PDB2DAT

Automating LAMMPS data file generation from PDB molecular systems using Python, Rdkit, and Pysimm

Assaf, Eli I.; Liu, Xueyan; Lin, Peng; Erkens, Sandra

DOI
10.1016/j.simpa.2024.100656

Publication date
2024

Document Version
Final published version

Published in
Software Impacts

Software Impacts 20 (2024) 100656

Contents lists available at ScienceDirect

Software Impacts

journal homepage: www.journals.elsevier.com/software-impacts

Original software publication

PDB2DAT: Automating LAMMPS data file generation from PDB molecular systems using Python, Rdkit, and Pysimm

Eli I. Assaf [a,*], Xueyan Liu [a], Peng Lin [b], Sandra Erkens [b]

[a] Delft University of Technology, Delft, The Netherlands
[b] Ministry of Infrastructure and Water Management (Rijkswaterstaat), The Netherlands

ARTICLE INFO

Keywords:
Molecular dynamics
Chemistry
Atomistic simulation
LAMMPS

ABSTRACT

Pdb2dat, developed in Python, is an open-source, self-contained utility that facilitates the conversion of PDB files into LAMMPS data files, catering to the need of initializing atomistic simulation from initial atomic configurations. It extracts molecular details from PDB files, uses Rdkit and Xyz2mol for bonding analysis and 3D conformer generation, and uses Pysimm for assigning force field types and charges. Designed to be lightweight and fully Pythonic, pdb2dat is suitable for use in privilege-limited high-throughput environments. The output details system topologies for use in MD simulations, significantly simplifying the preparatory steps needed by researchers to explore materials phenomena through LAMMPS.

Code metadata

Current code version	1.0.0
Permanent link to code/repository used for this code version	https://github.com/SoftwareImpacts/SIMPAC-2024-87
Permanent link to reproducible capsule	https://codeocean.com/capsule/0241504/tree/v1
Legal code license	GNU General Public License (GPL)
Code versioning system used	None
Software code languages, tools and services used	Python 3.12
Compilation requirements, operating environments and dependencies	Python 3.7+, Rdkit, Pysimm, and Numpy
If available, link to developer documentation/manual	https://codeocean.com/capsule/0241504/tree/v1/code/readme.md
Support email for questions	e.i.assaf@tudelft.nl

1. Introduction

The utilization of atomistic simulations in the investigation of novel materials has seen a notable increase, facilitating the exploration of fundamental phenomena with reduced dependence on experimental techniques. Traditionally, the accurate application of these modeling techniques was confined to specialists within the realms of Computational Chemistry, Physics, and high-performance numerical computing. Nevertheless, the enhanced availability of computational resources, encompassing both software and hardware, has extended this capability to researchers beyond these specialized fields [1]. This development has simplified the exploration of material responses and dynamics [2], attributed partly to the proliferation of comprehensive open-source simulation packages, such as LAMMPS [3]. However, the process of preparing and initializing atomistic models for simulation, even with advanced tools like LAMMPS, demands a profound understanding of the available tools and resources. A significant challenge in model preparation involves the initial placement of atoms, molecules, and their topologies. Yet, a critical bottleneck is the assignment of force field atom types and charges within the atomic files, enabling MD software like LAMMPS to calculate the forces acting on atoms, thereby driving particle motion. This step necessitates extensive knowledge of the selected force field, its applicability, and the ability to classify particle types based on chemical descriptors.

Several software solutions, tools, and scripts exist to facilitate the conversion of common atomic model files into the topology files required by MD engines. However, these solutions are typically part of licensed software packages, such as Materials Studio [4] or Materials

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: e.i.assaf@tudelft.nl (E.I. Assaf), x.liu@tudelft.nl (X. Liu), p.lin-2@tudelft.nl (P. Lin), s.m.j.g.erkens@tudelft.nl (S. Erkens).

https://doi.org/10.1016/j.simpa.2024.100656
Received 9 April 2024; Received in revised form 23 April 2024; Accepted 3 May 2024

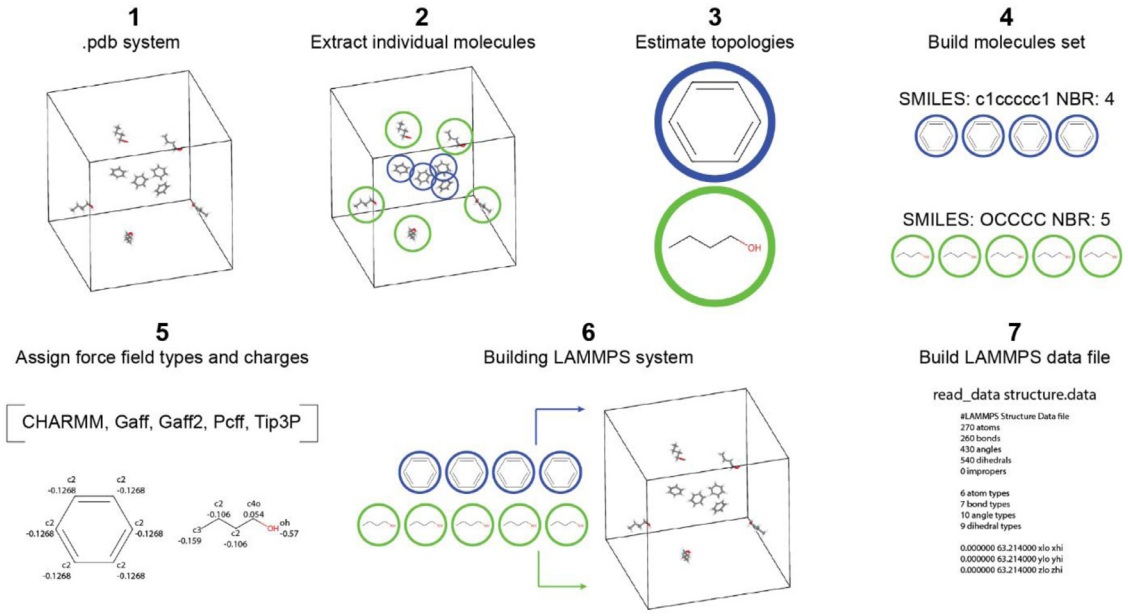

Fig. 1. Depiction of the steps performed by PDB2DAT to convert PDB files into LAMMPS-readable data files.

Design Medea, or are limited to open-source scripts that often lack effectiveness and simplicity and may not directly support commonly used file types or simulation engines like LAMMPS. For example, the 'msi2lmp' tool [5], bundled with LAMMPS, enables the transformation of Biovia's Materials Studio (.car/.mdf) files into topology files that are compatible with LAMMPS for specific force fields. Despite its efficacy, the reliance on Biovia's deprecated file format necessitates the use of licensed software such as Materials Studio for the preparation of atomistic simulations, further complicating the preparation of LAMMPS-ready atomistic systems. Moreover, the employment of more sophisticated open-source tools requires a comprehensive grasp of force field mechanics, advanced computational programming abilities, and detailed knowledge of implementation processes. This complexity limits their accessibility to those with expertise in the design of atomistic code algorithms.

In this manuscript, we introduce 'pdb2dat', a Python-based utility leveraging Rdkit [6] and Pysimm [7] to transform molecular systems from widely used PDB (Protein Data Bank) [8] file formats into structure data files immediately readable by LAMMPS. 'pdb2dat' processes the PDB file to isolate individual molecules, deduces stable and valid conformers (considering the lack of connectivity/bonding information in PDB files), constructs fully described molecular objects, and assigns force field types and charges to every atom in the system using a selection of force fields included in Pysimm (CHARMM [9], Gaff, Gaff2 [10], Pcff [11], and Tip3p). It then generates the topology *data* file essential for simulations within LAMMPS. The tool's ability to efficiently convert PDB files of organic mixtures into files ready for MD simulations with LAMMPS has been instrumental in numerous research projects, as evidenced by its application in the generation of hundreds of atomistic models across various studies and projects. Despite its simplicity, the robust and Pythonic nature of 'pdb2dat' renders it highly suitable for integration into high-throughput algorithms within HPC environments, which are often characterized by restricted availability of environments and dependencies.

2. Software description

PDB2DAT, developed in Python 3.12, processes PDB files, which encompass the atomistic details of a chemical mixture of molecules. It identifies and isolates the molecules contained within these files, capturing both atomic positions and the limited bonding information available in PDB files. Each molecule is then parsed into a fully described 3D conformer, incorporating complete bonding information, through the utilization of Rdkit and XYZ2MOL [12]. Subsequently, Pysimm is employed to assign force field types and charges to each molecule object. These molecules are then positioned within a Pysimm LAMMPS system, culminating in the generation of a data file that delineates the system's topologies, rendering it ready for use in LAMMPS simulations. The overall description of the steps performed by the script are depicted in Fig. 1 are described as follows:

1. PDB2DAT.py initiates its process by reading through a PDB file, isolating individual molecules through the extraction of HETATM/CONECT blocks that correspond to specific molecules within the PDB file. The dimensions of the original simulation box are recorded from the file's first line.
2. Subsequently, the script translates the molecular blocks derived from the PDB file into SMILES notations. Considering that SMILES notations necessitate comprehensive topological information [13], and PDB files lack the requisite bonding details to fully describe molecules, PDB2DAT employs Rdkit's adaptation of Jansen's XYZ2MOL script. This script attempts to infer the bonding information of atom groups based on their spatial positions.
3. With the derived SMILES notations, the program advances to construct Rdkit molecule objects. These objects, being highly malleable, facilitate the ensuing steps essential for the formulation of a system compatible with LAMMPS.
4. PDB2DAT continues by assembling a collection of molecules and calculating various molecular and systemic properties, such as molecular formulas. These properties, which can be reviewed in the program's log, serve to verify the accuracy of the molecular system reconstructed from the PDB file. Additionally, 2D representations of all identified molecule types may be generated if required.
5. In parallel, the array of molecule objects is introduced to the Pysimm_system class, where Rdkit molecule objects are transformed into formats interpretable by the Pysimm library.
6. The program then assigns force field types and charges to each molecule within the system. Recognizing the computational intensity of force field type assignment, and the likelihood of encountering multiple instances of identical molecules, PDB2DAT

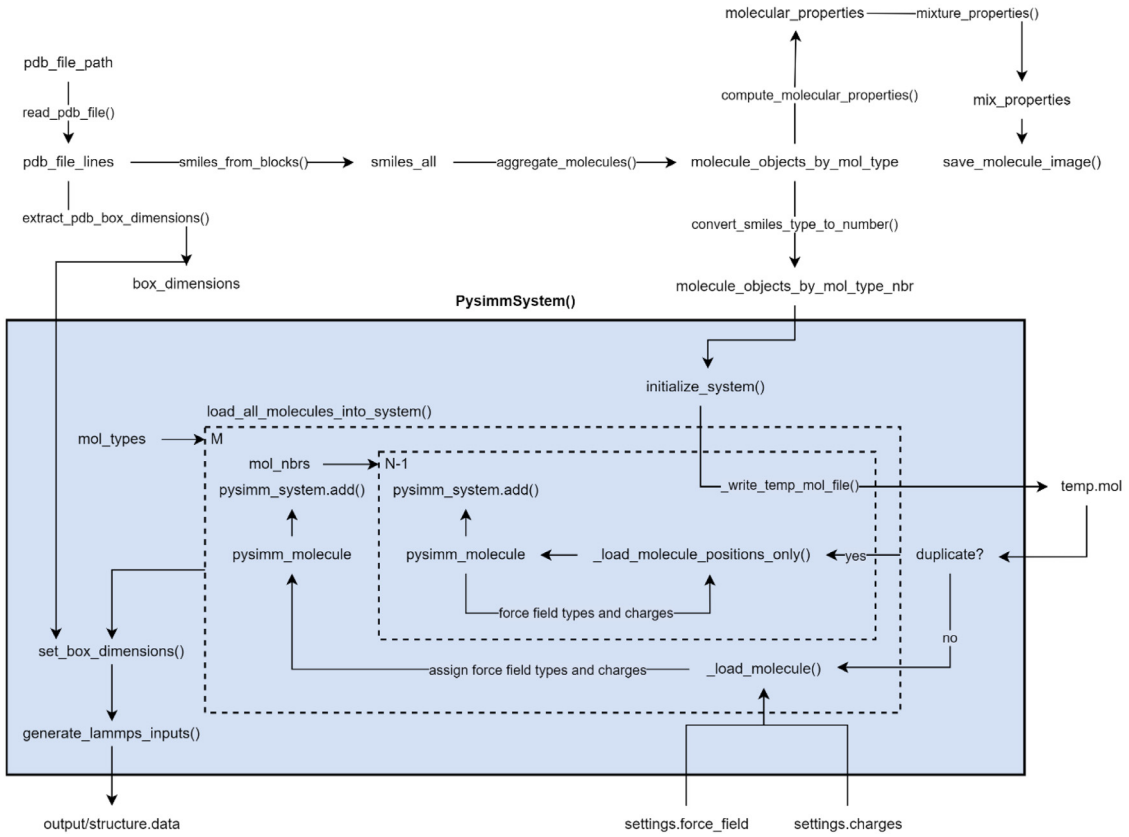

Fig. 2. Control flow diagram providing a visual representation of the functions, instances, and variables that govern the functional behavior of PDB2DAT during execution.

optimizes this step. It assigns force field types and charges to a single instance of each molecule type, replicating its object to construct additional instances of the same molecule type within the system. Only the atomic positions are modified for each molecule, whether they are original or duplicates.

7. The dimensions of the simulation box, as indicated in the PDB file, are applied to the Pysimm LAMMPS system.
8. Finally, a LAMMPS data file is produced based on the constructed system. The configuration of this file varies depending on the selected force field and the method employed for computing charges. The resulting data file can be visualized by loading it into a popular molecular visualization software, such as OVITO [14].

3. Software architecture

The operation of PDB2DAT necessitates three core dependencies: Rdkit, Pysimm, and Numpy. Additionally, the Pillow library serves as an optional dependency, facilitating the generation of images for the isolated molecules, should visual outputs be required. The software's architecture comprises five main components within its working directory. The primary executable, pdb2dat, runs the aforementioned processes. The core_functions.py file houses all callable functions essential for the tool's functionality. The pysimm_system.py file creates a Pysimm object, encapsulating all information necessary for the generation of a valid LAMMPS atomistic system. The settings.py file allows users to modify program parameters to adjust functionalities according to their requirements. Lastly, the output/ directory functions as a repository for the program's output files, ensuring organized storage of generated data. A comprehensive control diagram depicting the function calls during the program's execution can be found in Fig. 2.

4. Impact

PDB2DAT is designed to serve the scientific community by offering a streamlined approach to converting widely recognized atomistic file types into LAMMPS-ready data files. This tool holds particular significance for researchers outside the domain of computational chemistry who nonetheless seek to run MD simulations for the study, exploration, and development of novel materials. The program's simplicity and self-sufficiency facilitate the swift creation of comprehensive atomistic systems prepared for LAMMPS integration, establishing PDB2DAT as a vital resource for its intended purpose. It has been extensively utilized in the production of numerous atomistic systems, notably in investigations examining the effects of various additives, rejuvenators, solvents, and other organic compounds on bituminous materials and heavy oil-like substances [15,16].

Furthermore, PDB2DAT has been instrumental in generating a wide range of systems for the purpose of reference trajectory generation in the parameterization of both all-atom and Coarse-Grained force fields [17]. This process demands a substantial volume of systems and trajectories, underscoring the tool's value. PDB2DAT also facilitates the quick setup of molecular systems, enabling users to produce densely packed systems when used in conjunction with appropriate LAMMPS input scripts. These systems can subsequently be modified to incorporate more detailed or customized force field types and charges.

In essence, PDB2DAT's application in environments characterized by high throughput and extensive parallelism empowers scientists to efficiently assign force field types and charges to PDB files. This capability is particularly beneficial to those who may not have a deep engagement with the core principles of computational chemistry and atomistic modeling, thereby simplifying a task that has traditionally been challenging.

5. Limitations and future work

PDB2DAT's roadmap for development aims to enhance its functionality by integrating more features from Python libraries like Rdkit and Pysimm. Future updates include advanced molecular initialization, direct generation of LAMMPS input files for immediate simulation execution, and the adoption of more sophisticated force field and charge computation methods to improve accuracy. Another significant upgrade is replacing the current bond estimation method with a more accurate topology estimation technique to avoid incorrect atom bonding. Efforts will also extend PDB2DAT's applicability to a broader range of PDB systems, including elements beyond the current focus. While a User Interface (UI) is considered to simplify interactions, careful consideration will be given to ensure it does not complicate the program's use. These enhancements are geared towards leveraging the full potential of underlying libraries while preserving PDB2DAT's simplicity and effectiveness.

CRediT authorship contribution statement

Eli I. Assaf: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Xueyan Liu:** Validation, Supervision, Resources, Project administration, Funding acquisition, Conceptualization. **Peng Lin:** Visualization, Validation, Project administration, Conceptualization. **Sandra Erkens:** Validation, Resources, Project administration, Investigation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used OpenAI's ChatGPT4.0 to simplify verbose paragraph descriptions. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Acknowledgments

This paper/article is created under the research program Knowledge-based Pavement Engineering (KPE). KPE is a cooperation between Rijkswaterstaat, TNO, and TU Delft in which scientific and applied knowledge is gained about asphalt pavements and which contributes to the aim of Rijkswaterstaat to be completely climate neutral and to work according to the circular principle by 2030. The opinions expressed in these papers are solely from the authors.

References

[1] H. Gould, J. Tobochnik, W. Christian, An introduction to computer simulation methods, Comput. Phys. 10 (2007) 652–653.
[2] M.S. Shell, Thermodynamics and statistical mechanics : an integrated approach, 2015.
[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, LAMMPS-a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales, Comput. Phys. Comm. 271 (2022) 108171.
[4] M. Meunier, S. Robertson, Materials studio 20th anniversary, Mol. Simul. 47 (7) (2021) 537–539.
[5] J.A. Greathouse, Building LAMMPS Data Files with Car/Mdf Files and the Msi2lmp Utility, Sandia National Laboratories, Albuquerque, NM, USA, 2010.
[6] A.P. Bento, A. Hersey, E. Félix, G. Landrum, A. Gaulton, F. Atkinson, L.J. Bellis, M. De Veij, A.R. Leach, An open source chemical structure curation pipeline using rdkit, J. Cheminformat. 12 (2020) 1–16.
[7] A.G. Demidov, M.E. Fortunato, C.M. Colina, Update 0.2 to pysimm: a python package for simulation of molecular systems, SoftwareX 7 (2018) 70–73.
[8] S.K. Burley, H.M. Berman, G.J. Kleywegt, J.L. Markley, H. Nakamura, S. Velankar, Protein data bank (PDB): the single global macromolecular structure archive, Protein Crystallogr. Methods Protocols (2017) 627–641.
[9] A.D. MacKerell Jr., N. Banavali, N. Foloppe, Development and current status of the CHARMM force field for nucleic acids, Biopolymers: Original Res. Biomolecul. 56 (4) (2000) 257–265.
[10] J. Wang, R.M. Wolf, J.W. Caldwell, P.A. Kollman, D.A. Case, Development and testing of a general amber force field, J. Comput. Chem. 25 (9) (2004) 1157–1174.
[11] H. Sun, S.J. Mumby, J.R. Maple, A.T. Hagler, An ab initio CFF93 all-atom force field for polycarbonates, J. Am. Chem. Soc. 116 (7) (1994) 2978–2987.
[12] J.H. Jensen, Xyz2mol, GitHub repository 985, 2020.
[13] D. Weininger, SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules, J. Chem. Inf. Comput. Sci. 28 (1) (1988) 31–36.
[14] A. Stukowski, Visualization and analysis of atomistic simulation data with OVITO–the open visualization tool, Model. Simul. Mater. Sci. Eng. 18 (1) (2009) 015012.
[15] E.I. Assaf, X. Liu, P. Lin, S. Erkens, S. Nahar, L.I. Mensink, Studying the impact of phase behavior in the morphology of molecular dynamics models of bitumen, Mater. Des. 230 (2023) 111943.
[16] Y. Gao, X. Liu, S. Ren, E.I. Assaf, P. Liu, Y. Zhang, Nanostructure and damage characterisation of bitumen under a low cycle strain-controlled fatigue load based on molecular simulations and rheological measurements, Composites B (2024) 111326.
[17] E.I. Assaf, X. Liu, P. Lin, S. Erkens, Introducing a force-matched united atom force field to explore larger spatiotemporal domains in molecular dynamics simulations of bitumen, Mater. Des. (2024) 112831.