The Journal of Open Source Software

PyLops-MPI - MPI Powered PyLops with mpi4py

Rohan Babbar [1], **Matteo Ravasi** [2], **and Yuxi Hong** [3]

1 Cluster Innovation Center, University of Delhi, Delhi, India. **2** Earth Science and Engineering, Physical Sciences and Engineering (PSE), King Abdullah University of Science and Technology (KAUST), Thuwal, Kingdom of Saudi Arabia. **3** Lawrence Berkeley National Laboratory, Berkeley, California, United States of America.

DOI: 10.21105/joss.07512

Software

- Review
- Repository
- Archive

Editor: George K. Thiruvathukal

Reviewers:

- @gonsie
- @mikaem

Submitted: 09 October 2024
Published: 07 January 2025

Summary

Linear algebra and inverse problem theory are fundamental to many algorithms in signal processing, image processing, geophysics, and remote sensing. This paper introduces PyLops-MPI, an extension of the PyLops framework designed to enable distributed computing in the solution of large-scale inverse problems in Python. By leveraging the Message Passing Interface (MPI) standard, this library effectively harnesses the computational power of multiple nodes, allowing users to scale their inverse problems efficiently with minimal changes compared to their single-node PyLops code.

Statement of need

As scientific datasets grow and the demand for higher resolution increases, the use of distributed computing in matrix-free linear algebra becomes crucial. Models and datasets can in fact easily exceed the memory of a single machine—making it difficult to perform computations efficiently and accurately at the same time. Nevertheless, many linear operators in scientific inverse problems can be decomposed into a series of computational blocks that are well-suited for parallelization.

When addressing distributed inverse problems, we identify three distinct families of problems:

1. **Fully distributed models and data**: Both model and data are split across nodes, with each node processing its own portion of the model and data. This leads to minimal communication, mainly when performing dot products in the solver or in the regularization terms.
2. **Distributed data, model available on all nodes**: Data is distributed across nodes, whilst the model is available on all nodes. Communication happens during the adjoint pass to sum models and in the solver for data vector operations.
3. **Model and data available on all nodes**: All nodes have identical copies of the data and model. Communication only happens within the operator, with no communication in solver needed.

MPI for Python (mpi4py (Dalcin & Fang, 2021)) provides Python bindings for the MPI standard, allowing applications to leverage multiple processors. Projects like mpi4py-fft (Mortensen et al., 2019), mcdc (Morgan et al., 2024), and mpi4jax (Häfner & Vicentini, 2021) utilize mpi4py to provide distributed computing capabilities. Similarly, PyLops-MPI, which is built on top of PyLops (Ravasi & Vasconcelos, 2020), leverages mpi4py to solve large-scale problems in a distributed fashion. Its intuitive API provide functionalities to scatter and broadcast data and model vector across nodes and allows various mathematical operations (e.g., summation, subtraction, norms) to be performed. Additionally, a suite of MPI-powered linear operators and solvers is offered, and its flexible design eases the integration of custom operators and solvers.

Software Framework

PyLops-MPI is designed to tackle large-scale linear inverse problems that are difficult to solve using a single process (due to either extremely high computational cost or memory requirements).

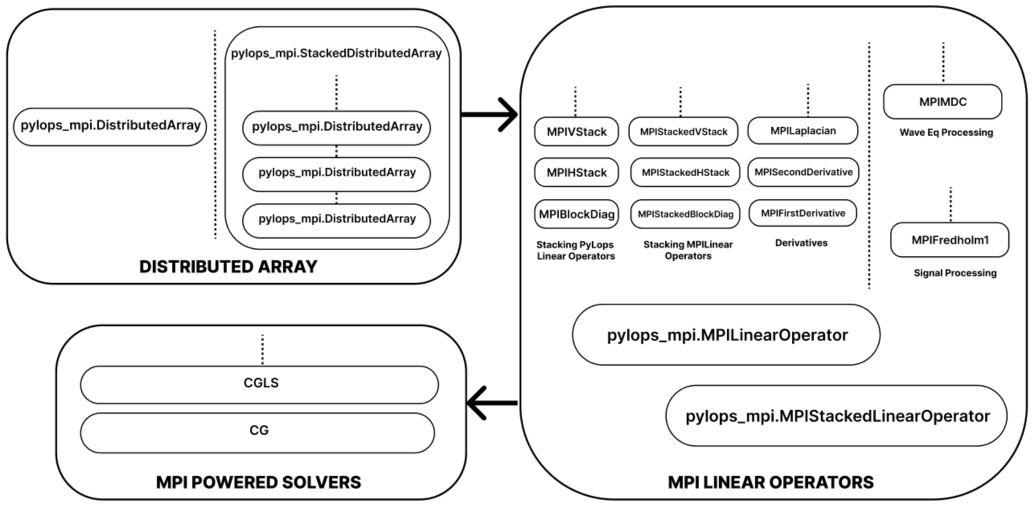
pylops_mpi.StackedDistributedArray
pylops_mpi.DistributedArray
pylops_mpi.DistributedArray
pylops_mpi.DistributedArray
pylops_mpi.DistributedArray
DISTRIBUTED ARRAY
MPIVStack
MPIStackedVStack
MPILaplacian
MPIMDC
Wave Eq Processing
MPIHStack
MPIStackedHStack
MPISecondDerivative
MPIBlockDiag
MPIStackedBlockDiag
MPIFirstDerivative
MPIFredholm1
Stacking PyLops Linear Operators
Stacking MPILinear Operators
Derivatives
Signal Processing
pylops_mpi.MPILinearOperator
pylops_mpi.MPIStackedLinearOperator
MPI LINEAR OPERATORS
CGLS
CG
MPI POWERED SOLVERS

Figure 1: Software framework representation of the `PyLops-MPI` API.

Fig. 1 illustrates the main components of the library, emphasizing the relationship between the DistributedArray class, stacked operators, and MPI-powered solvers.

DistributedArray

The `pylops_mpi.DistributedArray` class serves as the fundamental array class, enabling both partitioning and broadcasting of large NumPy (Harris et al., 2020) or CuPy (Okuta et al., 2017) arrays across multiple processes. It also supports basic math operations such as addition (+), multiplication (*), dot-product (@). Finally, multiple DistributedArray objects can be stacked using `pylops_mpi.StackedDistributedArray` for further operations.

HStack, VStack, BlockDiag Operators

PyLops facilitates the combinations of multiple linear operators via horizontal, vertical, or diagonal stacking. PyLops-MPI provides distributed versions of these operations. Examples include `pylops_mpi.MPIBlockDiag`, which applies different operators in parallel on separate portions of the model and data, `pylops_mpi.MPIVStack`, which applies multiple operators in parallel to the whole model, with its adjoint applies the adjoint of each individual operator to portions of the data vector and sums the individual output, and `pylops_mpi.MPIHStack`, which is the adjoint of MPIVStack.

Halo Exchange

PyLops-MPI uses halo exchange to transfer portions of the model and data between ranks. Users should ensure consistent local data shapes to avoid extra communication during matrix-vector products. If shapes differ, the operator exchanges boundary data ("ghost cells") between neighboring processes, aligning shapes to enable efficient local computations and minimize overhead.

MPI-powered Solvers

PyLops-MPI offers a small subset of PyLops linear solvers, which can deal with `pylops_mpi.DistributedArray` and `pylops_mpi.StackedDistributedArray` objects. These solvers utilize the mathematical operations implemented in these classes and call the operator's forward and adjoint passes.

Use Cases

Here we present three use cases in geophysical inverse problems that correspond to the previously mentioned families of problems:

- *Seismic Post-Stack Inversion* can be used to characterize the subsurface (Ravasi & Birnie, 2021) from seismic data. In 3D applications, when both the model and data are three-dimensional arrays, PyLops-MPI distributes one spatial axis across different ranks. Each rank therefore processes a subset of the entire data and model. Communication occurs due to the introduction of regularization terms that promote smooth or blocky solutions.
- *Least-Squares Migration (LSM)* explains seismic data via a Born modelling engine to produce high-resolution images of the subsurface reflectivity (Nemeth et al., 1999). PyLops-MPI distributes the available sources across different MPI ranks, and each rank applies the Born modeling operator for a subset of sources with the broadcasted reflectivity.
- *Multi-Dimensional Deconvolution (MDD)* is a powerful technique used to estimate seismic datasets without overburden effects (Ravasi et al., 2022). PyLops-MPI tackles this large-scale inverse problem by distributing the kernel of the Multi-Dimensional Deconvolution operator across ranks, allowing each process to apply a batched matrix-vector multiplication to a portion of the input model and data.

As similar patterns are likely to emerge in inverse problems across other disciplines, we expect a broad adoption of the PyLops-MPI framework in other scientific fields.

Acknowledgements

The PyLops team acknowledges the support from Google Summer of Code and the NumFOCUS organization, which have been vital to the development of PyLops-MPI.

References

Dalcin, L., & Fang, Y.-L. L. (2021). mpi4py: Status update after 12 years of development. *Computing in Science & Engineering*, *23*(4), 47–54. https://doi.org/10.1109/MCSE.2021.3083216

Häfner, D., & Vicentini, F. (2021). mpi4jax: Zero-copy MPI communication of JAX arrays. *Journal of Open Source Software*, *6*(65), 3419. https://doi.org/10.21105/joss.03419

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Morgan, J. P., Variansyah, I., Pasmann, S. L., Clements, K. B., Cuneo, B., Mote, A., Goodman, C., Shaw, C., Northrop, J., Pankaj, R., Lame, E., Whewell, B., McClarren, R. G., Palmer, T. S., Chen, L., Anistratov, D. Y., Kelley, C. T., Palmer, C. J., & Niemeyer, K. E. (2024).

Monte Carlo / dynamic code (MC/DC): An accelerated Python package for fully transient neutron transport and rapid methods development. *Journal of Open Source Software*, *9*(96), 6415. https://doi.org/10.21105/joss.06415

Mortensen, M., Dalcin, L., & Keyes, D. E. (2019). mpi4py-fft: Parallel fast Fourier transforms with MPI for Python. *Journal of Open Source Software*, *4*(36), 1340. https://doi.org/10.21105/joss.01340

Nemeth, T., Wu, C., & Schuster, G. T. (1999). Least-squares migration incomplete reflection data. *Geophysics*, *64*(1), 208–221. https://doi.org/10.1190/1.1444517

Okuta, R., Unno, Y., Nishino, D., Hido, S., & Loomis, C. (2017). CuPy: A NumPy-compatible library for NVIDIA GPU calculations. *Proceedings of Workshop on Machine Learning Systems (LearningSys) in the Thirty-First Annual Conference on Neural Information Processing Systems (NIPS)*. http://learningsys.org/nips17/assets/papers/paper_16.pdf

Ravasi, M., & Birnie, C. (2021). A joint inversion-segmentation approach to assisted seismic interpretation. *Geophysical Journal International*, *228*(2), 893–912. https://doi.org/10.1093/gji/ggab388

Ravasi, M., Selvan, T., & Luiken, N. (2022). Stochastic multi-dimensional deconvolution. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1–14. https://doi.org/10.1109/tgrs.2022.3179626

Ravasi, M., & Vasconcelos, I. (2020). PyLops - A linear-operator Python library for scalable algebra and optimization. *SoftwareX*, *11*. https://doi.org/10.1016/j.softx.2019.100361