The Journal of Open Source Software

AutoPDEx: An Automized Partial Differential Equation solver based on JAX

Tobias Bode [1]

1 Institute of Continuum Mechanics, Leibniz University Hannover, An der Universität 1, 30823 Garbsen, Germany

DOI: 10.21105/joss.07300

Software

- Review
- Repository
- Archive

Editor: Daniel S. Katz

Reviewers:

- @celliern
- @janvorisek

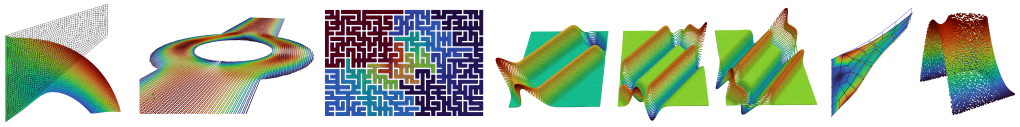
Submitted: 17 September 2024
Published: 11 April 2025

Summary

`AutoPDEx` is free and open-source software for solving partial differential equations (PDEs) based on the automatic code transformation capabilities of `JAX` (Bradbury et al., 2018). It is designed to provide a modular, flexible, and extendable environment for solving boundary and initial value problems, allowing seamless integration with machine learning algorithms and GPU acceleration through the Accelerated Linear Algebra (XLA) compiler.

At its core, AutoPDEx includes a versatile solver module that supports algorithms such as adaptive load stepping, Newton's method, and nonlinear minimizers. The PDEs to be solved and the chosen variational methods and solution spaces can be specified via user-defined JAX-transformable functions. Pre-built models and ansatz functions are available in the `models` and `spaces` modules. In addition to finite element methods, mesh-free approaches and neural networks can be used as solution spaces. This flexibility makes AutoPDEx suitable for researchers working at the intersection of numerical analysis and machine learning.

The `implicit_diff` module provides a wrapper to make the solution methods differentiable through automatic implicit differentiation (Blondel et al., 2022). This allows adaptive load stepping to be used in combination with arbitrary order sensitivity analyses in forward and reverse mode. For solving linear systems, it integrates with high-performance external solvers as, e.g., PARDISO (Schenk & Gärtner, 2004) and PETSc (Balay et al., 2019). Below, the solution of some example test cases available in the documentation is depicted.



Statement of Need

The efficient and accurate solution of partial differential equations (PDEs) is a central task in many scientific and engineering applications. Python has become a popular platform for PDE solving due to its versatility and the availability of mature libraries such as FEniCS (Alnæs et al., 2015) with dolfin-adjoint (Mitusch et al., 2019) for sensitivity analysis. However, many existing tools rely on traditional numerical approaches and require manual implementation of residuals and tangents, or use domain-specific symbolic manipulations.

AutoPDEx uses the code-to-code transformation library JAX to compute local derivatives such as spatial gradients, as well as residual and tangent contributions, through automatic differentiation. By leveraging automatic implicit differentiation, the entire solution procedure can also be differentiated in a unified way. This and the modular structure of AutoPDEx make it attractive for research applications in which having sensitivities is beneficial, e.g.,

as in material parameter identification, topology optimization, uncertainty estimation, and multi-scale analysis (Korelc & Wriggers, 2016). By building on JAX, AutoPDEx further enables the smooth combination of numerical simulations with machine learning models from the JAX ecosystem, for instance, those provided by Flax (Heek et al., 2024) and Equinox (Kidger & Garcia, 2021). This capability allows incorporating data-driven approaches into traditional simulation workflows.

The analysis in AutoPDEx can be used in combination with established tools like Gmsh (Geuzaine & Remacle, 2009) for mesh generation and PyVista (Sullivan & Kaszynski, 2019) or ParaView (Ahrens et al., 2005) for visualization, providing a comprehensive solution from model preparation to analysis.

Acknowledgements

This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through the project grant TRR 298 (SIIRI, Project No. 426335750), as well as under Germany's Excellence Strategy within the Cluster of Excellence PhoenixD (EXC 2122, Project ID 390833453).

References

Ahrens, J., Geveci, B., & Law, C. (2005). ParaView: An end-user tool for large-data visualization. In *Visualization handbook* (pp. 717–731). Butterworth-Heinemann. https://doi.org/10.1016/B978-012387582-2/50038-1

Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., & Wells, G. N. (2015). The FEniCS project version 1.5. *Archive of Numerical Software*, *3*(100). https://doi.org/10.11588/ans.2015.100.20553

Balay, S., Abhyankar, S., Adams, M., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W., & others. (2019). *PETSc users manual*. https://doi.org/10.2172/1577437

Blondel, M., Berthet, Q., Cuturi, M., Frostig, R., Hoyer, S., Llinares-López, F., Pedregosa, F., & Vert, J.-P. (2022). Efficient and modular implicit differentiation. *Advances in Neural Information Processing Systems*, *35*, 5230–5242. https://doi.org/10.48550/arXiv.2105.15183

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/google/jax

Geuzaine, C., & Remacle, J.-F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, *79*(11), 1309–1331. https://doi.org/10.1002/nme.2579

Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., & Zee, M. van. (2024). *Flax: A neural network library and ecosystem for JAX* (Version 0.9.0). http://github.com/google/flax

Kidger, P., & Garcia, C. (2021). Equinox: Neural networks in JAX via callable PyTrees and filtered transformations. *arXiv Preprint arXiv:2111.00254*. https://doi.org/10.48550/arXiv.2111.00254

Korelc, J., & Wriggers, P. (2016). *Automation of finite element methods*. Springer. https://doi.org/10.1007/978-3-319-39005-5

Mitusch, S., Funke, S., & Dokken, J. (2019). Dolfin-adjoint 2018.1: Automated adjoints for FEniCS and Firedrake. *Journal of Open Source Software*, *4*(38), 1292. https://doi.org/10.21105/joss.01292

Schenk, O., & Gärtner, K. (2004). Solving unsymmetric sparse systems of linear equations with PARDISO. *Future Generation Computer Systems*, *20*(3), 475–487. https://doi.org/10.1016/j.future.2003.07.011

Sullivan, C., & Kaszynski, 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