Generation-to-Storage Data Framework for Large-Scale Carbon Sequestration Simulation Datasets

Luiz Fernando C. dos Santos[1], **Gustavo P. Oliveira**[1,2]

[1] TRIL Lab – Department of Scientific Computing – Federal University of Paraíba (UFPB)
João Pessoa – PB – Brazil

[2]Graduate Program in Informatics – Federal University of Paraíba (UFPB)
João Pessoa – PB – Brazil

luiz.costa@academico.ufpb.br, gustavo.oliveira@ci.ufpb.br

Abstract. *Carbon sequestration is a promising CO_2 removal technology, particularly for the oil and gas industry. Consequently, monitoring the CO_2 injected into deep geological formations is crucial for containment and environmental safety. This paper presents an application of data engineering principles to managing large datasets from numerical flow simulations of underground CO_2 injection. By integrating data lifecycle phases with a Medallion-like architecture, the proposed framework delivers consumable assets for physics-informed machine learning experiments. One expects it provides insights into gas plume motion tracking, supporting reservoir monitoring in the post-injection phase of Carbon Capture and Storage (CCS) programs.*

1. Introduction

The global trend toward energy transition and digital transformation in strategic industries imposes challenges on several countries [GCCSI 2024]. For Brazil, in particular, carbon sequestration in deep geological formations is a potential technology for carbon mitigation in the so-called "hard-to-abate" sectors [Weber et al. 2024], requiring computational resources and data engineering protocols.

A key challenge in this context is the efficient and scalable management of large datasets generated from numerical flow simulations of CO_2 storage in depleted reservoirs. Unlike conventional databases, which primarily handle observational or operational data, high-fidelity simulations that predict subsurface gas plume behavior under varying conditions require tailored data pipelines to ensure a seamless framework for future machine-learning-driven analysis.

In the oil and gas sector, where a small-data regime predominates due to operational and economic costs for generating them, structuring databases for model training can be crucial for explaining the subsurface physics. One way to expand this knowledge is to feed emerging data-driven models, such as physics-informed machine learning (PIML), with numerical background data [Shokouhi et al. 2021, Wang et al. 2024]. By embedding physical principles into machine learning frameworks, PIML enhances predictive accuracy while reducing reliance on exhaustive data collection. Thus, a well-designed database pipeline is essential for facilitating machine learning experiments [da Silva et al. 2020]. By addressing these needs, our approach provides a methodology for managing geological CO_2 storage simulations, fostering reproducibility, scalability, and model interpretability in data-driven geoscientific research.

2. Methodology

The proposed methodology is based on a command line interface (CLI) program designed to read and load the simulation setup, run multiple parallel simulations, prevent redundancy, and clean the output for storage. Its goal is to streamline datasets of primary features responsible for the CO_2 plume motion underground, such as gas saturation, and pressure fields. While delivering an end-to-end pipeline encompassing data generation, ingestion, transformation, and storage, this approach offers flexibility for both local and cloud-based data persistence.

The data architecture follows the Medallion three-layers model (Fig. 1). At a "pre-bronze" stage, the user inputs the values of the physical parameters required by the simulator into a configuration file. Then, through design of experiments, those parameters that are admissible to vary in a controlled range are set, thus enabling the system to automatically generate multiple files for batch-mode execution. After all numerical experiments are concluded through `MRST co2-lab`'s solver [Lie et al. 2016] in `Octave` software [Eaton et al. 2023], the `MAT` files generated constitute the bronze layer as primary data. Further processing converts them into structured `JSON` files for facilitated traceability and metadata retrieval (e.g. parameter values, timestamps, execution status), thus shaping the silver layer.

Once the `JSON` files are organized, the system transitions to the storage phase by converting these files into efficient array-like formats natively designed for vectorized computing (currently NumPy's `NPY` or `ZARR` files), improving I/O throughput. Generally, they will be four-dimensional (3D space plus time) datasets that may scale up to millions of data points. Finally, they are either ingested locally or uploaded to a cloud-based service (currently Amazon S3) for immediate consumption. At this point, the gold layer emerges, closing the loop to have an indispensable framework for ensuing research on geological carbon storage in unknown conditions through artificial intelligence models.

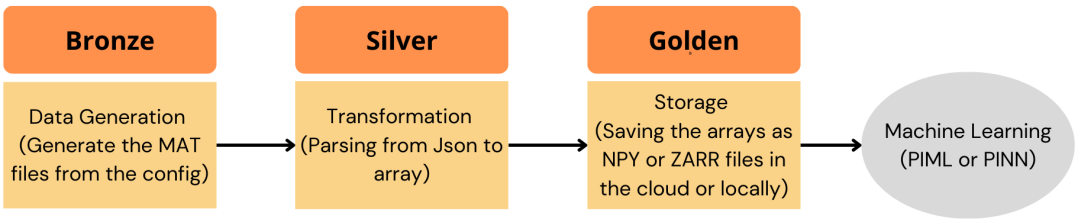

Figure 1. Medallion-inspired architecture flow underlying the proposed CLI.

3. Application

To demonstrate the applicability of the proposed framework (Fig. 2), we first conducted a numerical experiment of CO_2 injection using the widely known by the reservoir modeling community `UNISIM-I-D` model (CEPETRO/UNICAMP), which represents a sandstone reservoir in the Campos Basin. This model consists of a 3D corner-point grid with highly heterogeneous petrophysical properties. The main simulation parameters and their respective values were as follows: reference pressure ($35\,\mathrm{MPa}$), reference temperature

(95.15 K), salt mass fraction for brine (10 %), gas injection rate (1.5×10^5 m^3/year), injection time (15 years), and migration time (30 years).

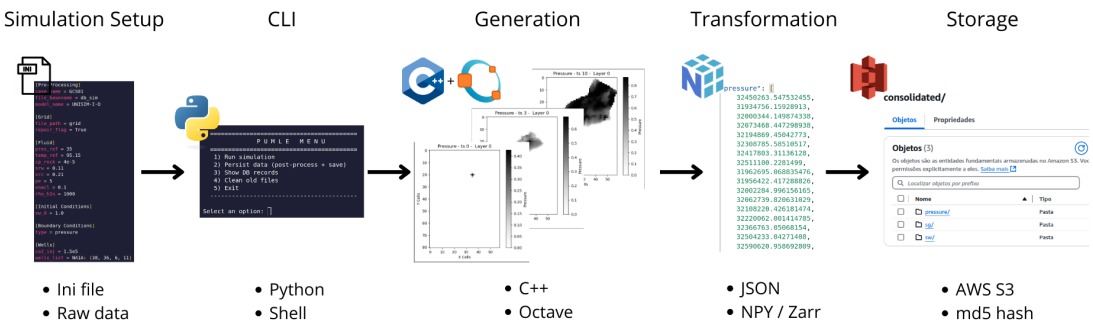

Figure 2. Illustrative scheme of the end-to-end pipeline application.

This single simulation generated a total of 22.5 megabytes of simulation data and approximately 3.3 million data points for just three features (pressure, gas saturation, and brine saturation fields). The simulation took around 16 minutes to complete. Considering this as the closure of the bronze layer and assuming uninterrupted operations until the storage stage, the total time spent streamlining the data was about 16 minutes and 10 seconds. When incorporating a local database, the tasks of ingestion, transformation, and storage extended the processing time to 16 minutes and 36 seconds. In other words, silver and gold layer operations accounted for only 1.37 % of the total experiment time, indicating that, despite the computational intensity of the bronze layer, subsequent layers operate significantly faster.

For a simple comparison, we then repeated the core experiment while varying the reference pressure to 24.5 MPa, 35.5 MPa, and 45.5 MPa. With parallelism, all three additional simulations were executed in approximately 26 minutes, generating nearly 13.2 million data points and totaling 90 Mb of stored datasets. In other words, the new batch of simulations increased the processing time by a factor of 1.56.

It should be noted that the efficiency of the framework is optimized primarily for the silver and gold layers, as the performance of the data generation is entirely dependent on the solver. While the solver's design and numerical implementation directly impact data generation, evaluating this performance is beyond the scope of this study. The parallelism employed here follows an "across the steps" approach, distributing computational jobs at the batch level. Thus, when considering only the post-generation tasks executed for the three simulations, the processing time increased by a factor of 1.18. If we calculate the time of the same 3 simulations without parallelism, we will get a total time of 38 minutes and a processing time increasing by a factor of 1.46.

Regarding the computational infrastructure, all the numerical experiments have been run in a laptop computer with 16Gb RAM / 13th Gen Intel® 16-Core™ i7-13620H processor. The language and software versions were the following: `C++` (C++11 standard), `Python` (3.12.4), `NumPy` (1.26.4), `Octave` (8.4.0).

4. Final considerations

The end-to-end process of generating, transforming, and storing simulation data presented here aims to provide resources for data-driven models capable of describing the CO_2 gas plume under various reservoir conditions and states. By systematically capturing both input parameters and simulation outputs within a unified framework, experts can better understand the interplay between geological properties, injection strategies, and plume dynamics. The resulting datasets are a valuable asset portfolio for further study, given their enriched spatial and temporal granularity.

Leveraging the data consolidated by this pipeline for training machine learning models is a natural extension of this work, with direct implications for decision-making in carbon sequestration programs. By applying advanced predictive algorithms to the multi-dimensional arrays, future studies can uncover hidden correlations between operational strategies—such as injection rates and well placement—and reservoir responses, including plume diffusion and storage efficiency.

Further steps will focus on enhancing the framework with fully queryable datasets and implementing physics-informed machine learning (PIML) frameworks to bridge the gap between purely data-driven approaches and traditional reservoir simulations. These actions will further improve the framework's robustness.

Acknowledgements. This research is part of the PUMLE open-source package developed under the CO2SS project. G.P.O. acknowledges CNPq/Brazil for the funding (Grant no. 405654/2022-7).

References

da Silva, A. C., Lustosa, H. L. S., da Silva, D. N. R., Porto, F. A. M., and Valduriez, P. (2020). Savime: An array dbms for simulation analysis and ml models prediction. *Journal of Information and Data Management*, 11(3).

Eaton, J. W., Bateman, D., Hauberg, S., and Wehbring, R. (2023). *GNU Octave version 8.4.0 manual: a high-level interactive language for numerical computations*.

GCCSI (2024). Global status of ccs 2024: Collaborating for a net-zero future. Technical report, Global CCS Institute.

Lie, K.-A., Nilsen, H. M., Andersen, O., and Møyner, O. (2016). A simulation workflow for large-scale co2 storage in the norwegian north sea. *Computational Geosciences*, 20(3):607–622.

Shokouhi, P., Kumar, V., Prathipati, S., Hosseini, S. A., Giles, C. L., and Kifer, D. (2021). Physics-informed deep learning for prediction of co2 storage site response. *Journal of Contaminant Hydrology*, 241:103835.

Wang, Y.-W., Dai, Z.-X., Wang, G.-S., Chen, L., Xia, Y.-Z., and Zhou, Y.-H. (2024). A hybrid physics-informed data-driven neural network for co2 storage in depleted shale reservoirs. *Petroleum Science*, 21(1):286–301.

Weber, N., de Oliveira, S. B., Cavallari, A., Morbach, I., Tassinari, C. C., and Meneghini, J. (2024). Assessing the potential for co2 storage in saline aquifers in brazil: Challenges and opportunities. *Greenhouse Gases: Science and Technology*, 14(2):319–329.