: 13-Apr-2023].

and flexibility. In contrast, Pyrite does not support solvation shell visualization and is primarily focused on macromolecule visualization.

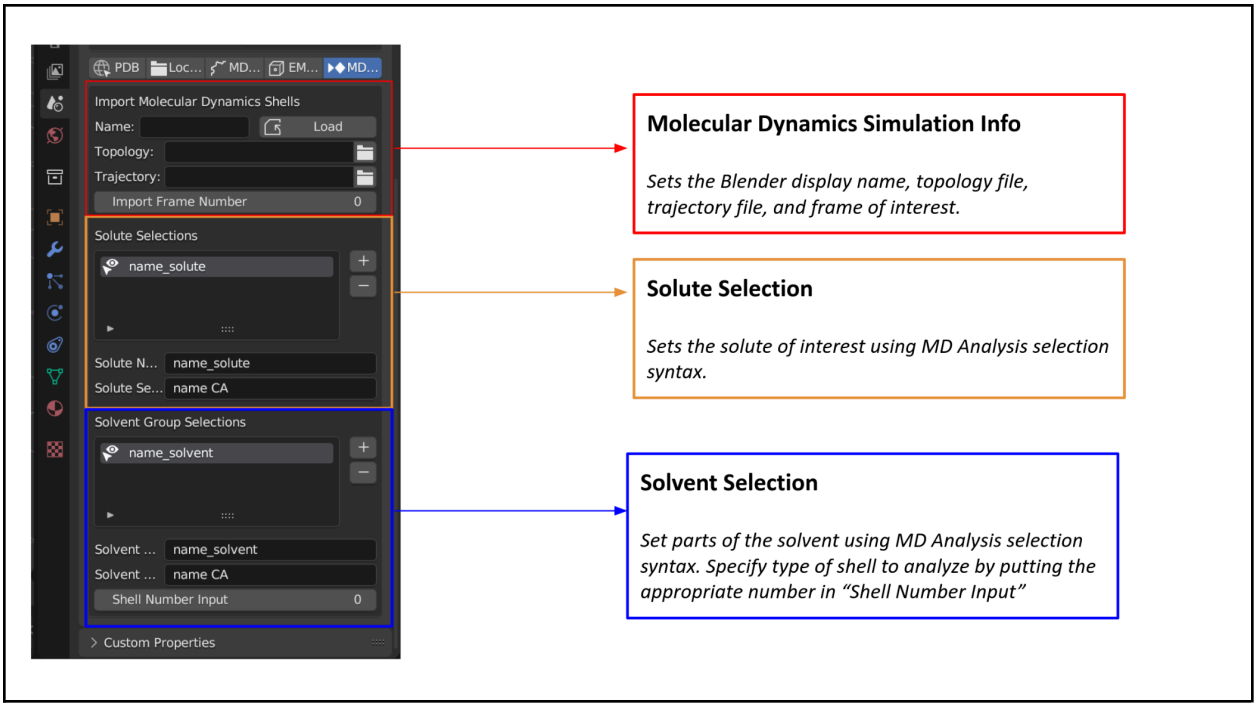


Figure 1: *The user-interface created for the solvation shell features 3 categorized inputs that pertain to MD Simulation Data, Solute Selection and Solvent Selection*

Testing

Software Testing

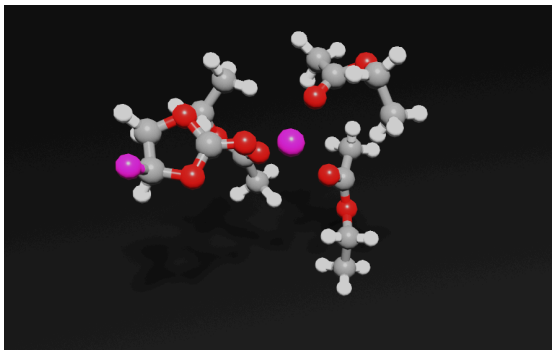
To ensure that the MolecularNodes visualization software works properly and outputs the correct solvation shells (Fig2A and Fig2B), we tested the UI in its entirety by generating visualization of solvent shells and comparing them to outputs from NGLView[20] an existing visualization software. This type of testing involves using the entire system as a finished product, including all components and modules, to ensure that they work together as intended, and is termed "end-to-end" testing.

The testing was carried out on various test cases involving different solutes and solvent systems for a given structure and frame. The trajectory and topology files were loaded into the software, and the user specified the solute and solvent groups, as well as the number of solvent shells to include in the analysis. The solvation shell visualization was then generated using the software.

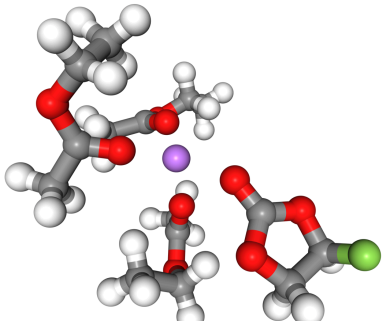
To ensure that the output of our implementation is consistent with existing visualization methods, the NGLView Jupyter widget (a status quo method for generating solvation structure

[20] "nglview," Digital Object Identifier System. [Online]. Available: https://doi.org/10.1093/bioinformatics/btx789. [Accessed: 06-Feb-2023].

that requires command line usage and outputs only coarse graphics was used in parallel on the same MD trajectory files. We compared the output of our Blender-based implementation with the output of NGLView to establish that the correct solvation structures were being depicted in each circumstance.



(A) Blender Image



(B) NGLView Image

Figure 2: *Ea_Fec in Frame 0 Shell 1*

Analysis of Testing

The results of the end-to-end testing showed that the MolecularNodes software successfully generated solvation shell visualizations that matched the expected output for all test cases. No discrepancies or errors were observed in the generated solvation shell visualizations. (Fig 2A) Overall, the software testing was effective in verifying the accuracy and reliability of the MolecularNodes visualization software for solvation shell analysis. (Fig 2B) It demonstrated that the software is capable of generating accurate solvation shell visualizations for various solutes and solvent systems. This finding also suggests that the software may be a valuable tool for researchers working in the field of molecular dynamics simulations, who require an easy-to-use software for solvation shell analysis.[21]

In our Approach or Methodology section, we described the development of a user interface in Blender that can display protein solvation in molecular dynamics simulations. To demonstrate the implementation of this approach, we conducted a series of computational experiments to test the functionality of the user interface and the SolvationAnalysis library.

First, we tested the user interface's ability to allow users to input solute and solvent group selections and specify the shell count. We input various solute and solvent structures into the interface, and the user interface was able to parse these structures and generate the proper selections for SolvationAnalysis.

Next, we tested the SolvationAnalysis library's ability to analyze solvation structures and extract specific solvation shells. We ran molecular dynamics simulations of a protein in a water

[21] O. Cohen, "MDAnalysis/Solvation-Analysis: A comprehensive suite of tools for analyzing liquid solvation structure.," GitHub. [Online]. Available: https://github.com/MDAnalysis/solvation-analysis. [Accessed: 16-Feb-2023].

environment and input the resulting trajectory into the user interface. The SolvationAnalysis library was able to generate the solvation shells for each frame, and we were able to select individual shells from the drop-down menu in the named attribute panel.[22]

We also tested the user interface's ability to display the solvation structures in 3D. The user interface was able to generate a 3D visualization of the solvation shells and display them in Blender (Fig 3).

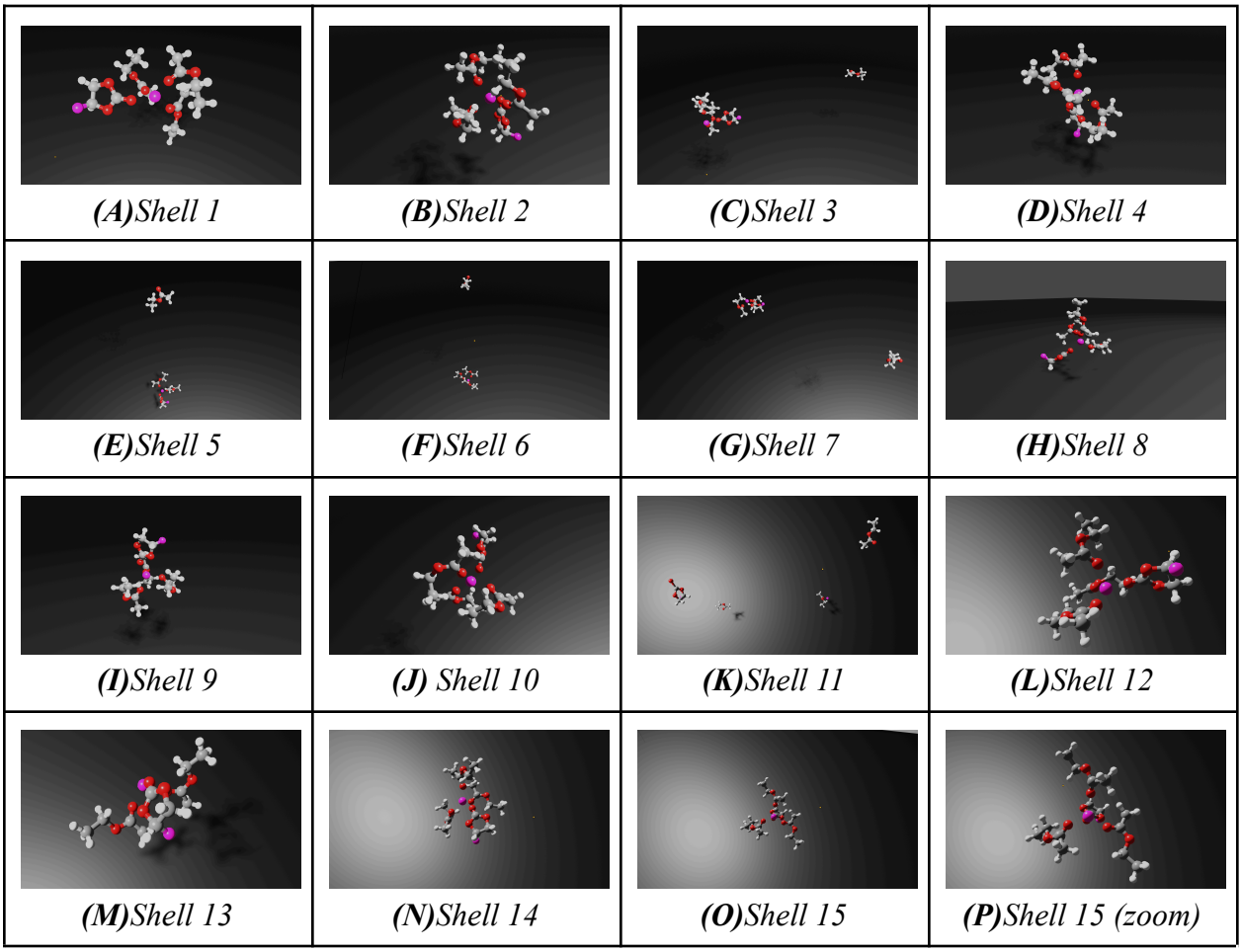


Figure 3: *These panels display the different shells from frame number 0.*

Future of The Software Project and Concluding Remarks

Public access to project

Moving forward, this software project will be made available to the public through the open-source MolecularNodes repository on GitHub[23]. Our goal is to merge this feature into the main MolecularNodes repository. I have been in contact with the lead software developer of

[22] O. Cohen, "MDAnalysis/Solvation-Analysis: A comprehensive suite of tools for analyzing liquid solvation structure.," GitHub. [Online]. Available: https://github.com/MDAnalysis/solvation-analysis. [Accessed: 16-Feb-2023].
[23] B. Johnston, D. Marson, and Y. Yao, "Bradyajohnston/molecularnodes: V2.3.0 for Blender 3.4.1+," Zenodo, 26-Jan-2023. [Online]. Available: https://doi.org/10.5281/zenodo.7572290. [Accessed: 06-Feb-2023].

MolecularNodes, and I am pleased to announce that this implementation is already available as a fork of MolecularNodes. The community is encouraged to contribute to the project by submitting issues and pull requests to enhance the interface's functionality and usability.[24] Further development plans include expanding the SolvationAnalysis library to support additional solvation types, such as ion solvation, and integrating additional analysis tools to the interface.

The development of a user-friendly interface for the analysis of protein solvation in molecular dynamics simulations has been a success. The interface's features, including the ability to input solute and solvent selections, specify shell count, and select individual solvation shells, have been thoroughly tested and found to work effectively. This product is ready for usage by the general, scientific public.

Future plans for maintenance and further development

Maintenance of the software project will be crucial for ensuring its continued relevance and utility in the scientific community. Regular updates will be made to address any bugs or compatibility issues with updated dependencies. Additionally, new features will be added to keep the interface up-to-date with the latest advancements in molecular dynamics simulations.

This project has been a valuable contribution to the field of computational biophysics, providing a user-friendly interface that makes molecular dynamics simulations more accessible to researchers. It is our hope that this project will serve as a useful tool for scientists and inspire further developments in this area of research.

References

1. Kimjoyc, "KIMJOYC/molecularnodes: Addon and nodes for working with structural biology and molecular data in Blender.," *GitHub*. [Online]. Available: https://github.com/kimjoyc/MolecularNodes. [Accessed: 12-Apr-2023].
2. B. Foundation, "Home of the blender project - free and open 3D creation software," *blender.org*. [Online]. Available: https://www.blender.org/. [Accessed: 21-Feb-2023].
3. B. Johnston, D. Marson, and Y. Yao, "Bradyajohnston/molecularnodes: V2.3.0 for Blender 3.4.1+," Zenodo, 26-Jan-2023. [Online]. Available: https://doi.org/10.5281/zenodo.7572290. [Accessed: 06-Feb-2023].
4. "nglview," Digital Object Identifier System. [Online]. Available: https://doi.org/10.1093/bioinformatics/btx789. [Accessed: 06-Feb-2023].
5. R. J. Gowers, M. Linke, J. Barnoud, T. J. E. Reddy, M. N. Melo, S. L. Seyler, D. L. Dotson, J. Domanski, S. Buchoux, I. M. Kenney, and O. Beckstein. MDAnalysis: A Python package for the rapid analysis of molecular dynamics simulations. In S. Benthall and S. Rostrup, editors, Proceedings of the 15th Python in Science Conference, pages 102-109, Austin, TX, 2016. SciPy. doi:10.25080/Majora-629e541a-00e

[24] Kimjoyc, "KIMJOYC/molecularnodes: Addon and nodes for working with structural biology and molecular data in Blender.," *GitHub*. [Online]. Available: https://github.com/kimjoyc/MolecularNodes. [Accessed: 12-Apr-2023].

6. Durrant JD. BlendMol: advanced macromolecular visualization in Blender. Bioinformatics. 2019 Jul 1;35(13):2323-2325. doi: 10.1093/bioinformatics/bty968. PMID: 30481283; PMCID: PMC6596883.
7. O. Cohen, "MDAnalysis/Solvation-Analysis: A comprehensive suite of tools for analyzing liquid solvation structure.," GitHub. [Online]. Available: https://github.com/MDAnalysis/solvation-analysis. [Accessed: 16-Feb-2023].
8. "Global structure of the intrinsically disordered protein tau emerges ..." [Online]. Available: https://pubs.acs.org/doi/pdf/10.1021/jacsau.1c00536. [Accessed: 21-Feb-2023].
9. "MDAnalysis: A python package for the rapid analysis of molecular ..." [Online]. Available:https://www.researchgate.net/publication/328774554_MDAnalysis_A_Python_Package_for_the_Rapid_Analysis_of_Molecular_Dynamics_Simulations. [Accessed: 13-Apr-2023].
10. S. Han, "Structure and dynamics in the lithium solvation shell of nonaqueous electrolytes," *Scientific Reports*, vol. 9, no. 1, Apr. 2019, doi: https://doi.org/10.1038/s41598-019-42050-y.
11. A. Smith *et al.*, "Topological Analysis of Molecular Dynamics Simulations using the Euler Characteristic," *Journal of Chemical Theory and Computation*, vol. 19, no. 5, pp. 1553–1567, Feb. 2023, doi: https://doi.org/10.1021/acs.jctc.2c00766.
12. I. V. Likhachev, N. K. Balabaev, and O. V. Galzitskaya, "Available Instruments for Analyzing Molecular Dynamics Trajectories," *The Open Biochemistry Journal*, vol. 10, no. 1, pp. 1–11, Mar. 2016, doi: https://doi.org/10.2174/1874091x01610010001.
13. A. Rácz, L. M. Mihalovits, D. Bajusz, K. Héberger, and R. A. Miranda-Quintana, "Molecular Dynamics Simulations and Diversity Selection by Extended Continuous Similarity Indices," *Journal of Chemical Information and Modeling*, vol. 62, no. 14, pp. 3415–3425, Jul. 2022, doi: https://doi.org/10.1021/acs.jcim.2c00433.

Appendix

Self-Reflection on Capstone Project Management

Apply fundamentals of software project management, including use of software productivity tools. Build a Software portfolio.

How did they keep track of their Capstone projects throughout the semester

Throughout the course of my Capstone project, I employed various software project management techniques and tools to help keep track of progress and ensure successful completion of the project. Initially, I created a detailed project plan that included a breakdown of tasks, timelines, and milestones. This plan was regularly updated and revised as the project progressed, with new tasks and deadlines being added as necessary.

To help keep track of progress and ensure that tasks were completed on time, a range of software productivity tools were used, including GitHub and Google Sheets. Google Sheets allowed us to track progress in real-time and collaborate on project documentation, while keeping all members of the collaboration updated on the progress of the project and any new additions to the shared GitHub repository.

How is the software and corresponding documentation being distributed? Why is this the most suitable approach? How was this decision made?

The final deliverable is available for public use on GitHub, a web-based platform for version control and collaboration. Rigorous documentation, as well as a first-time user tutorial, has been made available on GitHub. This approach was chosen as it allowed us to easily share code and documentation with our team members and project supervisors, while also providing version control to track changes and ensure that everyone was working with the latest version of the software. Furthermore, MolecularNodes is an open-source software available on GitHub. Storing our final deliverable on GitHub as a fork of MolecularNodes enables users of MolecularNodes to quickly find and download our contribution.

I believe that the project management techniques and tools employed throughout the course of the Capstone project were highly effective in helping to successfully complete the project. By keeping track of progress, assigning tasks, and collaborating effectively, we were able to deliver a high-quality software product on time and to the satisfaction of our project supervisor. Additionally, the use of GitHub for software distribution and version control provided a highly efficient and organized approach to managing the project documentation and code.

Apply best software engineering practices

Throughout the development of my Capstone project, I prioritized the use of best software engineering practices to guarantee that my code was scalable, maintainable, and efficient. To accomplish this, I followed various practices, including version control, code reviews, testing, documentation, and software repository organization. By using Git for version control, I tracked changes to my codebase, collaborated with my team members, and easily reverted to previous versions if necessary.

I conducted code reviews to ensure that my code was well-structured, adhered to best practices, and met our project requirements. Moreover, I utilized end-to-end testing to check that my code was functioning as expected and to ensure that it interacted with other components of the project correctly. I also created clear and concise documentation to help others understand my code and to make it easier for future developers to maintain and build upon my work.

Additionally, I organized my software repository into directories for code, documentation, and data, which made it simpler for team members to navigate and understand the project structure. These practices are demonstrated in my software deliverables through the organization of my code repository, and the documentation I created. My documentation explains the architecture and design of my codebase and includes installation and usage instructions, while my test image comparisons ensure that my code is functioning correctly and can be modified and extended in the future.

Work effectively within cross-functional teams

External Capstone Advisor: Dr. Jessica Nash

The feedback received from Dr. Nash and other peers was instrumental in shaping the final Capstone project product. Dr. Nash's contribution in outlining two implementation options: 1) the user inputs the solvent and shell information when the trajectory is loaded and 2) the user loads an MD trajectory with solvent information and can dynamically select the shell and frame. We evaluated both options and chose the first option, where the user inputs the solvent and shell information when the trajectory is loaded, as it aligned best with the project goals and constituted the minimum viable product (MVP) for the project. We also took into account the feedback received from other collaborators and classmates, incorporating her suggestions and ideas into the final product. Overall, effective collaboration and communication within the cross-functional team allowed us to create a successful capstone project product.

Collaborator: Orion Cohen

Orion's contribution to the final Capstone project product was the inclusion of solvation shell specifications in MolecularNodes. Orion's contribution to develop a solution to establish an interoperability of the SolvationAnalysis package, which was key to providing a set of methods for analyzing the solvation structure of a liquid. This in turn improved the functionality of the tool and provided a more comprehensive solution for users. The package integrates with MDAnalysis, which allows for the parsing of solvation information using the AtomGroup and Universe data structures. This addition allows users who are interested in understanding the solvation structure to use MolecularNodes as a tool for visualizing the results.

Classmate: Kevin Fong

Kevin's feedback was taken into consideration in order to ensure that the final product was accessible to a wide range of users, including those who may not be familiar with the basics of using MolecularNodes. To address this concern, we decided to target all documentation and instructional material to a general scientific audience, paying particular attention to the complexity of the explanations provided. This was done in order to ensure that users could easily understand how to use the software, and to make the product more accessible to a wider audience. In conclusion, the feedback received from Kevin was used to refine and improve the final product, ensuring that it was both effective and accessible to users of all skill levels.

Classmate: Xavier Linn

Xavier's feedback was used to make important decisions about the software engineering strategy, including the development of an MVP that would cover a majority of use cases that coincided with Dr. Nash's first implementation to make the software more accessible to contemporary scientists. We also recognized the need to target documentation and instructional material to a general scientific audience, paying attention to the complexity of explanations. All in all, the feedback received from Xavier helped guide the development of the final capstone project product.

Transfer technical concepts and techniques learned during the MMSSE program

I have learned various technical concepts and techniques that I have applied to my capstone project. For instance, I have acquired knowledge on software design principles, patterns, and methodologies in software engineering, object-oriented design, and design patterns. This knowledge was used to ensure that my software projects are modular, extensible, and maintainable. Additionally, I gained knowledge and skills in software testing, including test planning, and test design. This knowledge was utilized to develop and execute test cases and manually employ testing to ensure the quality of my capstone project.

Communication effectively when negotiating deliverables and submitting deliverables, and presenting in a professional setting

Communication is crucial in negotiating and submitting deliverables, as well as presenting in a professional setting. Effective communication skills enabled me to clearly convey my ideas, express concerns, and collaborate with others to achieve my goals. In the context of a capstone project, I communicated clearly and professionally with my collaborators and external capstone advisor as well as other stakeholders to ensure that the project is delivered on time and meets the desired quality standards.

During the capstone project, I had several opportunities to practice my communication skills, such as negotiating the project scope and timeline, submitting project deliverables to my supervisor, and presenting my work to an audience. I was able to apply the knowledge and skills to better my communication skills related to professional development, which provided me with strategies for effective communication, such as active listening, feedback, and conflict resolution.

By effectively communicating with my team members and stakeholders, I was able to manage expectations, resolve issues, and deliver a successful capstone project. This experience will serve me well in my future careers as software engineers, where effective communication skills are essential for success.

Mitigate project risks and points of failures

Project risks and points of failure can come from various factors, including insufficient resources, technical difficulties, changes in project scope, and unforeseen events. To manage these risks and points of failure, project teams need to identify them early and develop contingency plans. Additionally, continuous monitoring and effective communication among team members, stakeholders, and external partners are crucial to staying on track and meeting project objectives. Proper allocation of resources and prioritization of tasks are also important. Project management "creeps" such as scope, schedule, and budget creep can also affect project success, so it is essential to establish clear project goals and scope, develop a change management process, and monitor project progress regularly. Addressing issues and concerns promptly helped minimize my impact on the project timeline.

Write periodic project updates, including deliverable status and self assessment.

The project's progress will be regularly tracked and assessed according to a set schedule using Google Sheets. The project management sheets will be updated accordingly to ensure the project's deadlines are met. To facilitate this process, weekly meetings will be held every Tuesday with Dr. Drummond, the Capstone Project Course faculty, to discuss milestones and

address any concerns. Additionally, a weekly meeting with Dr. Nash, the External Capstone Supervisor, will be scheduled every Thursday. Self-assessments will also be conducted during these weekly meetings with both supervisors to evaluate the project's progress and identify areas for improvement. Maintaining open communication and regular assessment will ensure that the project remains on track and achieves its objectives.

Currently, we are on track with our planned deliverables. We have created the final deliverable to provide users with the ability to visualize solvation shells. We feel that we have made good progress on the project, and moving forward, we plan to prioritize a live demo and finish up the tutorial and documentation.