Gromologist: a Gromacs-Oriented Utility Library for Structure and Topology Manipulation

Miłosz Wieczór[1,2,*], Jacek Czub[2]

1 Molecular Modeling and Bioinformatics, IRB Barcelona
2 Department of Physical Chemistry, Gdańsk University of Technology

*** milosz.wieczor@irbbarcelona.org**

Abstract

Despite the increasing automation of workflows for the preparation of systems for molecular dynamics simulations, the custom editing of molecular topologies to accommodate non-standard modifications remains a daunting task even for experienced users. To alleviate this issue, we created Gromologist, a utility library that provides the simulation community with a toolbox of primitive operations, as well as useful repetitive procedures identified during years of research. The library has been developed in response to users' feedback, and will continue to grow to include more use cases, thorough automatic testing and support for a broader spectrum of rare features. The program is available at `gitlab.com/KomBioMol/gromologist` and via Python's *pip*.

Introduction

Molecular dynamics (MD) simulations are a cornerstone of modern-day computational chemistry and biology, and in the last 40 years a range of general-purpose codes have been developed to make simulations more reproducible and accessible to the community [1–7]. Although the preparation of typical biomacromolecular systems, e.g. proteins, lipid membranes or nucleic acids containing standard residues, has become extremely streamlined and automated [8–10], the corresponding topology files – containing a full description of the system's connectivity and energetics – can be unreadable and not amenable to simple manual manipulation. As a result, any non-automated departure from these standards, such as removing atoms or adding bonds, can become a painstakingly complex process, in particular for novice users not aware of nuances and caveats of these fixed file formats.

To deal with these routine but non-standard tasks, high-level utility libraries are needed that can accommodate the increasing complexity of inputs and dynamically respond to popular requests from the community. One notable example of such a library, Parmed, has already been incorporated into Amber and contains many OpenMM-specific functions [11], but its primary goal is to provide easy access to general properties and allow for smooth interconversion between standard file formats. Another one, pmx, serves to prepare hybrid topologies for "alchemical" simulations and the analysis of thus obtained results [12]. Many routine tasks, though, are not covered by any popular library, leaving plenty of room for new developments.

In an attempt to fill that gap, we present Gromologist, a Pythonic utility library oriented at custom manipulation of Gromacs topologies. For many years now, Gromacs has been not only the most broadly used but also one of the fastest-growing MD engines in the field of molecular simulations [13, 1] and has a well-established user base, which – combined with the ease of format interconversion allowed by other tools – makes Gromologist a broadly applicable tool for all types of non-standard operations including, but not limited to, adding and removing atoms and bonds, introducing amino acid mutations in both the structure and topology, checking and listing force field parameter compatibilities, looking for mismatches between structures and topologies, merging files, manipulating alchemical states, modifying force field types or combination rules, parameter optimization or automated editing of structures. In addition, we maintain a strict zero-dependency policy (except for a few very specific features), so that the library is conveniently lightweight and can be easily installed with any modern Python distributions without causing dependency conflicts.

Features

Topology manipulation

To handle Gromacs topologies, Gromologist implements a hierarchical representation of sections, subsections and entries. Sections represent high-level abstractions, such as individual molecule definitions, force field parameter sets or headers/footers. Subsections correspond to individual fragments of the topology file, delimited with the `[ subsection_name ]` syntax, and directly handle most operations. Entries correspond to individual lines and provide a low-level access to all key features of the contents of a topology.

Splitting and merging. By default, topologies are saved as a single standalone file, including all .itp files referenced therein, but splitting to individual .itp is also supported. If needed, files can be made lightweight by keeping (a) only molecules listed in the `[ system ]` subsection and/or (b) only parameters used by the system.

Bonds and atoms. Primitive operations on topologies include adding, removing or swapping atoms in molecules while preserving a correct numbering scheme across all sections. Adding or removing bonds is also supported, including new bonds between two molecules, in which case the two become merged into a single molecule entry, and all new bonded terms (pairs, angles, dihedrals) are automatically identified and added, while existing ones are preserved and renumbered correctly. This allows for easy creation of chemical adducts or cyclic molecules, as well as editing of protonation states or simple chemical modifications without the need of extending the basic residue library (.rtp/.hdb files). Another heavily requested feature was the automated introduction of ,,special bonds" – disulfides and transition metal coordination bonds – that are not always correctly guessed by default Gromacs tools. Finally, Gromologist also provides a low-level interface for the mutation library, and can be easily incorporated into custom modification workflows.

Mutations. In this vein, a mutations module allows to introduce standard amino acid mutations to the topology without the need to go through the full process of topology generation. This has proven particularly useful in the study of heavily glycosylated systems, such as the SARS-CoV-2 Spike [14], where ready-made topologies and systems were often shared publicly but introducing even minor changes (such as a point mutation) would entail going through the same complicated server-based pipeline, risking introducing additional errors at this stage. This idea is also extended to protonation states, so that individual titratable residues can be protonated or deprotonated easily and in a consistent manner. As long as standard atom name conventions are followed, the mutation module is force-field agnostic, and the user can point to a specific .rtp file containing the respective residue definitions.

Alchemistry. Another set of utilities is related to the management of alchemical states. Gromologist can add or remove selected alchemical states, or swap them – a useful feature e.g. for the equilibration of B-states that helps avoid the computational overhead associated with the use of the free energy code.

Force fields. Finally, some features are meant to facilitate force field development and the implementation of selected methods such as solute tempering (REST2) [15]. These include the possibility of automatically cloning types along with all their bonded interaction parameters and renaming all selected atoms to the new type, or by automatically adding NBFIX-type modifications just by specifying the deviation from the standard Lorentz-Berthelot rules [16, 17]. An additional module implements a generic strategy for multiple dihedral fitting to QM data, akin to that available in the FFTK module of VMD for NAMD topologies [18].

Structure manipulation

The structure module can deal with both PDB and GRO files, interconverting between them when necessary. Common operations include constructing new atoms from existing ones, interpolating between structures, atom/residue renumbering according to custom rules or generation of virtual sites, e.g. upgrading 3-point water molecules to 4-point ones.

In addition, many often-required features of the PDB files can be filled in automatically. Chain assignment, an information typically lost in PDB to GRO conversion can be inferred based on a simple distance criterion, similarly to the CONECT entries required by some analysis programs. Elements can be inferred from atom names, and beta-factors can be set to reflect external data sets, including an option to smooth the values out spatially.

To assist with these operations, Gromologist implements a robust selection language akin to that of VMD [19], allowing for logical operations, selections "within X of Y", and a number of predefined molecule classes such as "protein", "nucleic" or "solvent". Periodic boundary condition (PBC) treatment is also implemented for any generalized triclinic box.

Simultaneous processing of topology and structure

In Gromologist, topologies can be matched with a compatible structure, so that where relevant, topology operations can be simultaneously performed on the structure attribute (and vice versa), facilitating the work with simulation-ready systems, and fully leveraging the information available in both files. Examples of such procedures include atom addition/removal, or introduction of mutations. When a topology is available, chain assignment can be made using molecule definitions therein. Moreover, when Gromacs often only indicates an unspecified mismatch between e.g. the number of atoms in structure and topology, Gromologist provides a specific list of all unmatched atom names, facilitating the identification of missing atoms or residues in either one.

Checking and printing

The last area of applications of Gromologist covers extracting information about the topologies and structures, or making it better visible to the user. On the most basic level, one can list molecules in the system, atoms in the molecule, as well as selected bonded terms (bonds, 1-4 pairs, angles, dihedrals) using atom names or atom types. Charges and masses of both the whole system and individual molecules are easy-to-access attributes. For a protein or a nucleic acid structure, the sequence can be printed chain by chain, and per-residue missing atoms identified, helping identify cases in which residues have to be rebuilt. Gradually diagnostic functions are being incorporated to quickly identify structural issues prior to simulation, like one currently allowing for validation of all chiral centres in a protein.

When required, atom names can be explicitly stated in the comments of bonded interaction entries to facilitate debugging, and numerical values of the parameters can be explicitly included in the topology. In the same vein, fields set using the `define` syntax can be set to their explicit values. Parameters whose numerical definitions are missing can also be identified with a single function call.

Dependencies

To be applicable as broadly as possible, Gromologist uses no dependencies other than base Python libraries. Although a few optional features, such as smoothing of beta-factors or dihedral optimization, do require `numpy` and `scipy`, these make up extremely rare use cases and therefore do not form part of the official dependency list.

Outlook

Given the existing set of capabilities, Gromologist is becoming an easily extensible platform for the introduction of further convenience functions, editing tools and workflows. Further development of the platform could benefit the Gromacs community by rapidly implementing utilities tailored to new or popular protocols, and a closer collaboration with the Gromacs development team is expected to directly address these issues. Moreover, extensive testing and some code reorganization will be required to ensure that the library covers all features supported by Gromacs, including more cryptic ones. In the long run, however, we expect Gromologist to become a stable element of the Gromacs environment and a valuable contribution to the broad simulation community.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 894489.

References

1. Leopold Talirz, Luca M. Ghiringhelli, and Berend Smit. Trends in atomistic simulation software usage [Article v1.0]. *Living Journal of Computational Molecular Science*, 3(1):1483–1483, 2021.

2. James C. Phillips, David J. Hardy, Julio D.C. Maia, John E. Stone, João V. Ribeiro, Rafael C. Bernardi, Ronak Buch, Giacomo Fiorin, Jérôme Hénin, Wei Jiang, Ryan McGreevy, Marcelo C.R. Melo, Brian K. Radak, Robert D. Skeel, Abhishek Singharoy, Yi Wang, Benoît Roux, Aleksei Aksimentiev, Zaida Luthey-Schulten, Laxmikant V. Kalé, Klaus Schulten, Christophe Chipot, and Emad Tajkhorshid. Scalable molecular dynamics on CPU and GPU architectures with NAMD. *Journal of Chemical Physics*, 153(4):044130, 2020.

3. A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Comp. Phys. Comm.*, 271:108171, 2022.

4. Romelia Salomon-Ferrer, David A. Case, and Ross C. Walker. An overview of the Amber biomolecular simulation package. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 3(2):198–210, 2013.

5. Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindah. Gromacs: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1-2:19–25, 2015.

6. B. R. Brooks, C. L. Brooks, A. D. Mackerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V. Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, and M. Karplus. CHARMM: The biomolecular simulation program. *Journal of Computational Chemistry*, 30(10):1545–1614, 2009.

7. Peter Eastman, Jason Swails, John D. Chodera, Robert T. McGibbon, Yutong Zhao, Kyle A. Beauchamp, Lee Ping Wang, Andrew C. Simmonett, Matthew P. Harrigan, Chaya D. Stern, Rafal P. Wiewiora, Bernard R. Brooks, and Vijay S. Pande. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS Computational Biology*, 13(7):e1005659, 2017.

8. Dmitry Suplatov, Yana Sharapova, and Vytas Švedas. EasyAmber: A comprehensive toolbox to automate the molecular dynamics simulation of proteins. *Journal of Bioinformatics and Computational Biology*, 18(6), 2020.

9. Pau Andrio, Adam Hospital, Javier Conejero, Luis Jordá, Marc Del Pino, Laia Codo, Stian Soiland-Reyes, Carole Goble, Daniele Lezzi, Rosa M. Badia, Modesto Orozco, and Josep Ll Gelpi. BioExcel Building Blocks, a software library for interoperable biomolecular simulation workflows. *Scientific Data*, 6(1):1–8, 2019.

10. Sunhwan Jo, Xi Cheng, Jumin Lee, Seonghoon Kim, Sang Jun Park, Dhilon S. Patel, Andrew H. Beaven, Kyu Il Lee, Huan Rui, Soohyung Park, Hui Sun Lee, Benoît Roux, Alexander D. MacKerell, Jeffrey B. Klauda, Yifei Qi, and Wonpil Im. CHARMM-GUI 10 years for biomolecular modeling and simulation. *Journal of Computational Chemistry*, 38(15):1114–1124, 2017.

11. Michael R. Shirts, Christoph Klein, Jason M. Swails, Jian Yin, Michael K. Gilson, David L. Mobley, David A. Case, and Ellen D. Zhong. Lessons learned from comparing molecular dynamics engines on the SAMPL5 dataset. *Journal of Computer-Aided Molecular Design*, 31(1):147–161, 2017.

12. Vytautas Gapsys, Servaas Michielssens, Daniel Seeliger, and Bert L. De Groot. pmx: Automated protein structure and topology generation for alchemical perturbations. *Journal of Computational Chemistry*, 36(5):348–354, 2015.

13. João V. Ribeiro, Rafael C. Bernardi, Till Rudack, John E. Stone, James C. Phillips, Peter L. Freddolino, and Klaus Schulten. QwikMD - Integrative Molecular Dynamics Toolkit for Novices and Experts. *Scientific Reports*, 6(1):1–14, 2016.

14. CHARMM-GUI COVID-19 archive. `https://charmm-gui.org/?doc=archive&lib=covid19`. Accessed: 2022-03-22.

15. Lingle Wang, Richard A. Friesner, and B. J. Berne. Replica exchange with solute scaling: A more efficient version of replica exchange with solute tempering (REST2). *Journal of Physical Chemistry B*, 115(30):9431–9438, 2011.

16. Jejoong Yoo and Aleksei Aksimentiev. New tricks for old dogs: Improving the accuracy of biomolecular force fields by pair-specific corrections to non-bonded interactions. *Physical Chemistry Chemical Physics*, 20(13):8432–8449, 2018.

17. Yun Luo and Benoît Roux. Simulation of osmotic pressure in concentrated aqueous salt solutions. *Journal of Physical Chemistry Letters*, 1(1):183–189, 2010.

18. Christopher G. Mayne, Jan Saam, Klaus Schulten, Emad Tajkhorshid, and James C. Gumbart. Rapid parameterization of small molecules using the force field toolkit. *Journal of Computational Chemistry*, 34(32):2757–2770, 2013.

19. William Humphrey, Andrew Dalke, and Klaus Schulten. VMD: Visual molecular dynamics. *Journal of Molecular Graphics*, 14(1):33–38, 1996.