The Journal of Open Source Software

APyCE: A Python module for parsing corner-point grids and visualizing 3D reservoir models

Mateus Tosta[1*], **Bin Wang**[2*], **and Gustavo P. Oliveira**[1¶]

1 TRIL Lab, Center of Informatics, Federal University of Paraíba, João Pessoa, Brazil **2** State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum-Beijing, Beijing 102249, P. R. China ¶ Corresponding author * These authors contributed equally.

DOI: N/A

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: 01 January 1970

Summary

Data visualization resources are indispensable for any software intended to handle 3D models of oil and gas (O&G) reservoirs (Fanchi, 2005). As never seen before, the groundbreaking power of computer graphics and scientific visualization have brought endless capabilities for geoscientists, engineers, and analysts to understand subsurface phenomena. Multidimensional views greatly improve seismic interpretation, reservoir characterization, and real-time monitoring of producing fields, since high-resolution images and 3D models can reconstitute the physical world almost perfectly.

Many simulators with popularity among the O&G community use discrete formulations based on corner-point grids (Ponting, 1989). Corner-point grids can be considered an international standard to represent all the geologic features found in complex reservoirs, including dips, bends, pinches, and faults. While many tools for post-processing of corner-point grids are available, most of them are independent and have their own internal semantic and object hierarchy, which prevents their full integration. Furthermore, because the interoperability of grid files (in terms of importing/exporting data) among such programs depends on complex programming routines, the representation of modeling entities and features from a given software may be mismatched when read into another. Data integration and data visualization are permanent challenges for software and algorithm developers working on solutions for the O&G sector. In particular, researchers often need visualization tools to communicate their discoveries, make reports with interactive plots, and easily generate publication-ready graphics of post-processed data.

A considerable number of commercial and open-source software capable to deal with corner-point grids for visualization purposes are known. Either they have embedded viewers as secondary functionality or are primarily tailored for visualization. In the first group, we can cite CMG Results©, Schlumberger Petrel©, ESSS Kraken©, and Amarile RE-Studio©; in the second group, SINTEF's MRST (Lie, 2019) and Ceetron Solution's ResInsight are a few options. Minor open-source software projects have been proposed by scholars and independent scientists with an interest in O&G data, but they are usually focused on flow simulations instead of visualization and corner-point grid manipulation.

This paper introduces a lightweight Python-based package for the plotting of reservoir models discretized through Cartesian or corner-point grids called *APyCE*. The objective of this software is to provide a simple pipeline for post-processing through the PyVista module (Sullivan & Kaszynski, 2019) and VTK (Kitware, 2010) exporting for 3D visualization inside Kitware Paraview (Ayachit & Avila, 2015).

Statement of need

APyCE (pronounced as "ah-pees") is an enhanced and extended version of the early project PyGRDECL (Wang, 2018), developed to handle Schlumberger Eclipse© (Schlumberger, 2014) deck files for visualization. APyCE is multi-platform, object-oriented, and easy to use, being prepared to fulfill its objectives with only 4 lines of Python code to be run by whoever with a minimum background in programming. It is recommended for researchers who need to render high-quality figures for inclusion into scientific papers, reports, presentations, handouts, interactive notebooks, and general documents for teaching purposes, or auxiliary tools for data analysis within reservoir modeling or related domains.

APyCE was developed to allow quick analysis and visualization of 3D oil and gas reservoirs, so that the tool can be used by people in academia or industry. It is an open-source tool maintained on GitHub platform, fully accessible to any developer interested to contribute via pull requests. Given its object-oriented structure, the code can be extended and improved for further applications.

Design and implementation

APyCE is developed on top of Eclipse deck files' structure (Pettersen, 2006). However, one should stress that similar topologies are found in most of the input files managed by competing software. Such files are internally divided into sections. Each section admits a broad set of keywords that reflect the model's complexity. The higher is the number of keywords to be processed, the greater is the level of details on stratigraphy, fluid properties, well control, numerical settings, and so on.

In particular, Eclipse's more general files that contain grid specifications have a **`.GRDECL`** extension. Initially, we implemented a parser routine that recognizes only the essential keywords to build either a corner-point (Figure 1) or block-centered grid (Figure 2). All keywords of the file are identified by a regular expression like **`^[A-Z][A-Z0-9]{0,7}`** (Figure 3).

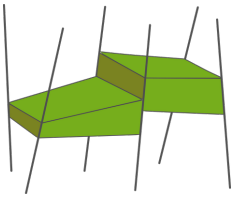

Figure 1: Corner-point grid. This geometry enables us to describe reservoirs with geological realism. Sometimes referred to as "pillar grid" because of ensembles of pillars that span from the top to the bottom of the model, their grid cells are defined by eight paired nodes encountered over four adjacent pillars. Since the nodes are free to slide along the pillars, several features, such as pinch-outs and folds, are accurately and consistently represented, even with degenerated or nonconform cells.

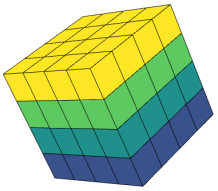

Figure 2: Block-centered grid. Grids like this are Cartesian-like domains specified by multidirectional step sizes and top reference coordinates. The cells are similar to "sugar"'' cuboids that arrange to approximate a regular three-dimensional ("shoebox") volume.

```
SPECGRID
20 20 4 1 F /
COORD
0.6081E+03 -0.1219E+04 0.2249E+04 0.5989E+03 -0.1241E+04 0.2499E+04
0.6504E+03 -0.1204E+04 0.2245E+04 0.6434E+03 -0.1227E+04 0.2492E+04
0.6921E+03 -0.1190E+04 0.2246E+04 0.6879E+03 -0.1213E+04 0.2491E+04
0.7322E+03 -0.1173E+04 0.2248E+04 0.7317E+03 -0.1197E+04 0.2493E+04
0.7871E+03 -0.1150E+04 0.2249E+04 0.7919E+03 -0.1176E+04 0.2496E+04
...
```

Figure 3: Excerpt of a **`.GRDECL`** file for a reservoir model highlighting the *SPECGRID* and *COORD* keywords. The ellipsis is not part of the syntax and here they only illustrate continuity of the file content.

APyCE is developed under a object-oriented paradigm. It has a main class called **`Grid`** where the main methods are contained. A second class called **`VTK`** comprises all the methods related to exporting operations of VTK files. The third class, **`Errors`**, handles error messages. However, the essential functions to load and visualize a reservoir grid of which any user of APyCE will mostly need are included in the former class.

APyCE's workflow, in short, receives a deck-like Eclipse/Builder file and outputs a VTU-like file. After creating the VTK points, the software will retrieve the cells that follow the VTK pattern by dealing with hexahedral objects, as this pattern is different from the one followed by the Eclipse software. Then, by simply calling the **`export_data`** function, the VTK file will be exported for later exhibition in the Paraview software.

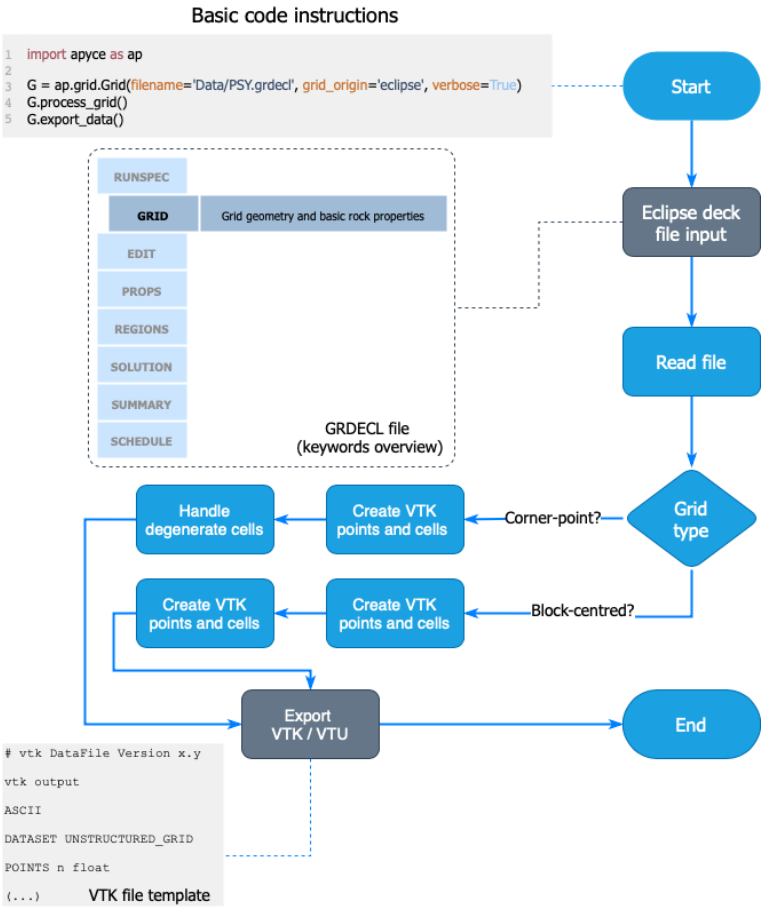

Figure 4: General APyCE's workflow.

APyCE's current version supports the following keywords (those marked with an asterisk are special add-on features not present in PyGRDECL):

- *SPECGRID*: specifies the number of cells in each direction of a corner-point grid.
- *DIMENS*: ditto for block-centered grids.
- *COORD*: specifies the 3D coordinates of the pillars.
- *ZCORN*: indicates the depth of each cell corner over the pillars.
- *PORO*: stores the porosity values per cell.
- *PERMX*: stores the X-direction permeability for each grid cell.
- *PERMY*: ditto for the Y-direction.
- *PERMZ*: ditto for the Z-direction.
- *DX*: specifies the step size over the X-direction for block-centered grids.
- *DY*: ditto Y-direction.
- *DZ*: ditto Z-direction.
- *TOPS*: stores the reference value from the top for each cell in a block-centered grid.
- *INCLUDE**: appends supplementary files to the main file.
- *ACTNUM**: tags grid cells as active/inactive (boolean value).

The example script **`Getting_Started.py`**, which basically reproduces the basic code instructions (see Figure 4) generates multiple VTK file, which can be visually improved from Paraview as depicted in the showcase (Figure 5).

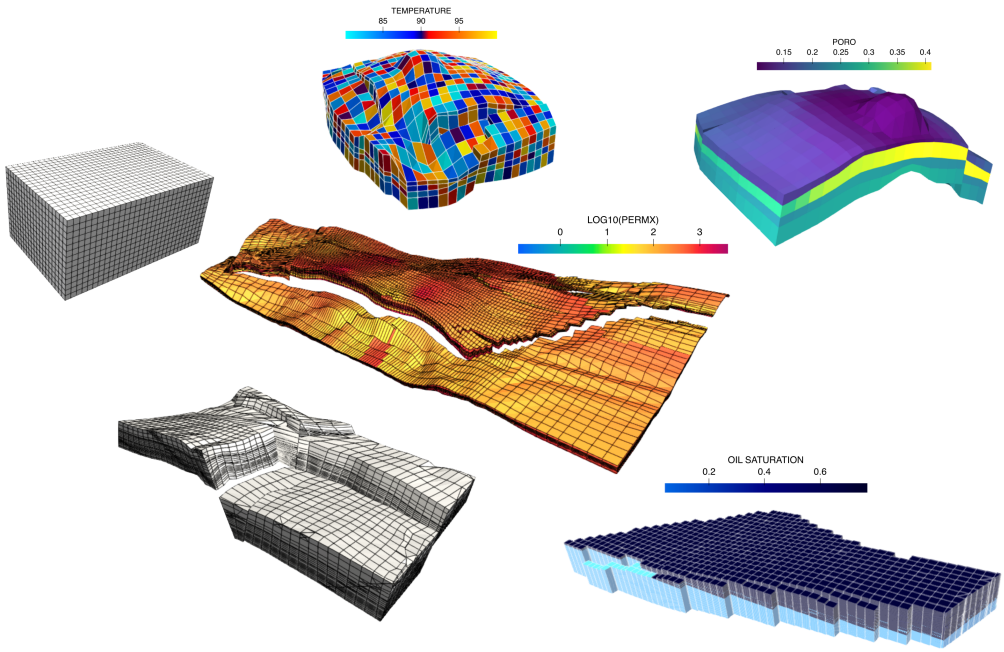

Figure 5: Several models of reservoirs generated by APyCE with different cuts after rendering and filtering operations.

Additional information

APyCE was written in Python 3.8. The choice for this language is justified by the vast amount of modules available to perform scientific computing, signal processing, and graphic manipulation tasks. Another point that strengthens the choice for Python is its simplicity, which reduces software maintenance. Through the available modules and packages, modularized programming and code reuse is encouraged.

As for processing, highly refined grids (hundreds or millions of cells) may demand a robust

hardware ensemble (RAM and processor). Yet less complex grids are manageable affordably. Also, to work properly, APyCE requires the following Python packages: **`NumPy`** (v. 1.19.2 or above), **`VTK`** (v. 8.2.0 or above), **`PyVista`** (v. 0.29 or above), and **`Matplotlib`** (v. 3.4.1 or above).

The most current version was tested on Ubuntu 20.04 distribution with the 5.11.0-25-generic kernel anw with **`PyTest`** (Krekel et al., 2004). All the workflow for exporting/visualizing corner-point grids are fully workable, but block-centered grids still have partial keyword support.

Acknowledgments

G.P.O. and M.T. acknowledge the funding received by the National Council for Scientific and Technological Development (CNPq-Brazil) under the PIBIC/UFPB Program 2020-2021, grant no. PVL11528-2020.

References

Ayachit, U., & Avila, L. S. (2015). *The ParaView guide: Updated for ParaView version 4.3*. Kitware. ISBN: 9781930934306

Fanchi, J. R. (2005). *Principles of applied reservoir simulation* (3rd ed.). Elsevier.

Kitware. (2010). *The VTK user's guide*. Kitware. ISBN: 9781930934238

Krekel, H., Oliveira, B., Pfannschmidt, R., Bruynooghe, F., Laugher, B., & Bruhin, F. (2004). *Pytest 6.2.4*. https://github.com/pytest-dev/pytest

Lie, K.-A. (2019). *An introduction to reservoir simulation using MATLAB/GNU octave: User guide for the MATLAB reservoir simulation toolbox (MRST)*. Cambridge University Press.

Pettersen, Ø. (2006). *Basics of reservoir simulation with the eclipse reservoir simulator*. http://www.mj-oystein.no/index_htm_files/ResSimNotes.pdf.

Ponting, D. K. (1989). Corner point geometry in reservoir simulation. *ECMOR i-1st European Conference on the Mathematics of Oil Recovery*, cp–234.

Schlumberger. (2014). *ECLIPSE reservoir simulation software reference manual*. Schlumberger.

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

Wang, B. (2018). *PyGRDECL a python-based GRDECL visualization library*. https://github.com/BinWang0213/PyGRDECL