SMI2PDB
A self-contained Python tool to generate atomistic systems of organic molecules using their SMILES notations

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

DOI
10.1016/j.simpa.2024.100655

Publication date
2024

Document Version
Final published version

Published in
Software Impacts

Citation (APA)
Assaf, E. I., Liu, X., Lin, P., & Erkens, S. (2024). SMI2PDB: A self-contained Python tool to generate atomistic systems of organic molecules using their SMILES notations. *Software Impacts*, *20*, Article 100655. https://doi.org/10.1016/j.simpa.2024.100655

Contents lists available at ScienceDirect

Software Impacts

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

Original software publication

SMI2PDB: A self-contained Python tool to generate atomistic systems of organic molecules using their SMILES notations

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

The advent of computational techniques, particularly atomistic simulations, has lessened the dependency on physical experiments in various scientific fields. Yet, the preparation complexity for simulations using platforms like LAMMPS and GROMACS persists. We introduce SMI2PDB, a Python tool that automates molecular systems assembly from SMILES to PDB format, easing molecular dynamics simulation setups. SMI2PDB manages molecule configuration and quantification effortlessly, establishes stable conformers, applies random rotations, and positions them in a simulation box with a Sobol sequence to reduce overlaps. This script facilitates the rapid preparation of complex organic mixtures for use in simulations, enhancing the exploration of novel materials.

Code metadata

Current code version	1.0.0
Permanent link to code/repository used for this code version	https://github.com/SoftwareImpacts/SIMPAC-2024-81
Permanent link to reproducible capsule	https://codeocean.com/capsule/9730644/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+, Numpy, Numba, Rdkit, and Sobol-seq
If available, link to developer documentation/manual	https://codeocean.com/capsule/9730644/tree/v1/code/readme.md
Support email for questions	e.i.assaf@tudelft.nl

1. Introduction

Computational methods are now integral to scientific research, significantly reducing the need for physical experimentation. Atomistic modeling is a prime example of this trend, where molecular dynamics (MD) simulations, once limited to the fields of physics and chemistry, are now utilized across diverse fields [1]. This shift allows for the accurate exploration of novel materials, circumventing the need for extensive experimental work. Simulation tools like LAMMPS [2] and GROMACS [3] have set industry standards but are not without their complexities, particularly in system preparation for simulations, which requires exact molecular configurations as starting points [4].

Researchers often resort to combining tools like Packmol [5], Open-Babel [6], and OVITO [7] to ready simulations for platforms like LAMMPS, a less than optimal use of their time, more so for those examining organic mixtures needing numerous models for their studies. Hence, an accessible, standalone script like SMI2PDB is of great value, offering a straightforward method for assembling molecular systems, easing the technical and time constraints often encountered in this preparatory phase (see Fig. 1).

2. Software description

SMI2PDB is a Python-based script that takes in a molecular system described by the SMILES [8] notation of the molecules needed, their number, and some additional system-describing parameters, and outputs a molecular system in PDB format, a popular atomistic file format,

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.2024.100655
Received 13 April 2024; Received in revised form 23 April 2024; Accepted 3 May 2024

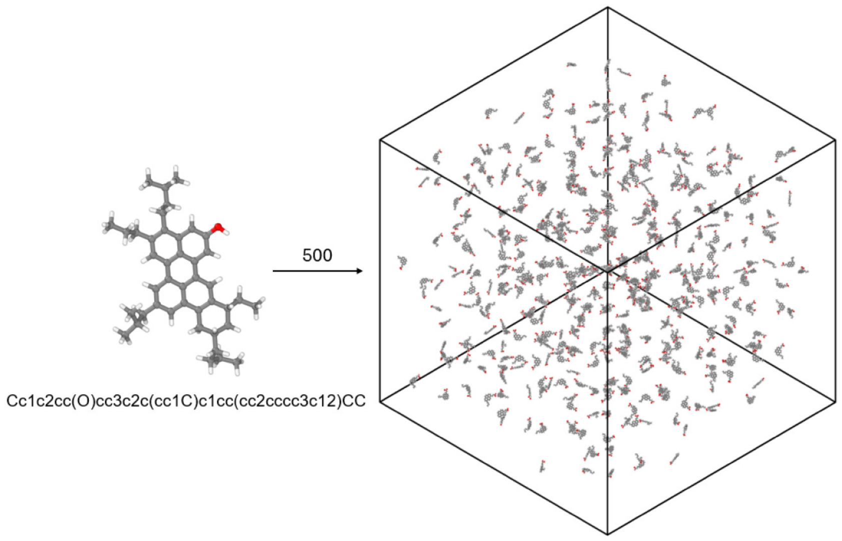

Fig. 1. Molecular system comprised of 500 phenolic asphaltene molecules generated using SMI2PDB visualized using OVITO. This system has a density of 200 kg/m^3 and is generated in about 10 s.

ready to be used by many simulation packages and other third-party software [9]. The user is only expected to modify an *inputs.py* file which then the *smi2pdb.py* script uses to run the expected task. The overall description of the steps performed by the script are described as follows:

1. The user configures the inputs.py file, setting the necessary parameters to define the molecular system. These parameters include molecular types, quantities, and various simulation properties outlined in a Table 1.
2. The smi2pdb.py script is executed, which begins by interpreting the molecules_dictionary. It initializes stable molecular conformers for each molecule type using RDKit [10], which involves hydrogenation and energy minimization.
3. Each molecule is duplicated until the required quantity is achieved for each type. If enabled, molecules are subjected to random rotations around their geometric center.
4. The software then initiates the mixture generation process by creating an empty cubic simulation box. The box's dimensions are determined by the density value from the input.py file.
5. Positions within the simulation box are determined using a Sobol sequence to ensure an even spatial distribution of molecules with minimal overlap.
6. A confinement margin is established by ensuring that the molecules are placed within the box's boundaries, offset by half the value given by inputs.layer_offset.
7. The molecular objects are translated to their assigned positions in the simulation box by aligning their geometric center with the Sobol sequence points.
8. The software calculates the interatomic potential energy using a Lennard-Jones potential utilizing energy and equilibrium distance parameters set to 1.0.
9. If the calculated potential energy is below the inputs.energy_threshold, the configuration is accepted. This mixture is then exported in PDB format.
10. In case the potential energy threshold is not met, the mixture is discarded. The process generates a new simulation box with a fresh set of Sobol positions and repeats the evaluation with newly generated molecular conformations and rotations.
11. This cycle continues until the desired number of mixtures, as defined by inputs.nbr_of_mixtures_needed, is produced.

A pictographic representation of these steps is presented in Fig. 2.

3. Software architecture

The software, implemented in Python 3.12 and reliant on Numpy, Numba [11], Rdkit [10], and Sobol-seq libraries, comprises seven primary components: "output/", "static_functions.py", "inputs.py", "log_functions.py", "smi2pdb.py", "mixture.py", and "molecule.py". Auxiliary modules "log_functions.py" and "static_functions.py" are called upon for their utility functions, including logging, file i/o, and generic computations. The script commences with "smi2pdb.py", initiating a "JobRunner()" instance that manages job execution and output storage in the directory "output/$job_id". It proceeds by interpreting the "inputs.py" file to instantiate a "Mixture()" from "mixture.py", encapsulating global attributes of the mixture such as molecule types, spatial coordinates, and other relevant properties. This "Mixture()" then generates "Molecule()" instances, each representing a singular molecule's objects that include their SMILES string, 3D conformations and relevant methods to alter them as needed (e.g., randomly rotate, translate, or minimize them). Upon populating the "Mixture.molecules" array, the system's molecular positions are determined, translated, and the overall mixture's potential energy evaluated. Conforming mixtures are saved as PDB files in "output/$job_id/Mixture_$i". This iterative process is repeated by "JobRunner()" until the requisite number of mixtures is reached. A through depiction of the programmatic elements involved in the execution of the script is shown in Fig. 3.

4. Impact

SMI2PDB is a Python-based script specifically designed to enhance the efficiency and scope of MD materials research. Developed over a two-year period, this script addresses critical needs in the generation of molecular systems by facilitating the rapid construction of complex hydrocarbon mixtures. The script utilizes SMILES notation to quickly generate atomistic models, making it invaluable for studies involving heavy oils, bitumens, and their interactions with various additives and rejuvenators.

One of the primary advantages of SMI2PDB is its ability to facilitate the creation of molecular systems without the extensive manual definition typically required. This includes automating the placement of molecules within a simulation box, ensuring stable spatial conformations, and minimizing interatomic overlaps. The output generated by SMI2PDB is compatible with most open-source MD visualization and engine programs, simplifying the integration into existing research workflows.

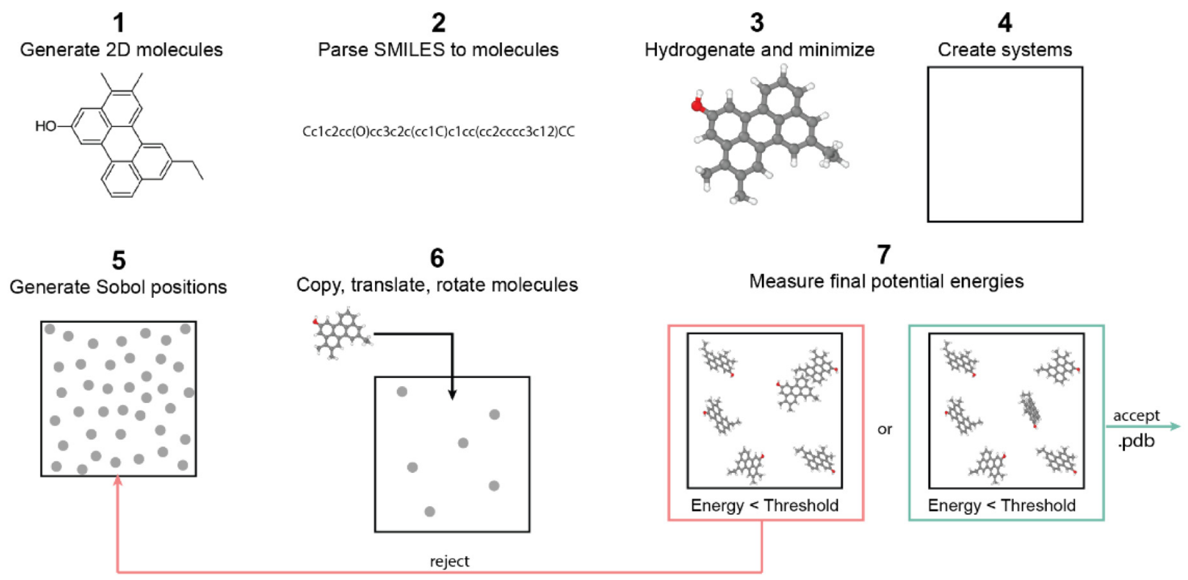

Fig. 2. Depiction of the steps performed by SMI2PDB to generate molecular mixtures using the SMILES notation of molecules.

Table 1
Description of the required parameters in inputs.py for SMI2PDB to run.

Parameter	Description	Typical range
mixtures_needed	Number of independent molecular systems required (unitless)	1 to 10
density	Density of the simulation box to place the molecules into (kg/m^3)	1.00 to 400
layer_offset	Distance (from the boundaries) to confine the molecules into (Angstroms)	0.0 to 50
energy_threshold	Lennard-Jones interatomic potential energy limit for a mixture to be accepted or rejected (LJ units)	−5.0 to 5.0
molecules_dictionary	Dictionary whose keys are the name of the molecules, and its items are subdictionaries with keys: "smiles", "nbr_of_molecules", and "rotate"	–
nbr_of_trials	Number of iterations beyond which the program stops trying to place molecules without being rejected	100–10000

The tool has significantly contributed to the field by enabling the development of over 8000 molecular models used to parameterize a new United Atom force field in LAMMPS. Additionally, it has facilitated the creation of over 2000 diverse molecular models to study the impact of different molecular structures on bituminous materials. These capabilities have been crucial for advancing the understanding of material properties and their modification through modeling techniques or by introducing a wide variety of chemical additives [12–14].

SMI2PDB's design emphasizes simplicity and self-containment, making it adaptable to various computational environments, including High Performance Computing setups. This aspect is particularly beneficial for researchers working in environments with restricted computational privileges. The script's lightweight, standalone nature allows for efficient parallel execution, enabling the simultaneous production of numerous molecular systems. This efficiency is critical for projects requiring the rapid generation of a large array of models, which would otherwise be impractical due to the time-consuming nature of manual model construction.

The success and utility of SMI2PDB are evidenced by its ability to integrate seamlessly into more extensive Python projects, providing a robust tool for researchers not primarily focused on MD methods but who require detailed molecular simulations to support their research in bituminous materials.

5. Limitations and future work

While tools like PackMol exist that provide extensive features for initializing molecular systems, SMI2PDB will deliberately not expand in this area. The primary goal is to preserve the simplicity, autonomy, and self-contained nature of SMI2PDB. This focus is designed to ensure that the program remains effective when integrated into broader scripts and automated frameworks, which is crucial for its use during the feasibility assessment phases of molecular modeling research.

Efforts will be directed towards enhancing SMI2PDB's ability to handle inputs and outputs that are fundamental and universally applicable. This is to reduce the reliance on third-party chemistry software for defining molecular systems. Additionally, future updates will aim to refine the process of positioning molecules within the simulation box—a task that currently demands significant computational resources. The objective is to reduce the program's execution time while maintaining the production of valid molecular systems.

This will necessitate the development of more sophisticated particle distribution algorithms, especially to accommodate the diverse spatial conformations of organic molecules. For example, placing a highly branched hydrocarbon in the simulation box requires different algorithms compared to less complex molecules. Such improvements are expected to enhance the efficiency and robustness of SMI2PDB, particularly in unattended operations, minimizing potential issues and maximizing reliability.

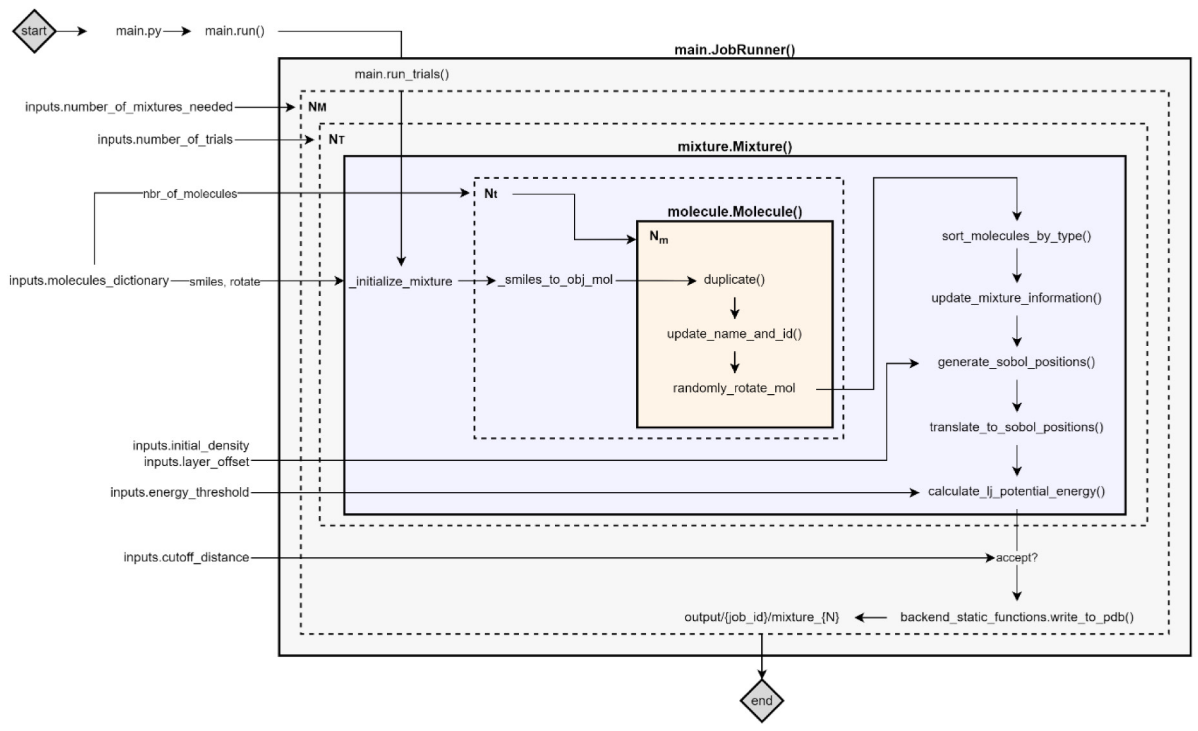

Fig. 3. A diagram depicting how the program elements' (files, instances, functions, and variables) calls to generate molecular systems.

CRediT authorship contribution statement

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

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] 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.
[3] M.J. Abraham, T. Murtola, R. Schulz, S. Páll, J.C. Smith, B. Hess, E. Lindahl, GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers, SoftwareX 1 (2015) 19–25.
[4] S. Sharma, Molecular Dynamics Simulation of Nanocomposites using BIOVIA Materials Studio, Lammps and Gromacs, Elsevier, 2019.
[5] L. Martínez, R. Andrade, E.G. Birgin, J.M. Martínez, PACKMOL: A package for building initial configurations for molecular dynamics simulations, J. Comput. Chem. 30 (13) (2009) 2157–2164.
[6] N.M. O'Boyle, M. Banck, C.A. James, C. Morley, T. Vandermeersch, G.R. Hutchison, Open babel: An open chemical toolbox, J. Cheminform. 3 (2011) 1–14.
[7] A. Stukowski, Visualization and analysis of atomistic simulation data with OVITO–the open visualization tool, Model. Simul. Mater. Sci. Eng. 18 (1) (2009) 015012.
[8] D. Weininger, SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules, J. Chem. Inform. Comput. Sci. 28 (1) (1988) 31–36.
[9] 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 Protoc. (2017) 627–641.
[10] G. Landrum, RDKit: A software suite for cheminformatics, Comput. Chem. Predict. Model. Greg Landrum 8 (31.10) (2013) 5281.
[11] S.K. Lam, A. Pitrou, S. Seibert, Numba: A llvm-based Python jit compiler, in: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, 2015, pp. 1–6.
[12] 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.
[13] 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.
[14] 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.