The Journal of Open Source Software

pycopm: An open-source tool to tailor OPM Flow geological models

David Landa-Marbán [1¶]

1 NORCE Research AS, Bergen, Norway ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Andrew Walker

Reviewers:

- @frank1010111
- @nasserma

Submitted: 13 October 2025
Published: unpublished

Summary

Reservoir simulations help the energy industry make better decisions by predicting how fluids like oil, gas, water, hydrogen, and carbon dioxide will flow underground. To keep these predictions accurate, engineers often need to update geological models quickly as new information becomes available. `pycopm` is a tool designed to make this process faster and easier. It allows users to adjust geological models in several ways, such as simplifying complex grids, focusing on specific parts of a reservoir, or changing the shape and position of the model. These capabilities help engineers test different scenarios efficiently. Although `pycopm` was first used on two well-known public datasets, it has since become useful in many other situations because of its easy-to-use features and recent extensions. Today, it supports studies involving model refinement, comparing coarse and detailed models, analyzing interactions between nearby sites, and speeding up troubleshooting in large simulations.

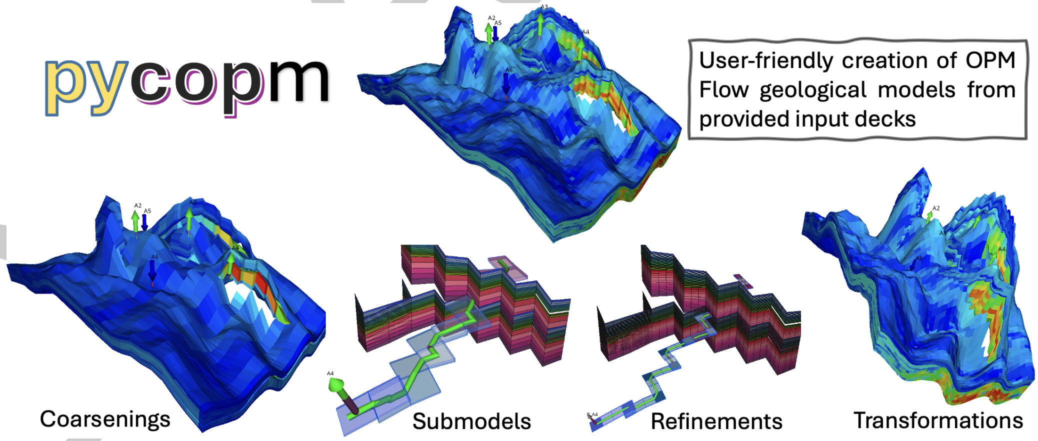

Figure 1: Graphical representation of pycopm's functionality (here are details to reproduce this).

Statement of need

The first step in reservoir simulations is to chose a simulation model, which serves as the computational representation of a geological model, incorporating properties such as heterogeinity, physics, fluid properties, boundary conditions, and wells. Once the spatial model is designed, it is discretized into cells containing average properties of the continuous reservoir model. All this information is then comunicated to the simulator, which internally solves conservation equations (mass, momentum, and energy) and constitutive equations (e.g., saturation functions, well models) to perform the predictions. OPM Flow is an open-source simulator for subsurface applications such as hydrocarbon recovery, CO_2 storage, and H_2

The Journal of Open Source Software

storage (Rasmussen et al., 2021). The input files of OPM Flow follows the standar-industry Eclipse format. The different functionality is defined using keywords in a main input deck with extension .DATA and additional information is usually added in additional .INC files such as tables (saturation functions, PVT) and the grid discretization. We refer to the OPM Flow manual OPM Flow manual for an introduction to OPM Flow and all suported keywords.

Simulation models can be substantial, typically encompassing millions of cells, and can be quite complex due to the number of wells and faults, defined by cell indices in the x, y, and z direction (i,j,k nomenclature). While manually modifying small input decks is feasible, it becomes impractical for large models. In addition, these models commonly rely on corner-point grids defined through pillars and horizons, and they may include further geometric modifications specified through deck keywords. Such representations are not intuitive to manipulate, particularly for users who are not familiar with the internal structure of simulation decks.

These challenges inspired the development of `pycopm`, a user-friendly Python tool designed to taylor geological models from provided input decks. `pycopm` is intended for researchers, engineers, and students who need to apply model transformations such as coarsening, refinement, submodel extraction, and structural adjustments. The coarsening and refinement capabilities are especially relevant in current workflows, since multi-fidelity modeling has become an active and widely adopted research direction that requires the flexibility to generate models with different levels of complexity.

State of the field

Two key properties in a reservoir model are its storage capacity, measured by pore volume, and the ability of fluids to flow between cells, known as transmissibilities. Therefore, these properties must be properly handled when generating a new model. While grid refinements and transformations do not pose a significant issue, submodels and grid coarsening present challenges due to lack of unique methods for adressing these properties. In other words, the approach depends on the specific model, and while there are a few methods in literature, this remains an active are of research.

opm-upscaling is part of the OPM initiative, and provides a set of C++ tools focused on single-phase and steady-state upscaling of capillary pressure and relative permeability. However, it does not include functionality for grid refinement or affine transformations, and its upscaling routines operate mainly on the grid structure. As a result, users must manually adjust the remaining components of the deck to match the new i, j, and k indices. Examples include updating the locations of wells, numerical aquifers, and other model elements.

ResInsight is an open-source C++ tool designed for postprocessing reservoir models and simulations. It can export modified simulation grids and supports operations such as grid refinement and submodel extraction. Nevertheless, it does not generate an updated input deck reflecting the modified i, j, and k coordinates. It also lacks built-in capabilities for model coarsening or for applying general affine transformations. Users on macOS may encounter installation challenges, and the software has difficulty handling models with very small cell dimensions (below 1 mm) or with very large cell counts (greater than 100 million).

To the author's knowledge, prior to the development of `pycopm` there was no integrated Python-based solution that combined coarsening, refinement, submodel extraction, and geometric transformations for modifying geological models compatible with OPM Flow. Python offers a significantly more accessible environment than C++, which lowers the entry barrier for researchers, engineers, and students who need flexible model manipulation tools. `pycopm` offers flexibility in selecting different approaches, allowing end-users to compare methods and choose the one that best fits their needs. For additional information about the different approaches implemented in `pycopm` for coarsenings, refinements, submodels, and transformations, see the theory in `pycopm`'s documentation.

Software design

pycopm leverages well-stablished and excellent Python libraries. The Python package numpy (Harris et al., 2020) forms the basis for performing arrays operations. The pandas package (The pandas development team, 2025), is used for handling cell clusters, specifically employing the methods in pandas.Series.groupby. The Shapely package (Gillies et al., 2025), particularly the contains_xy, is fundamental for submodel implementation to locate grid cells within a given polygon. To parse the output binary files of OPM Flow, the opm Python libraries is utilized. The primary methods developed in pycopm include handling of corner-point grids, upscaling transmissibilities in complex models with faults (non-neighbouring connections) and inactive cells, projecting pore volumes on submodel boundaries, interpolating to extend definition of i,j,k dependent properties (e.g., wells, faults) in grid refinement, and parsing and writing input decks.

Interaction with the tool is performed through a terminal executable named pycopm, which provides a set of command-line flags (27 at the time of writing; see the online documentation for the current list). These flags control the desired functionality, such as specifying the input deck, defining how the model should be modified, and selecting the output file name. This design enables users to chain multiple operations by further editing the generated decks. For example, a user may refine a model first and subsequently extract a submodel. An illustrative example is provided in test_4_submodel.py in the project repository. Advanced users who are familiar with Python can access the underlying functionality directly through Python scripts. This provides greater flexibility for integrating the tool into more sophisticated workflows and for customizing model transformations to meet specific research or engineering needs.

Research impact statement

pycopm is already being adopted across several projects at NORCE Research AS and Equinor ASA. The user base is growing, with increased activity observed across multiple locations. GitHub traffic insights also show a steady rise in repository clones, indicating expanding interest and usage.

The software has supported several research publications, including:

- Sandve et al. (2024), where it was used to coarsen the Drogon model for history matching,
- Nilsen et al. (2025), which applied coarsening of the same model to optimize operations using wind energy,
- Sandve et al. (2025), where submodel extraction and coarsening were applied to the Troll aquifer model to analyze pressure interference, and
- Landa-Marbán et al. (2025), which used coarsening of the Troll aquifer model to optimize well placement in CO_2 storage simulations.

The pycopm is part of the software suite developed within the Centre for Sustainable Subsurface Resources and maintained under the cssr-tools GitHub organization. A key objective of these tools is to support research outputs that adhere to the FAIR principles (Findable, Accessible, Interoperable, Reusable) originally formalized in Wilkinson et al. (2016). These principles have not been consistently implemented in subsurface research in recent years (Liu et al., 2025), limiting the long-term impact and reproducibility of published results. To address this, significant effort has been dedicated to building comprehensive online documentation that enables users to reproduce figures, tables, and computational workflows from recent publications. For example, the TCCS-13 documentation includes step-by-step terminal commands required to generate the results presented in (Landa-Marbán et al., 2025). This ensures that published work is not only transparent but also directly reusable by other researchers, enhancing scientific rigor and accelerating future developments.

Looking ahead to increase the research impact, the plan for pycopm's future development includes extending its functionality to support additional keywords from input decks beyond those in geological models, which pycopm has been sucessfully tested on (Drogon, Norne, Smeaheia, SPE10, Troll aquifer model). This support will be added as pycopm is applied in further models, and external contributions to the tool are welcomed. Additionally, extending pycopm's capabilities includes implementing a feature to generate a single input deck by combining geological models from different input decks.

AI usage disclosure

No generative AI tools were used in the development of this software. Microsoft M365 Copilot (powered by a GPT-5–class large language model developed by Microsoft) was used to check and improve the writing of this manuscript.

Acknowledgements

The author acknowledges funding from the Center for Sustainable Subsurface Resources (CSSR), grant nr. 331841, supported by the Research Council of Norway, research partners NORCE Norwegian Research Centre and the University of Bergen, and user partners Equinor ASA, Harbour Energy, Sumitomo Corporation, Earth Science Analytics, GCE Ocean Technology, and SLB Scandinavia. The author also acknowledges funding from the Expansion of Resources for CO2 Storage on the Horda Platform (ExpReCCS) project, grant nr. 336294, supported by the Research Council of Norway, Equinor ASA, A/S Norske Shell, and Harbour Energy Norge AS. The author extends sincere gratitude to Tor Harald Sandve and Sarah Gasda for their valuable insights that significantly enhanced the development of pycopm.

References

Gillies, S., Wel, C. van der, Van den Bossche, J., Taves, M. W., Arnott, J., Ward, B. C., & others. (2025). *Shapely* (Version 2.0.7). https://doi.org/10.5281/zenodo.5597138

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*, 357–362. https://doi.org/10.1038/s41586-020-2649-2

Landa-Marbán, D., Sandve, T. H., & Gasda, S. E. (2025). *A coarsening approach to the troll aquifer model*. https://arxiv.org/abs/2508.08670

Liu, N., Both, J. W., Ersland, G., Nordbotten, J. M., & Fernø, M. (2025). *Trends in porous media laboratory imaging and open science practices*. https://arxiv.org/abs/2510.05190

Nilsen, M. M., Lorentzen, R. J., Leeuwenburgh, O., Stordal, A. S., & Barros, E. (2025). Closed-loop workflow for short-term optimization of wind-powered reservoir management. *Cleaner Energy Systems*, *12*, 100213. https://doi.org/https://doi.org/10.1016/j.cles.2025.100213

Rasmussen, A. F., Sandve, T. H., Bao, K., Lauser, A., Hove, J., Skaflestad, B., Klöfkorn, R., Blatt, M., Rustad, A. B., Sævareid, O., Lie, K.-A., & Thune, A. (2021). The open porous media flow reservoir simulator. *Computers & Mathematics with Applications*, *81*, 159–185. https://doi.org/10.1016/j.camwa.2020.05.014

Sandve, T. H., Boon, W., Landa-Marbán, D., Tveit, S., & Gasda, S. E. (2025). *Multi-scale simulation strategies for managing pressure interference in multi-site CO2 storage in large regional aquifers*. *2025*(1), 1–4. https://doi.org/https://doi.org/10.3997/2214-4609.202521134

Sandve, T. H., Lorentzen, R. J., Landa-Marbán, D., & Fossum, K. (2024). *Closed-loop reservoir management using fast data-calibrated coarse models*. *2024*(1), 1–15. https://doi.org/https://doi.org/10.3997/2214-4609.202437071

The pandas development team. (2025). *Pandas-dev/pandas: pandas* (Version v2.3.3). Zenodo. https://doi.org/10.5281/zenodo.17229934

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18