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:

Submitted: 13 October 2025
Published: unpublished

Summary

Reservoir simulations are essential for optimizing resource management, facilitating accurate predictions and informed decision-making in the energy industry. The ability to rapidly update geological models is crucial in the dynamic field of reservoir management. `pycopm` serves as a geomodeling tool for quickly tailoring geological models. Its functionality includes grid coarsening, extraction of selected zones in the reservoir (submodels), localized grid refinements, and affine transformations on grid spatial locations, including scaling, rotation, and translation. The impact of `pycopm` is expected to extend far beyond its initial application to coarse two wide recognized open datasets (Norne and Drogon) in a history matching application (Sandve et al., 2024), due to its user-friendly features (i.e., generation of different models from provided input files via command flags) and recent extension to refinements, submodels, and transformations (e.g., studies focusing on grid refinement, upscaling/coarsening approaches, pressure interference between site models within a regional aquifer, and faster debugging of numerical issues in large models).

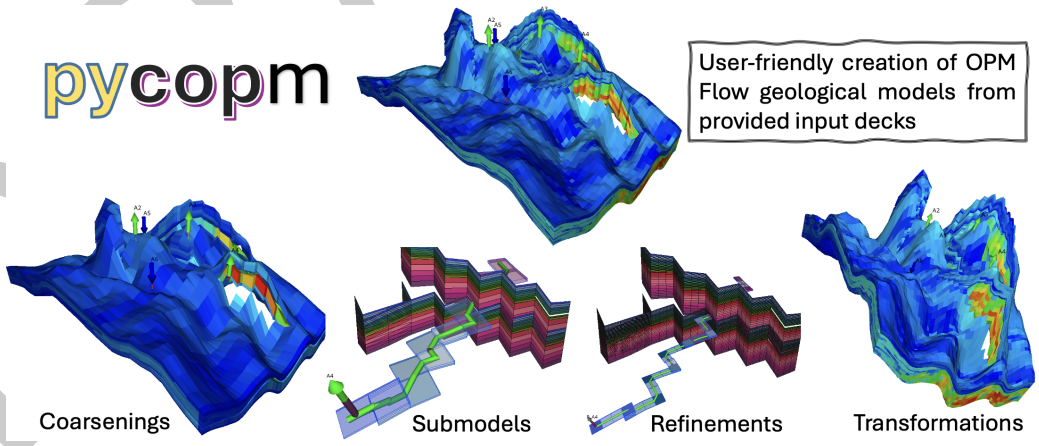

Figure 1: Graphical representation of pycopm's functionality (here are details to reproduce this).

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 func-

tions, 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 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. ResInsight is an open-source C++ tool designed for postprocessing reservoir models and simulations. It provides capabilities to export simulation models, including grid refinements and submodels. However, to the author's knowledge, the main input deck with the modified i,j,k locations is not generated, and there is no build-in functionality to coarsen models or apply general afine transformations. Additionally, the installation of ResInsight for macOS users is not straightforward, and the tool struggles with models that have small grid sizes (below 1 mm) and cases with a large number of cells (over 100 million). These challenges inspired the development of `pycopm`, a user-friendly Python tool designed to taylor geological models from provided input decks.

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. To address this, `pycopm` offers flexibility in selecting different approaches, allowing end-users to compare methods and choose the one that best fits their needs. This also provides an oportunity to test novel approaches. For additional information about the different approaches implemented in `pycopm` for coarsenings, refinements, submodels, and transformations, see the theory in `pycopm`'s documentation.

`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 (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 resdata or opm Python libraries are utilized. While resdata is easier to install on macOS, opm seems to be faster and capable of handling large cases (tested with more than 100 million cells). 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.

Outlook

`pycopm` is designed for use by researchers, engineers, and students. It is currently utilized in various projects at NORCE Research AS, with the number of users in different locations expected to grow. Looking ahead, 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.

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

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., 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

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