pyopmspe11: A Python framework using OPM Flow for the SPE11 benchmark project

David Landa-Marbán [1¶] **and Tor H. Sandve** [1]

1 NORCE Norwegian Research Centre AS, Bergen, Norway ¶ Corresponding author

DOI: 10.21105/joss.07357

Software

- Review
- Repository
- Archive

Editor: Rachel Wegener

Reviewers:

- @MatthewFlamm
- @gassmoeller

Submitted: 28 June 2024
Published: 28 January 2025

Summary

The imperative to achieve climate change goals and the increasing worldwide demand for energy have made geological carbon storage (GCS) technology more relevant today. Since utilizing computational models is essential for planning large-scale GCS projects, it is crucial to benchmark simulation tools to enhance confidence in their results. Inspired by a recent validation study for laboratory-scale CO_2 storage (Flemisch et al., 2024), a new comparative solution project (CSP) was launched to simulate both lab- and field-scale CO_2 storage (Nordbotten et al., 2024). This project is called the 11th Society of Petroleum Engineers CSP, and we refer to it as the SPE11 benchmark. The main objective for the SPE11 benchmark is to provide a common platform and reference case for numerical simulation of GCS. A community effort was run by the "Early Access Team" to create utility scripts and input files for popular simulators to make participation more accessible. As part of the "Early Access Team", we have developed and made open the `pyopmspe11` tool which facilitates reproducible solutions to the SPE11 benchmark. This tool serves as a common starting point for developing and testing new GCS simulation technology. Due to its user-friendly functionality (e.g., generation of different types of grids at different grid resolutions, flexibility to choose different rock and fluid properties, flexibility to define well/source locations and schedule for operations), it is expected that its impact will extend far beyond the initial benchmark study (e.g., studies focusing on grid refinement, upscaling/coarsening approaches, numerical solvers, optimization/history matching techniques).

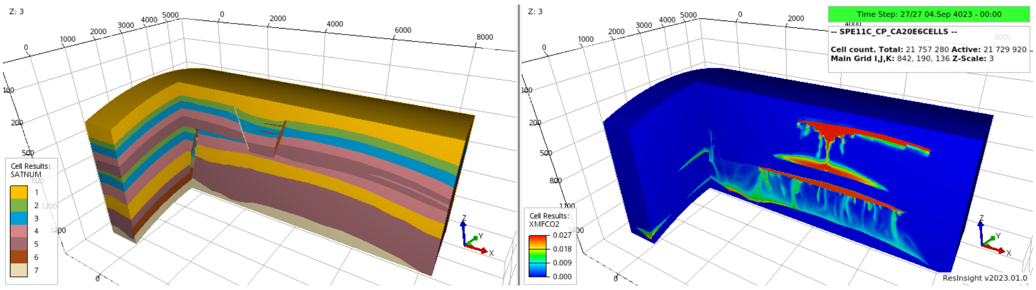

Figure 1: Generated model by the configuration file `spe11c_cp_ca20e6cells.txt` in the examples folder (the model corresponds to the SPE11C Case using a corner-point grid with 21729920 active cells).

Statement of need

Geological carbon storage (GCS) applications benefit from both commercial and open-source simulators. OPM Flow is an open-source simulator for subsurface applications such as hydrocarbon recovery, CO_2 storage, and H_2 storage. The typical workflow in GCS simulations starts with defining the simulation model (e.g., grid, heterogeinity, physics, fluid properties, boundary conditions, wells), then setting the simulator parameters (e.g., tolerances, linear solvers, partition algorithms), then running the simulation, and finally visualization/analysis of the simulation results (e.g., CO_2 plume distance to the boundaries, caprock integrity). Here we refer to the first two steps as preprocessing and the final step as postprocessing. Notable works are available in JOSS for pre-/postprocessing of simulation data, e.g., Beucher et al. (2019), Sullivan & Kaszynski (2019), Fraters et al. (2024), Kaus et al. (2024). However, preprocessing and postprocessing can be challenging for everyone even if you know what you are doing. Additionally, setting up and running simulations requires computational expertise. To bridge this gap, developers can simplify the setup of numerical studies by using user-friendly approaches, such as configuration files. This not only ensures reproducibility of results but also facilitates flexible testing of different simulator parameters and allows for easy extension to further studies.

Based on the acquired knowledge by contributing to other open-source projects such as OPM Flow, then we have developed and made open the `pyopmspe11` tool which facilitates reproducible solutions to the SPE11 benchmark, which focus on GCS at different scales (Nordbotten et al., 2024). A previous benchmark study for GCS can be found in Class et al. (2009). One key difference of the SPE11 benchmark from the benchmark in Class et al. (2009) is that no specific size and type of grids were given in the description, i.e., one of the main task for the SPE11 benchmark participants was to create suitable grids (e.g., structured grids such as Cartesian or unstructured grids such as corner-point grids) and to run the cases, i.e., computational grids. To ease the comparison of results between participants, the SPE11 benchmark organizers requested the reporting of the spatial maps to follow a specific format using Cartesian grids with fixed cell sizes, i.e., reporting grids. The participants were encouraged to share data (e.g., input decks, code, submitted results), with the opportunity to store the data for open access. This is where developing tools that made all steps reproducible (i.e., preprocessing and postprocessing) became handy, and for this benchmark study, one available tool is `pyopmspe11`. Examples of pre-/postprocessing simulation tools which also have application in GCS and rely on the OPM Flow simulator include: `pyopmnearwell` (Landa-Marbán & pschultzendorff, 2023) and `expreccs` (Landa-Marbán, 2024). The former focuses on near well dynamics, while the latter on seamless, dynamic, and non-invasive exchange of pressure-related information between local and regional scales.

`pyopmspe11` is a simplified and flexible Python tool to execute the three cases in the SPE Comparative Solution Project using configuration files. `pyopmspe11` relies on the OPM Flow numerical simulator (Rasmussen et al., 2021), where the implementation of the CO_2 model can be found in Sandve et al. (2021). The primary contribution of `pyopmspe11` lies in its data preprocessing and postprocessing capabilities. It offers flexibility in generating various types of grids, including Cartesian, tensor, and corner-point grids. These grids adhere to the standard industry format (i.e., Eclipse grid format), making them compatible not only with OPM Flow but also with other simulators. Here, we mention two existing widely-used visualization/postprocessing software for OPM Flow: ParaView and ResInsight. While these tools are very useful, to the authors knowledge, there are no available built-in options via the GUIs of these tools to handle all necessary postprocessing to generate all data reporting as required in the SPE11 benchmark study. Instead, this could be achieved in these tools by code development using the Python interface. The benefit of having a specialized tool like pyopmspe11, which integrates all stages from preprocessing to simulation and postprocessing, is that it simplifies usage.

`pyopmspe11` supports varying resolutions, having been tested to generate up to 160 million cells. In the context of data postprocessing, `pyopmspe11` not only generates the necessary reporting data as specified by the benchmark, but it also produces .png figures for rapid inspection of individual simulations and for making comparisons between different runs (e.g., to assess sensitivities). The postprocessing methods efficiently interpolate quantities over time and map non-overlapping cell values (both intensive and extensive quantities) between the computational grid and the reporting grid. The Python package Scipy (Virtanen et al., 2020), specifically the interp1d Class, is used for the time interpolation. The Python package Shapely (Gillies et al., 2024), specifically the Polygon Class, is the base for the developed methods in `pyopmspe11` to handle the mapping from the computational grid to the reporting grid.

Outlook

`pyopmspe11` is designed for use by researchers, engineers, and students. During the preliminary intercomparison workshops for the benchmark study, the authors received positive feedback about the framework. Some participant groups have utilized `pyopmspe11`, particularly for grid generation. Additionally, the authors have been contacted to provide specific support on how to use the tool for setting up further studies, such as optimizing injection strategies. Looking ahead, the plan for `pyopmspe11`'s future development includes extending its functionality to support the generation of input decks to run simulations of physical models available in OPM Flow in addition to CO_2 storage (e.g., hydrogen storage, salt precipitation, biofilm effects).

Acknowledgements

The authors acknowledge 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 authors also acknowledge funding from the HPC Simulation Software for the Gigatonne Storage Challenge project, grant nr. 622059, supported by Equinor ASA and CLIMIT DEMO/Gassnova.

References

Beucher, R., Moresi, L., Giordani, J., Mansour, J., Sandiford, D., Farrington, R., Mondy, L., Mallard, C., Rey, P., Duclaux, G., Kaluza, O., Laik, A., & Morón, S. (2019). UWGeodynamics: A teaching and research tool for numerical geodynamic modelling. *Journal of Open Source Software*, *4*(36), 1136. https://doi.org/10.21105/joss.01136

Class, H., Ebigbo, A., Helmig, R., Dahle, H. K., Nordbotten, J. M., Celia, M. A., Audigane, P., Darcis, M., Ennis-King, J., Fan, Y., Flemisch, B., Gasda, S. E., Jin, M., Krug, S., Labregere, D., Naderi Beni, A., Pawar, R. J., Sbai, A., Thomas, S. G., … Wei, L. (2009). A benchmark study on problems related to CO2 storage in geologic formations. *Computational Geosciences*, *13*(4), 409–434. https://doi.org/10.1007/s10596-009-9146-x

Flemisch, B., Nordbotten, J. M., Fernø, M., Juanes, R., Both, J. W., Class, H., Delshad, M., Doster, F., Ennis-King, J., Franc, J., Geiger, S., Gläser, D., Green, C., Gunning, J., Hajibeygi, H., Jackson, S. J., Jammoul, M., Karra, S., Li, J., … Zhang, Z. (2024). The FluidFlower validation benchmark study for the storage of CO_2. *Transport in Porous Media*, *151*(5), 865–912. https://doi.org/10.1007/s11242-023-01977-7

Fraters, M. R. t., Billen, M. I., Gassmöller, R., Saxena, A., Heister, T., Li, H., Douglas, D., Dannberg, J., Bangerth, W., & Wang, Y. (2024). The geodynamic world builder: A planetary structure creator for the geosciences. *Journal of Open Source Software*, *9*(101), 6671. https://doi.org/10.21105/joss.06671

Gillies, S., Wel, C. van der, Van den Bossche, J., Taves, M. W., Arnott, J., Ward, B. C., & others. (2024). *Shapely* (Version 2.0.6). Zenodo. https://doi.org/10.5281/zenodo.13345370

Kaus, B. J. p., Thielmann, M., Aellig, P., Montserrat, A. de, Siena, L. de, Frasukiewicz, J., Fuchs, L., Piccolo, A., Ranocha, H., Riel, N., Schuler, C., Spang, A., & Weiler, T. (2024). GeophysicalModelGenerator.jl: A julia package to visualise geoscientific data and create numerical model setups. *Journal of Open Source Software*, *9*(103), 6763. https://doi.org/10.21105/joss.06763

Landa-Marbán, D. (2024). *Cssr-tools/expreccs: v2024.10* (Version v2024.10). Zenodo. https://doi.org/10.5281/zenodo.14224355

Landa-Marbán, D., & pschultzendorff. (2023). *daavid00/pyopmnearwell: v2023.10* (Version v2023.10). Zenodo. https://doi.org/10.5281/zenodo.10266790

Nordbotten, J. M., Ferno, M. A., Flemisch, B., Kovscek, A. R., & Lie, K.-A. (2024). The 11th Society of Petroleum Engineers Comparative Solution Project: Problem Definition. *SPE Journal*, *29*(05), 2507–2524. https://doi.org/10.2118/218015-PA

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., Gasda, S. E., Rasmussen, A., & Rustad, A. B. (2021). Convective dissolution in field scale CO2 storage simulations using the OPM flow simulator. In N. A. Røkke & K. Knuutila (Eds.), *TCCS-11. CO2 capture, transport and storage. Trondheim 22nd–23rd june 2021. Short papers from the 11th international trondheim CCS conference* (pp. 113–119). SINTEF Academic Press.

Sullivan, C. B., & Kaszynski, A. A. (2019). PyVista: 3D plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK). *Journal of Open Source Software*, *4*(37), 1450. https://doi.org/10.21105/joss.01450

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2