The Journal of Open Source Software

BoxKit: A Python library to manage analysis of block-structured simulation datasets

Akash Dhruv [1]

1 Argonne National Laboratory, USA

DOI: 10.21105/joss.05649

Software

- Review
- Repository
- Archive

Editor: Kelly Rowland

Reviewers:

- @rvg296
- @Abinashbunty

Submitted: 20 June 2023
Published: 04 December 2023

Summary

BoxKit is a library that provides building blocks to parallelize and scale data science, statistical analysis, and machine learning applications for block-structured simulation datasets. Spatial data from simulations can be accessed and managed using tools available in this library to interface with packages like SciKit, PyTorch, and OpticalFlow for post-processing and analysis.

The library provides a Python interface to efficiently access Adaptive Mesh Refinement (AMR) data typical of simulation outputs, and leverages multiprocessing libraries like JobLib and Dask to scale analysis on Non-Uniform Memory Access (NUMA) and distributed computing architectures.

Statement of need

Simulation sofware instruments like Flash-X (Dubey et al., 2022) store output in the form of Hierarchical Data Format (HDF5) datasets. Each dataset is often gigabytes (GB) in size and requires cache efficient techniques to enable its integration with Python packages. BoxKit data structures act as a wrapper around simulation output stored in HDF5 files and provide metadata for AMR blocks that describe the simulation domain. The wrapper objects are lightweight in nature and represent chunks of data stored on disk, acting as array like input for Python functions/methods. This approach allows for selective loading of data from disk to memory in form of chunks/blocks which improves cache efficiency. The library also enables creation of new datasets for data-intensive workflows, and can be extended beyond its current application to numerical simulations.

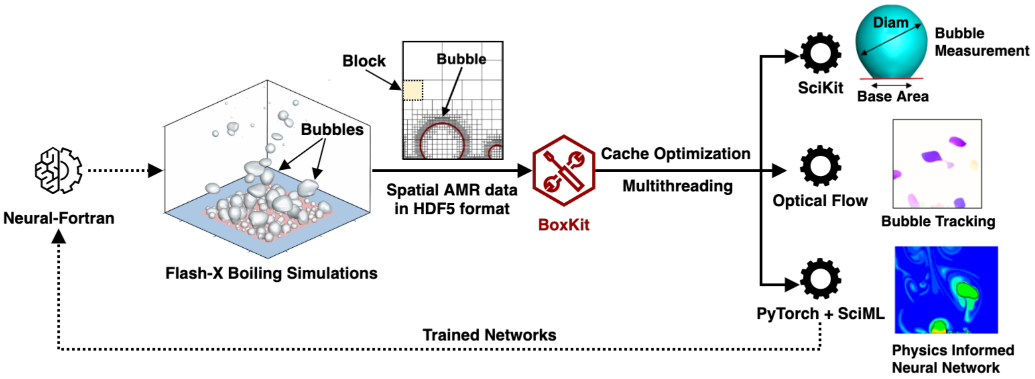

Figure 1: BoxKit is designed to integrate simulation software instruments like Flash-X with Python-based machine learning and data analysis packages. Large simulation datasets (~10 GB) can leverage BoxKit to improve performance of offline training/analysis. This mechanism is part of a broader workflow to integrate simulations with machine learning using a Fortran-Python bridge shown with dotted lines.

Compared to existing data analysis packages like yt (Turk et al., 2011), BoxKit offers more intuitive abstraction layers over AMR blocks through its metadata wrappers. This provides raw access to simulation data allowing users to develop their own low-level methods for spatio-temporal interpolation and stenciled computations. We aim for this library to complement existing packages rather than replace them.

BoxKit also offers wrappers to scale the process of deploying workflows on NUMA and distributed computing architectures by providing decorators that can parallelize Python operations over a single data structure to operate over a list. This can be understood better using the workflow described in Figure 1 that has been applied to data analysis and machine learning applications in chemical and thermal science engineering (Dhruv, 2023; Hassan et al., 2023). Output from Flash-X boiling simulations is created and stored on multinode clusters. Processing this output through BoxKit allows for scaling a simple operation over block to a list of blocks as shown below,

```python
# Decorate function on a block with desired configuration for parallelization
@Action(num_procs, parallel_backend)
def operation_on_block(block, *args):
    pass

# Call the function with list of blocks as the first argument
dset = boxkit.read_datset(...)
operation_on_block((block for block in dset.blocklist), *args)
```

The Action wrapper converts the function, operation_on_block, into a parallel method which can be deployed on a multinode cluster with the desired backend (JobLib/Dask). BoxKit does not interfere with parallelization schema of target applications like SciKit, OpticalFlow, and PyTorch which function independently using available resources.

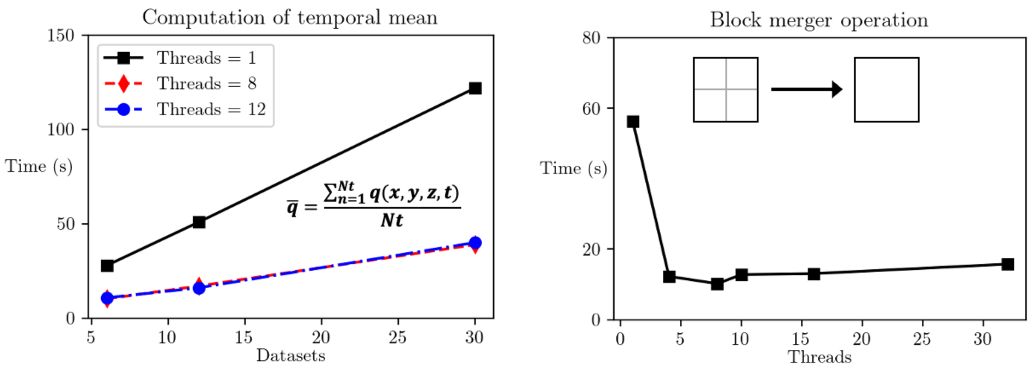

Figure 2: Preliminary performance analysis of BoxKit on a single 22 core IBM Power9 node (L1 cache - 32+32 kilobytes (KiB) per core, L2 cache - 512 KiB per core) for operations involving calculation of temporal mean across multiple datasets (left), and merging block-structured AMR datasets into contiguous arrays (right).

Figure 2 provides results of performance tests performed on a single 22 core node on Summit (*ORNL*, 2023) for two basic operations: (1) Calculation of temporal mean of heat flux in Flash-X boiling simulations $q(x, y, z, t)$, and (2) A block merger operations to convert AMR data into contiguous arrays.

Calculation of temporal mean requires operation on data across multiple datasets, with each dataset approximately 10 GB in size. Following is the mathematical representation of the problem where Nt represents the total number of datasets,

$$\overline{q} = \frac{\sum_{n=1}^{Nt} q(x,y,z,t)}{Nt} \tag{1}$$

Loading all the datasets into cache memory at the same time is very inefficient for this problem and requires use of BoxKit's metadata wrappers to efficiently load data chunks from disk, operate locally in space, and scale its computation across multiple threads. Based on the graph in Figure 2 the parallel performance scales better as Nt increases.

Mapping of AMR data to contingous arrays becomes important for applications where global operations in space are required. An example of this is SciKit's `skimage.measure` method, which can be used to measure bubble shape and size for Flash-X boiling simulations. BoxKit improves performance of this operation by ~5x. Data for these performance studies along with corresponding IPython notebooks can be found in (*BoxKit Performance*, n.d.).

Ongoing work

Our ongoing work focuses on developing BoxKit to improve performance of Scientific Machine Learning (SciML) applications and using it as part of a broader workflow that integrates Fortran/C++ based applications with state-of-art machine learning packages available in Python as highlighted in Figure 1.

Acknowledgements

This material is based upon work supported by Laboratory Directed Research and Development (LDRD) funding from Argonne National Laboratory, provided by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-06CH11357.

References

BoxKit Performance. (n.d.). https://github.com/akashdhruv/boxkit-performance.

Dhruv, A. (2023). *A vortex damping outflow forcing for multiphase flows with sharp interfacial jumps*. https://arxiv.org/abs/2306.10174

Dubey, A., Weide, K., O'Neal, J., Dhruv, A., Couch, S., Harris, J. A., Klosterman, T., Jain, R., Rudi, J., Messer, B., Pajkos, M., Carlson, J., Chu, R., Wahib, M., Chawdhary, S., Ricker, P. M., Lee, D., Antypas, K., Riley, K. M., … Papatheodore, T. (2022). Flash-X: A multiphysics simulation software instrument. *SoftwareX*, *19*, 101168. https://doi.org/10.1016/j.softx.2022.101168

Hassan, S. M. S., Feeney, A., Dhruv, A., Kim, J., Suh, Y., Ryu, J., Won, Y., & Chandramowlishwaran, A. (2023). *BubbleML: A Multi-Physics Dataset and Benchmarks for Machine Learning* (Version 1.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.8039787

ORNL. (2023). https://www.olcf.ornl.gov/summit/.

Turk, M. J., Smith, B. D., Oishi, J. S., Skory, S., Skillman, S. W., Abel, T., & Norman, M. L. (2011). yt: A Multi-code Analysis Toolkit for Astrophysical Simulation Data. *The Astrophysical Journal Supplement Series*, *192*, 9. https://doi.org/10.1088/0067-0049/192/1/9