Groundwater

Technology Spotlight/ Technology Editor / M. Gefell

Flopy: The Python Interface for MODFLOW

Andrew T. Leaf[1] and Michael N. Fienen[2]

Introduction

In 2016, a Python interface to MODFLOW called Flopy (Bakker et al. 2016) was published in *Groundwater*. This tool has proven popular, with thousands of downloads and nearly 200 literature references. Flopy is designed as a pre- and postprocessor for MODFLOW models that interacts with native MODFLOW file formats. The development of Flopy was motivated by principles of repeatability, extensibility, and flexibility in an open-source framework. When combined with geographic information systems (GIS), Flopy can serve a similar purpose as a graphical user interface (GUI) to enable visual inspection of model structure, properties, and results. As an open-source project, 50 contributors have committed to the development of the software, led by a core design team who also approve or deny proposed changes. This allows any user to customize Flopy for their needs and, if their additions are found to be generally useful and robust, they are incorporated into the main tool.

Flopy is developed in the Python scripting language, which makes it possible to document all model construction and data processing steps for a modeling project. Aspirationally, this enables reproducibility where another modeler could pickup, understand, repeat, and build from the scripts a modeler creates. Practically, this is a high bar and scripts developed to use Flopy for MODFLOW models may be challenging to fully repeat. However, having a scripted record of model-related processing increases the likelihood that important decisions and challenges that could result in errors (e.g., unit conversion or geolocation) are discovered more efficiently than if there was no record of the process.

[1]Corresponding author: U.S. Geological Survey Upper Midwest Water Science Center, 1 Gifford Pinchot Drive, Madison, WI, 53726; aleaf@usgs.gov

[2]U.S. Geological Survey Upper Midwest Water Science Center, 1 Gifford Pinchot Drive, Madison, WI, 53726

Received September 2022, accepted September 2022.

doi: 10.1111/gwat.13259

Developments from Initial Publication to Today

Flopy was originally developed to support the model structure of MODFLOW-2005 and MODFLOW-NWT. These versions of MODFLOW are still supported, but a major revision to the code resulted in support for MODFLOW6. With MODFLOW6 comes support for unstructured grids, multiple models within a single simulation, and integrated transport modeling.

More support for geospatial data (e.g., intersecting geolocated data with MODFLOW grids—structured and unstructured) has also been developed. This includes the ability to resample data to MODFLOW grids and to calculate intersections of various geometries (points, lines, and polygons) with MODFLOW cells. MODFLOW does not, inherently, contain a geospatial reference but Flopy accepts a grid definition to enable interaction of data with the model grid.

Improving documentation has been a major focus for Flopy development. Documentation now takes several forms. Comprehensive, programmatic documentation is provided at https://flopy.readthedocs.io/en/latest/ following the Numpy documentation style. This documentation covers major Flopy objects and methods and is particularly helpful as a reference. Supporting example-based learning has been a core goal of documenting Flopy with interactive Jupyter Notebooks (https://jupyter.org/). Several sources of example notebooks are available including the main documentation site (https://flopy.readthedocs.io/en/latest/tutorials.html) and in the main source-code repository (https://github.com/modflowpy/flopy/tree/develop/examples/Notebooks). The former set of notebooks is arranged in a logical progression but with limited scope, whereas the latter contains more examples that are not as clearly organized. Between the two sources are many examples that cover most of the functionality of Flopy.

Why Use Flopy?

Numerical models such as MODFLOW provide powerful tools for simulating real world systems that are heterogeneous and often have complex boundaries or structures and transience. However, the myriad of

operations required to construct a numerical model presents a fundamental challenge to efficient and robust science. Without automation, effective stepwise modeling, fixing inevitable errors, exploring model uncertainty and even understanding how a model was built can all be difficult or impossible, thereby limiting the usefulness of a model, and often consuming nights and weekends of a modeler's time. GUIs can overcome some of these challenges by providing easy visualization and setup of an initial model, and handling of often obscure input and output formats. However, most GUI workflows are not readily automatable, and therefore prone to the issues mentioned above. Furthermore, even the most capable GUIs cannot handle every idiosyncratic situation that inevitably arises in a groundwater modeling project-- usually some ad hoc operations are needed, which only adds to workflow complexity. A recent debate about the use of GUIs versus scripting explores this topic in more detail: https://gmdsi.org/blog/gmdsi-tech-talk-scripted-workflows-vs-guis/.

As a Python package, Flopy offers a model construction environment that is by definition automatable, and in principle repeatable. Moreover, by facilitating the transfer of model information to and from generic Python data structures, Flopy provides a gateway to the entire scientific Python ecosystem, allowing the relatively small groundwater modeling community to leverage extensive and robust software tools that are developed and used by tens to hundreds of millions of people worldwide. In addition to the visualization package matplotlib (https://matplotlib.org/), this includes geospatial packages (e.g., geopandas; https://geopandas.org/) that facilitate model visualization alongside supporting data in a GIS GUI (e.g., QGIS; https://qgis.org/). Perhaps most importantly, all model construction operations simple or complex can be included together in a unified, repeatable workflow. All of these tools, along with automation, can allow modelers to work more efficiently, and ultimately devote more energy to the motivating questions behind the model.

Challenges

The challenge, of course is that using Flopy requires learning Python. The deeper one's knowledge and experience with Python, the more effective they can be at harnessing the power of Flopy. The documentation for Flopy is extensive, but it can be challenging to find the terminology that leads to documentation of the precise action desired by a new user. As open-source software, there is always the option to search through the code, but this is not practical for all users. Working through examples helps build the necessary background but, as with any sophisticated tool, there is a learning curve and the need to invest time.

Python has posed challenges of portability in the past, but the advent of Conda environments (https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html) and Conda Pack (https://conda.github.io/conda-pack/) have made it generally

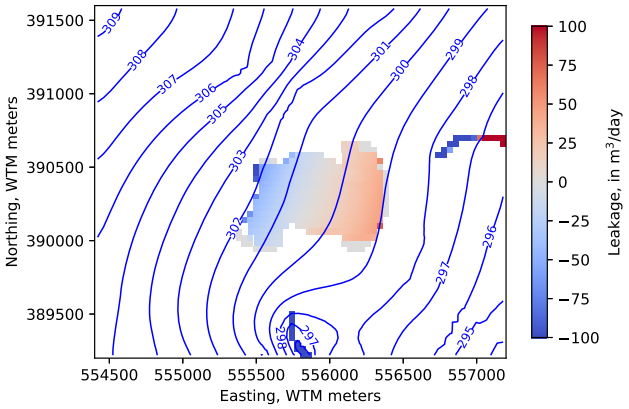

Figure 1. Simulated heads and groundwater/surface water interactions for the Pleasant Lake example problem, as visualized through the Flopy mapping utility and the Matplotlib Python package.

robust and practical to distribute and run Flopy on any modern computer running Windows, Linux, or the MacOS. One thing we would like to see in Flopy is more native integration with Pandas (https://pandas.pydata.org), which has become the de facto standard for working with tabular data.

A Worked Example

To illustrate how Flopy works, we have provided a worked example (https://github.com/doi-usgs/pleasant-lake-flopy-example; Leaf and Fienen 2022). The example is based on the Pleasant Lake (MODFLOW 6) model published by Fienen et al. (2021, 2022), and includes a level of complexity that is similar to a "real world" application, in that it includes Streamflow Routing (SFR) and Lake Package boundary conditions and specifies MODFLOW input through external text array files that are amenable to parameter estimation. Flopy is used to develop a geolocated model grid, map input to the grid, and create MODFLOW input files. The example is developed in a Jupyter Notebook, which allows for interactive computing and the display of plots (such as Figure 1) and documentation alongside code. The example notebook can be downloaded and run locally or run in the cloud through a web browser.

How to Get Started

Flopy can be installed using the popular package managers Pip (https://pypi.org/project/pip/) and Conda (https://docs.conda.io/en/latest/) or from source (https://github.com/modflowpy/flopy). Numerous online materials for learning Python exist, for example this website from the University of Waterloo: https://cscircles.cemc.uwaterloo.ca. Many scientists also learn by modifying existing code from colleagues or online examples such as the aforementioned Flopy tutorials, or by searching Stack Overflow (https://stackoverflow.com/), which provides an online forum where programmers of all levels can interact

and get answers to questions. Pair programming, where two people work together in real time on the same piece of code, is another potential approach, especially if one of the programmers has more experience. The free Visual Studio Code editor (VSCode; https://code.visualstudio.com/) has a LiveShare feature that allows multiple people to code together in real time over the internet, as well as many other tools to facilitate writing clean, error-free code. Organizations like Software Carpentry (https://software-carpentry.org/) teach in-person classes and the annual SciPy Conference (https://conference.scipy.org/) includes in-person tutorials at all levels that are later posted to YouTube. Finally, the U.S. Geological Survey hosts regular in-person classes focused on Flopy and Python for hydrologists. The Flopy GitHub site (https://github.com/modflowpy/flopy) also includes both a discussion forum and an Issues tab where users can submit bug reports or feature requests. Stack Overflow includes a Flopy tag, though the discussions forum on GitHub currently appears to be more active.

Disclaimer: Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

References

Bakker, M., V. Post, C.D. Langevin, J.D. Hughes, J.T. White, J.J. Starn, and M.N. Fienen. 2016. Scripting MODFLOW model development using python and Flopy. *Groundwater* 54, no. 5: 733–739. https://doi.org/10.1111/gwat.12413

Fienen, M.N., M.J. Haserodt, A.T. Leaf, and S.M. Westenbroek. 2022. Simulation of regional groundwater flow and groundwater/lake interactions in the Central Sands. Wisconsin: U.S. Geological Survey Scientific Investigations Report 2022-5046. http://doi.org/10.3133/sir20225046

Fienen, M.N., M.J. Haserodt, and A.T. Leaf. 2021. MODFLOW models used to simulate groundwater flow in the Wisconsin Central Sands Study Area, 2012-2018. U.S. Geological Survey Data Release. https://doi.org/10.5066/P9BVFSGJ

Leaf, A.T. and M.N. Fienen. 2022. Pleasant Lake worked Flopy example, version 0.1. U.S. Geological Survey Software Release, 1 Sept 2022. https://doi.org/10.5066/P9EFHF9H