The Journal of Open Source Software

molify: Molecular Structure Interface

Fabian Zills [1]

1 Institute for Computational Physics, University of Stuttgart, 70569 Stuttgart, Germany

DOI: 10.21105/joss.08829

Software

- Review
- Repository
- Archive

Editor: Lucy Whalley

Reviewers:

- @colinbousige
- @csadorf

Submitted: 23 June 2025
Published: 10 December 2025

Summary

The increasing prevalence of Machine-Learned Interatomic Potentials (MLIPs) has shifted requirements for setting up atomistic simulations. Unlike classical force fields, MLIPs primarily require atomic positions and species, thereby removing the need for predefined topology files used for classical force fields in molecular dynamics software like GROMACS (Abraham et al., 2015), LAMMPS (Thompson et al., 2022), ESPResSo (Weik et al., 2019), or OpenMM (Eastman et al., 2024). Consequently, the Atomic Simulation Environment (ASE) (Larsen et al., 2017) has become a popular Python toolkit for handling atomic structures and interfacing with MLIPs, particularly within the material science and soft matter communities. ASE originates from the electronic structure community, which shares the same setup as MLIP-driven studies. In contrast to *ab initio*, MLIPs are much faster and can run on much larger systems, making high-throughput simulations of more complex systems feasible and increasing the need for efficient initial structure generation.

Concurrently, RDKit (Landrum et al., 2023) offers extensive functionality for cheminformatics and manipulating chemical structures. However, standard RDKit workflows are not designed for MLIP-driven simulation, while typical ASE-MLIP workflows may lack rich explicit chemical information such as bond orders or molecular identities, as well as capabilities for generating different conformations or searching substructures.

The `molify` package bridges this gap, providing an interface between RDKit's chemical structure generation and cheminformatics capabilities and ASE's handling of 3D atomic structures. Furthermore, `molify` integrates with PACKMOL (Martínez et al., 2009) to facilitate the creation of complex, periodic simulation cells with diverse chemical compositions, all while preserving crucial chemical connectivity information. In addition, `molify` simplifies the representation of molecular structures as graphs using NetworkX (Hagberg et al., 2008), e.g., enabling traversing or comparing them. Lastly, the combination of these packages enables selection and manipulation of atomistic structures based on chemical knowledge rather than manual index handling. While designed for MLIP data, the usage of `molify` is not limited and can be expanded, e.g., by utilizing the bond order information in other ASE-based workflows for classical MD simulations or integrating with machine-learning driven bond order predictions.

Statement of need

`molify` serves as a vital link between RDKit, ASE, NetworkX, and PACKMOL, designed to aid working with molecular structures in systems without dedicated topology files and formats. While its core function is to interface these tools, it thereby unlocks new capabilities and significantly reduces the manual coding and data wrangling typically required for preparing and analyzing molecular simulations. For example, ASE has no tools for handling topological information such as bonds or molecular identities, while RDKit cannot natively interface with MLIPs.

`molify` simplifies workflows that previously involved laborious tasks such as sourcing individual structure files from various databases (e.g., the Materials Project (Jain et al., 2013) or the ZINC database (Tingle et al., 2023)) and custom setups of simulation cells. With `molify`, more complex and chemically diverse simulation cells are easier to set up and process.

One challenge in MLIP-driven simulations is the post-simulation identification and analysis of molecular fragments or chemical changes, as explicit topological information is not available and changes in connectivity can occur. `molify` addresses this by enabling the use of RDKit's powerful SMILES (Weininger, 1988)/SMARTS-based substructure searching on ASE structures. In addition, the resulting molecular graph can be exported to a NetworkX (Hagberg et al., 2008) object for further analysis. This selection and handling allows for similar functionality as is provided by the MDAnalysis (Gowers et al., 2016) atom selection language, designed for simulations with a fixed topology.

Features and Implementation

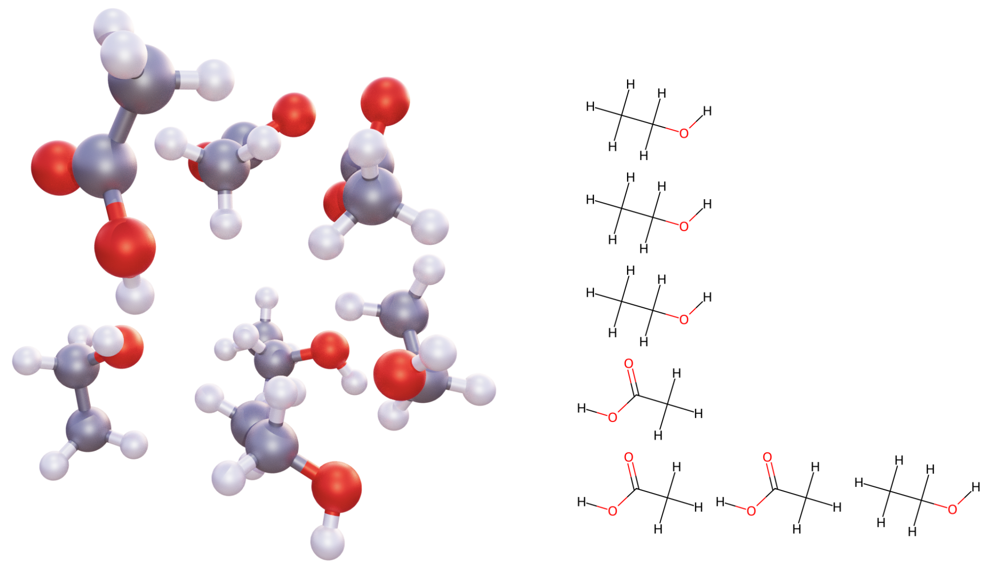

Figure 1: Visualization of a 3D structure from ASE, visualized with ZnDraw (Elijošius et al., 2024) (left) and its corresponding RDKit 2D chemical structure representation (right).

The generation of atomic configurations in `molify` is centered around SMILES for defining molecular species. A typical workflow often follows these steps:

1. Generating 3D conformers for individual molecular species from their SMILES using RDKit.
2. Packing these conformers into a simulation box with a target density using PACKMOL and returning the simulation box as an ASE Atoms object.
3. Running MLIP-based simulations using ASE.
4. Post-processing the simulation data by identifying and selecting structures based on SMARTS.

```python
from molify import pack, smiles2conformers
from ase.optimize import LBFGS
from mace.calculators import mace_mp


water = smiles2conformers("O", numConfs=2)
print(water[0].info['connectivity'])
```

```
>>> [(0, 1, 1.0), (0, 2, 1.0)] # (atom_idx1, atom_idx2, bond_order)
ethanol = smiles2conformers("CCO", numConfs=5)
density = 1000  # kg/m^3
box = pack([water, ethanol], [7, 5], density)
print(box)
>>> Atoms(symbols='C10H44O12', pbc=True, cell=[8.4, 8.4, 8.4])
box.calc = mace_mp()  # MLIP calculator
opt = LBFGS(box)
opt.run(fmax=0.01)
```

All ASE Atoms objects generated or processed by `molify` store `connectivity` information (bonds and their orders) within the `ase.Atoms.info` dictionary. If this information is available, `molify` uses it to convert between ASE, NetworkX and RDKit. If an ASE structure is converted to an RDKit molecule without pre-existing connectivity, `molify` leverages RDKit's bond perception algorithms (Kim & Kim, 2015) to estimate this information.

A visualisation of the 2D and 3D structure from the simulation is shown in Figure 1.

```python
from molify import ase2rdkit
from rdkit.Chem import Draw

mol = ase2rdkit(box)
img = Draw.MolToImage(mol)
```

Additionally, `molify` can convert structures to and from NetworkX graphs, enabling graph-based algorithms such as traversal, isomorphism checks, or component analysis:

```python
from molify import ase2networkx

graph = ase2networkx(box)  # NetworkX graph with atomic numbers and bond orders
```

This bidirectional conversion capability allows the use of RDKit's chemical analysis tools together with ASE for MLIP-based simulations.

For example, after atomic positions in an ASE Atoms object are updated, `molify` can convert this structure back to an RDKit molecule allowing for further analysis or visualization. One common example is the extraction of substructures based on SMILES or SMARTS to track their structure and dynamics within a simulation. For example, `molify` streamlines the extraction of the CH_3 alkyl group from the ethanol molecules inside the simulation cell, without manual index lookup.

```python
from molify import get_substructures

frames: list[ase.Atoms] = get_substructures(
  atoms=box,
  smiles="[C]([H])([H])[H]"
)
```

Related software

The functionality of `molify` relies critically on the following packages:

- RDKit: For cheminformatics tasks, SMILES parsing, conformer generation, and substructure searching.
- ASE: For representing and manipulating atomic structures, and interfacing with simulation engines.
- PACKMOL: For optimizing the placement of molecules within a specified simulation box size.
- NetworkX: For the handling and analysis of molecular graphs.

The molify package is currently a crucial part of the following software packages:

- IPSuite: For generating structures for training MLIPs.
- ZnDraw: Interactive generation of simulation boxes and selection of substructures through a graphical user interface inside a web-based visualization package.
- mlipx: Creating initial structures for benchmarking different MLIPs on real-world test scenarios.

The OpenBabel (O'Boyle et al., 2011) package provides similar cheminformatics functionality to RDKit along with extensive file format support. However, OpenBabel is primarily designed as a format conversion tool with a focus on command-line usage and file I/O, while molify is designed for Python-native workflows with in-memory object conversions. Currently, OpenBabel does not provide direct support for ASE Atoms objects, ASE calculators (including MLIPs), or seamless RDKit-ASE interconversion within Python. Furthermore, molify's integration with PACKMOL and NetworkX provides capabilities beyond OpenBabel's core focus on format conversion.

Acknowledgements

F.Z. acknowledges support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) in the framework of the priority program SPP 2363, "Utilization and Development of Machine Learning for Molecular Applications – Molecular Machine Learning" Project No. 497249646. Further funding through the DFG under Germany's Excellence Strategy – EXC 2075 – 390740016 and the Stuttgart Center for Simulation Science (SimTech) was provided.

Abraham, M. J., Murtola, T., Schulz, R., Páll, S., Smith, J. C., Hess, B., & Lindahl, E. (2015). GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, *1–2*, 19–25. https://doi.org/10.1016/j.softx.2015.06.001

Eastman, P., Galvelis, R., Peláez, R. P., Abreu, C. R. A., Farr, S. E., Gallicchio, E., Gorenko, A., Henry, M. M., Hu, F., Huang, J., Krämer, A., Michel, J., Mitchell, J. A., Pande, V. S., Rodrigues, J. P., Rodriguez-Guerra, J., Simmonett, A. C., Singh, S., Swails, J., … Markland, T. E. (2024). OpenMM 8: Molecular Dynamics Simulation with Machine Learning Potentials. *The Journal of Physical Chemistry B*, *128*(1), 109–116. https://doi.org/10.1021/acs.jpcb.3c06662

Elijošius, R., Zills, F., Batatia, I., Norwood, S. W., Kovács, D. P., Holm, C., & Csányi, G. (2024). *Zero Shot Molecular Generation via Similarity Kernels* (No. arXiv:2402.08708). arXiv. https://doi.org/10.48550/arXiv.2402.08708

Gowers, R. J., Linke, M., Barnoud, J., Reddy, T. J. E., Melo, M. N., Seyler, S. L., Domański, J., Dotson, D. L., Buchoux, S., Kenney, I. M., & Beckstein, O. (2016). MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. *Proceedings of the 15th Python in Science Conference*, 98–105. https://doi.org/10.25080/Majora-629e541a-00e

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008, June). Exploring network structure, dynamics, and function using NetworkX. *Proceedings of the 7th Python in Science Conference*. https://doi.org/10.25080/TCWV9851

Jain, A., Ong, S. P., Hautier, G., Chen, W., Richards, W. D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G., & Persson, K. A. (2013). Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL Materials*, *1*(1), 011002. https://doi.org/10.1063/1.4812323

Kim, Y., & Kim, W. Y. (2015). Universal Structure Conversion Method for Organic Molecules: From Atomic Connectivity to Three-Dimensional Geometry. *Bulletin of the Korean Chemical Society*, *36*(7), 1769–1777. https://doi.org/10.1002/bkcs.10334

Landrum, G., Tosco, P., Kelley, B., Ric, Cosgrove, D., sriniker, gedeck, Vianello, R., NadineSchneider, Kawashima, E., N, D., Jones, G., Dalke, A., Cole, B., Swain, M., Turk, S., AlexanderSavelyev, Vaucher, A., Wójcikowski, M., … strets123. (2023). *Rdkit/rdkit: 2023_03_2 (Q1 2023) Release*. Zenodo. https://doi.org/10.5281/zenodo.8053810

Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Dułak, M., Friis, J., Groves, M. N., Hammer, B., Hargus, C., Hermes, E. D., Jennings, P. C., Jensen, P. B., Kermode, J., Kitchin, J. R., Kolsbjerg, E. L., Kubal, J., Kaasbjerg, K., Lysgaard, S., … Jacobsen, K. W. (2017). The atomic simulation environment—a Python library for working with atoms. *Journal of Physics: Condensed Matter*, *29*(27), 273002. https://doi.org/10.1088/1361-648X/aa680e

Martínez, L., Andrade, R., Birgin, E. G., & Martínez, J. M. (2009). PACKMOL: A package for building initial configurations for molecular dynamics simulations. *Journal of Computational Chemistry*, *30*(13), 2157–2164. https://doi.org/10.1002/jcc.21224

O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., & Hutchison, G. R. (2011). Open Babel: An open chemical toolbox. *Journal of Cheminformatics*, *3*(1), 33. https://doi.org/10.1186/1758-2946-3-33

Thompson, A. P., Aktulga, H. M., Berger, R., Bolintineanu, D. S., Brown, W. M., Crozier, P. S., Veld, P. J. in 't, Kohlmeyer, A., Moore, S. G., Nguyen, T. D., Shan, R., Stevens, M. J., Tranchida, J., Trott, C., & Plimpton, S. J. (2022). *LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales*. *271*, 108171. https://doi.org/10.1016/j.cpc.2021.108171

Tingle, B. I., Tang, K. G., Castanon, M., Gutierrez, J. J., Khurelbaatar, M., Dandarchuluun, C., Moroz, Y. S., & Irwin, J. J. (2023). ZINC-22-A Free Multi-Billion-Scale Database of Tangible Compounds for Ligand Discovery. *Journal of Chemical Information and Modeling*, *63*(4), 1166–1176. https://doi.org/10.1021/acs.jcim.2c01253

Weik, F., Weeber, R., Szuttor, K., Breitsprecher, K., de Graaf, J., Kuron, M., Landsgesell, J., Menke, H., Sean, D., & Holm, C. (2019). ESPResSo 4.0 – an extensible software package for simulating soft matter systems. *The European Physical Journal Special Topics*, *227*(14), 1789–1816. https://doi.org/10.1140/epjst/e2019-800186-9

Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, *28*(1), 31–36. https://doi.org/10.1021/ci00057a005