The Journal of Open Source Education

A course on the setup, running, and analysis of biomolecular simulations

Matteo T. Degiacomi [1,2], **Richard J. Gowers** [3], **Micaela Matta** [4], and **Antonia S. J. S. Mey** [2]

1 School of Informatics, University of Edinburgh, Informatics Forum, 10 Crichton St, Edinburgh EH8 9AB, United Kingdom **2** EaStCHEM School of Chemistry, University of Edinburgh, Joseph Black Building, David Brewster Road, Edinburgh, EH9 3FJ, United Kingdom **3** CHARM Therapeutics Ltd, 7 Pancras Square, London, N1C 4AG **4** Department of Chemistry, King's College London, London, SE1 1DB, United Kingdom

DOI: 10.21105/jose.00265

Software

- Review
- Repository
- Archive

Submitted: 15 September 2024
Published: 29 January 2025

Summary

We present an open-source course on how to set-up and analyse molecular dynamics (MD) simulations of biomolecules using proteins as a use-case. The course consists of a blend of lectures and practical sessions using Jupyter notebooks.

Statement of Need

Biomolecular systems were among the first to be studied using molecular dynamics (MD) simulations (Levitt & Warshel, 1975). As a result, biomolecular simulations are built on half a century of rich methodological development, embodied in a wide range of specialized software. The improvement in physical models dictating interatomic interactions coupled with an ever-increasing availability of computational power have enabled MD simulations to establish themselves as a technique complementary to experimental data (Hollingsworth & Dror, 2018) (Ciccotti et al., 2022). Starting from the simulation of small proteins for only a few nanoseconds (Levitt & Warshel, 1975), nowadays large biomolecular complexes featuring millions of atoms can be simulated for timescales orders of magnitude longer (Lindorff-Larsen et al., 2011). The data produced by MD simulations is noisy and high-dimensional though, and its usefulness is directly dependent on how faithfully the molecular system simulated recapitulates the physiochemical conditions of its real-world counterpart. Since the mid-1970s, significant progress has been made in automating the preparation of biologically relevant atomistic models and the analysis of simulation data. Nonetheless, modern computational scientists must still make critical decisions on how to assemble and simulate the system, as well as on which quantities to extract from the resulting data to accurately explain or predict experimental outcomes.

The material presented in this course has been developed as training material for the CCPBioSim consortium. Since 2022, it has been delivered to three cohorts of 25–35 international postgraduates attending the UK-based CCP5 Summer School on Molecular simulation. A first key aspect of this course is that, under the same hood, it provides information on both the set-up and the analysis of MD simulations, typically presented separately. A second key aspect is that it demonstrates how machine learning techniques can be integrated in the analysis of MD simulations and used to extract relevant information from an MD simulation.

Overview, Content, and Structure

Target Audience

This is a graduate-level course, aimed at beginners in biomolecular simulation. It is expected that students are already familiar with key concepts of molecular dynamics simulation theory, and have a basic working knowledge of Python and its core scientific packages (numpy, scipy, matplotlib).

Content

The objective of this course is not to make students proficient in one or a few specific software tools e for MD simulation preparation, execution, or analysis. Instead, it is aimed at providing students with a general overview of the key decision-making required to carry out MD simulations of biomolecules and extracting quantitative data from them. In this context, the course is divided into two Units featuring practical sessions and lectures. Practical sessions demonstrate how key concepts in molecular modelling are put into practice by exposing student to authentic tasks leveraging on commonly used Python packages, such as MDAnalysis (Michaud-Agrawal et al., 2011) (Gowers et al., 2016) (Alibay et al., 2023) and scikit-learn (Pedregosa et al., 2011). Lectures are software-agnostic and provide additional material to the course. While each practical session can be run by students on their own computer, these are also available in Google colab. This solution, requiring no local installation, is especially suitable for those unfamiliar with setting up a Python environment, or having limited access to computational resources.

Unit 1: Simulation Preparation

The first Unit is dedicated to providing background on protein structure, and how to prepare a protein for biomolecular simulation. In this Unit, students learn about how to critically observe a protein structure, and make informed decisions required to set-up a simulation that faithfully recapitulates a biologically relevant system.

Session	Materials
L1: Introduction to Proteins	Lecture Slides
L2: Understanding Protein Systems	Lecture Slides
L3: Protein-Ligand Docking	Lecture Slides
P: Protein-Ligand Docking	Notebook
L4: Simulation Setup	Lecture Slides
P: Simulation Setup	Notebook

Unit 2: Simulation Analysis

The second Unit is dedicated to providing the students with means to extract relevant quantitative information from a molecular dynamics simulation trajectory. A key aspect of this Unit lies in the demonstration of how machine learning techniques (clustering, dimensionality reduction, classification) can be used to extract meaningful information from noisy and high-dimensional data associated with biomolecular MD simulations.

Session	Materials
L5: Simulation Basic Analyses	Lecture Slides
P: Simulation Basic Analyses	Notebook
L6: Dimensionality Reduction	Lecture Slides
P: Dimensionality Reduction, part 1	Notebook

Session	Materials
P: Dimensionality Reduction, part 2	Notebook
L7: Clustering	Lecture Slides
P: Clustering	Notebook
L8: Data Classification	Lecture Slides
P: Data Classification	Notebook

Assessment and feedback

Each Jupyter notebook contains information on a specific topic, as well as tasks the student is asked to carry out independently. The tasks range from interpreting data previously produced, to running presented code with different parameters, to solving a specific problem by implementing a short Python code. Solutions to all questions are provided in each notebook as drop-down cells, enabling students to self-assess their understanding.

In our teaching practice, we provide students with post-its of two different colours that can be displayed on their computer screen — yellow indicating that everything is clear, pink indicating that help is required. At the end of each practical session, students are asked to use these same post-its to provide instructors with feedback on something they liked (yellow post-it), and something that requires improvement (pink post-it). In the three years we have delivered this course, this approach has enabled us to gather comprehensive feedback, which has helped us fine-tune both the teaching material and our delivery style. A key observation is that students, when presented with a new notebook, especially appreciate the instructors taking a few minutes to describe the overall structure of the notebook and the tasks it contains before beginning the practical work.

Conclusion

Thanks to the increasing availability of computational power and software automating many of the processes associated with biomolecular simulation and analysis, the palette of questions addressable with MD is broadening. While this is positive, it remains crucial for computational scientists to have a clear understanding of what is being simulated and how. Indeed, to date many decisions associated with system building and analysis cannot be delegated to a machine without human verification. In this context, we see our course as a first stepping-stone, detailing the key decisions that need to be made, providing examples of how this can be done in practice, and directing learners to relevant software and specialized analysis techniques for further education.

Despite its long history, MD remains an evolving field. New techniques that push the boundaries of what is possible keep emerging, as exemplified by the current revolution associated with the integration of modern machine learning techniques in molecular modelling pipelines. While we expect the majority of the concepts presented in this course to remain valid for many years, we are striving to keep the course material up-to-date by highlighting current methodological trends. For example, in the latest iteration of this course, we have introduced a discussion on how to interpret and use models produced by AlphaFold. (Jumper et al., 2021).

Contributions to the course

MTD and ASJSM conceived the course with contributions from RJG and MM.

Acknowledgements

Parts of the material in this course have been adapted from other sources with the agreement of respective authors. In this context, we would like to acknowledge Matt Thompson, Jeff Wagner, and Josh Mitchell for the material featured in the OpenFF tutorials, César Ramírez-Sarmiento and collaborators for material featured in the Autodock tutorial. We also wish to acknowledge Charlie Laughton for inspiration on discussing RMSD in conjunction with equilibration, and Irfan Alibay and Rohan Gorantla for help with setting up Colab environments.

References

Alibay, I., Barnoud, J., Beckstein, O., Gowers, R. J., Loche, P. R., MacDermott-Opeskin, H., Matta, M., Naughton, F. B., Reddy, T., & Wang, L. (2023). Building a community-driven ecosystem for fast, reproducible, and reusable molecular simulation analysis using mdanalysis. *Biophysical Journal*, *122*(3), 420a. https://doi.org/10.1016/j.bpj.2022.11.2277

Ciccotti, G., Dellago, C., Ferrario, M., Hernández, E. R., & Tuckerman, M. E. (2022). Molecular simulations: Past, present, and future (a Topical Issue in EPJB). *Eur. Phys. J. B*, *95*(1), 3. https://doi.org/10.1140/epjb/s10051-021-00249-x

Gowers, Richard J., Linke, Max, Barnoud, Jonathan, Reddy, Tyler J. E., Melo, Manuel N., Seyler, Sean L., Domański, Jan, Dotson, David L., Buchoux, Sébastien, Kenney, Ian M., & Beckstein, Oliver. (2016). MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In Sebastian Benthall & Scott Rostrup (Eds.), *Proceedings of the 15th Python in Science Conference* (pp. 98–105). https://doi.org/10.25080/Majora-629e541a-00e

Hollingsworth, S. A., & Dror, R. O. (2018). Molecular Dynamics Simulation for All. *Neuron*, *99*(6), 1129–1143. https://doi.org/10.1016/j.neuron.2018.08.011

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., & others. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, *596*(7873), 583–589. https://doi.org/10.1038/s41586-021-03819-2

Levitt, M., & Warshel, A. (1975). Computer simulation of protein folding. *Nature*, *253*(5494), 694–698. https://doi.org/10.1038/253694a0

Lindorff-Larsen, K., Piana, S., Dror, R. O., & Shaw, D. E. (2011). How Fast-Folding Proteins Fold. *Science*, *334*(6055), 517–520. https://doi.org/10.1126/science.1208351

Michaud-Agrawal, N., Denning, E. J., Woolf, T. B., & Beckstein, O. (2011). MDAnalysis: A toolkit for the analysis of molecular dynamics simulations. *Journal of Computational Chemistry*, *32*(10), 2319–2327. https://doi.org/10.1002/jcc.21787

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830.