The Journal of Open Source Software

PyHOPE: A Python Toolkit for Three-Dimensional Unstructured High-Order Meshes

Patrick Kopper [1*¶], **Marcel P. Blind** [1*], **Anna Schwarz** [1], **Marius Kurz** [2], **Felix Rodach** [1], **Stephen M. Copplestone** [3], **and Andrea D. Beck** [1]

1 Institute of Aerodynamics and Gas Dynamics, University of Stuttgart, Stuttgart, Germany **2** Centrum Wiskunde & Informatica, Amsterdam, Netherlands **3** boltzplatz - numerical plasma dynamics GmbH, Stuttgart, Germany ¶ Corresponding author * These authors contributed equally.

DOI: 10.21105/joss.08769

Software

- Review
- Repository
- Archive

Editor: Patrick Diehl

Reviewers:

- @mohd-afeef-badri
- @jwallwork23

Submitted: 25 June 2025
Published: 03 November 2025

Summary

PyHOPE (Python High-Order Preprocessing Environment) is a framework for generating and converting high-order meshes composed of standard 3D element types, designed for massively parallel spectral element solvers on high-performance computing (HPC) systems. PyHOPE builds on and extends Gmsh (Geuzaine & Remacle, 2009), which is used for the initial mesh generation and/or mesh read-in before conversion of the mesh to its internal representation and application of boundary conditions. Parallel read-in is crucial on HPC clusters, which typically use parallel distributed file systems to enable and scale storage access by striping data across multiple servers. The primary output format of PyHOPE is the HOPR (Hindenlang, 2014) HDF5 curved mesh format, which is specifically designed for parallel read-in of unstructured three-dimensional meshes of arbitrary order, including tetrahedra, pyramids, prisms, and hexahedra. Information stored in HOPR format facilitates non-overlapping input/output (I/O) through collocation of the required mesh information, including the vertex and side information together with element connectivity, in per-element packages. Each package is assigned a unique identifier via ordering along structured dimensions or a space-filling curve.

Statement of Need

The Discontinuous Galerkin Spectral Element Method (DGSEM) is a powerful numerical approach for solving partial differential equations, particularly in high-performance computing applications. Prominent examples of DGSEM codes originating and being actively developed at the University of Stuttgart include the FLEXI family[1] geared towards solving the compressible Navier-Stokes equations, along with PICLas[2], which focuses on plasma simulation with a Particle-in-Cell/Direct Simulation Monte Carlo approach. A crucial aspect of DGSEM is the mapping of the equations to be solved from reference space to physical space, which relies on the computation of the Jacobian determinant to ensure accurate transformations for curved high-order elements. At the same time, DGSEM features a highly local stencil since grid elements are connected solely via the numerical flux through adjacent element faces. Providing mesh and solution data in an on-disc data format, which facilitates non-overlapping I/O, is key to efficient parallel data access, thereby minimizing execution time. While HOPR has traditionally served as the reference implementation of a mesh generator for this process, PyHOPE is a modern alternative that enhances readability and extensibility. Designed with a clear code structure and modularization in mind, PyHOPE offers a more user-friendly and

[1] https://numericsresearchgroup.org/codes.html
[2] https://github.com/piclas-framework/piclas

adaptable solution for researchers and engineers working on high-order mesh generation and transformation.

State of the Field

PyHOPE shares similarities with other high-order mesh generation tools such as HOPR, which serves as the reference implementation. Although the HOPR mesh format is supported by a variety of solvers, there is limited code support to generate meshes in this format. Compared to HOPR, PyHOPE places a special focus on enhanced adaptability, including powerful element splitting functionality, improved usability, and broader support for modern mesh formats, including curved and mixed meshes to enable more complex mesh generation. The following spectral element solvers have (optional) support for meshes generated in PyHOPE.

Framework	Language	Equation System	Reference
FLEXI	Fortran	NSE	Krais et al. (2021)
∃LEXI	Fortran	NSE/MRG	Kopper et al. (2023)
GALÆXI	Fortran/C	NSE	Kurz et al. (2025)
FLUXO	Fortran	NSE/MHD/Maxwell	Rueda-Ramirez et al. (2017)
HORSES3D	Fortran	NSE/Cahn-Hilliard	Ferrer et al. (2023)
PICLas	Fortran	Maxwell/Poisson	Fasoulas et al. (2019)
SELF	Fortran	Shallow Water/Euler	github.com/FluidNumerics/SELF

Equation Systems: NSE - Navier-Stokes, MRG - Maxey-Riley-Gatignol, MHD - Magnetohydrodynamics

Features

PyHOPE provides a simple and intuitive interface for generating and transforming high-order meshes. PyHOPE is distributed as open-source software on GitHub[3] and as a package on the Python Package Index (PyPI)[4]. It reads user input from a single plain-text configuration file in INI format[5]. The software supports the generation of block-structured meshes using any combination of canonical volumetric element types (tetrahedral, pyramidal, prismatic, and hexahedral), allowing for mesh stretching and post-deformation at arbitrary polynomial orders. PyHOPE can automatically create rectilinear boundary layer meshes based on a desired wall resolution or stretching factor. It can read in and merge both internally and externally created curved meshes while incorporating boundary conditions and automatically adapting sub-meshes to the desired polynomial order. PyHOPE also offers conversion of simplex elements into fully hexagonal cells through geometric subdivision, as well as mesh sorting along structured dimensions or space-filling Hilbert curves. Additionally, it calculates mesh connectivity information, supporting both periodic and non-conforming interfaces with hanging nodes, which can accommodate potential anisotropy. To ensure robustness, PyHOPE performs comprehensive consistency checks, verifying mesh watertightness, correct surface orientation, and valid Jacobian mappings. PyHOPE detects available simultaneous multithreading (SMT) capabilities and automatically enables process-based parallelism using the Python multiprocessing module.

Examples

PyHOPE is used to generate unstructured high-order HDF5 meshes for a variety of engineering applications, ranging from canonical cases such as channel flows or the Taylor-Green vortex to complex geometries such as airfoils and complete airplanes. High-order meshes in HOPR format

[3] https://github.com/hopr-framework/PyHOPE
[4] https://pypi.org/project/PyHOPE
[5] https://hopr-framework.github.io/PyHOPE

also find application in electromagnetics and plasma simulation, such as optical lenses and gyrotrons. PyHOPE comes with several tutorials which are included in the GitHub repository, together with the external mesh files where appropriate. These tutorials cover both the creation of block-structured grids using PyHOPE's built-in functionality as well as the read-in of externally created meshes. The available post-deformation options and topology conversion features are outlined as well. All tutorial cases are also used for code coverage and regression checking during Continuous Integration/Continuous Deployment (CI/CD).

One notable example from the field of aerospace engineering is the application of PyHOPE to the Common Research Model (CRM), a widely used aerodynamic benchmark. For this example, the CAD file is initially meshed using second-order simplex elements in the external generator ANSA v24 (BETA CAE Systems S.A., 2024) and the three-dimensional grid is exported in high-order curved CGNS 4.2.0 format based on HDF5. While the specific mesh used here does not include boundary layers, ANSA supports layered and mixed-element meshing, which has been successfully tested in other configurations. In all cases, a fully curved volume mesh is generated using ANSA's internal second-order volume mesher. PyHOPE reads the exported CGNS file, reconstructs boundary conditions, and creates a fully curved HOPR mesh including connectivity information for vertices, edges, and sides. The final mesh contains 147763 second-order elements and can be processed by PyHOPE on any standard machine using approximately 520 MB RAM (peak RSS). Notably, this workflow enables full utilization of CFD meshes generated by commercial tools such as ANSA and Pointwise (Pointwise, Inc., 2025), which are widely used in industry. Figure 1 shows the second-order unstructured surface grid and the instantaneous distribution of surface pressure on the NASA CRM, respectively. The latter was calculated using FLEXI, an open-source framework for the solution of the compressible Navier-Stokes equations using DGSEM.

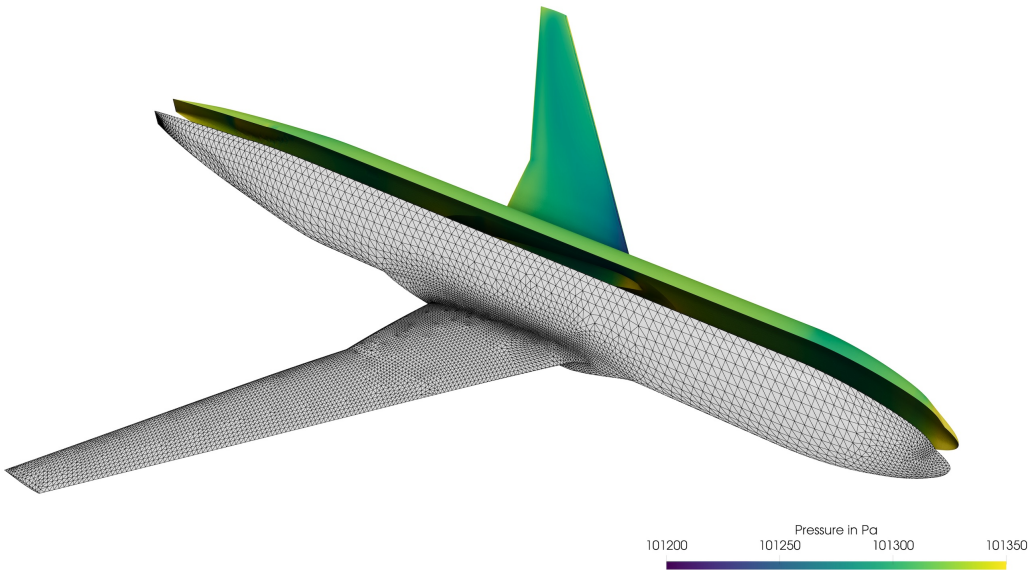

Figure 1: NASA Common Research Model (CRM). Left: Second-order unstructured surface grid. Right: Instantaneous distribution of surface pressure.

Acknowledgements

This work was funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark under grant agreement No 101093393. The research presented in this paper was funded in parts by the state of Baden-Württemberg under the project Aerospace 2050 MWK32-7531-49/13/7 "FLUTTER".

References

BETA CAE Systems S.A. (2024). *ANSA pre-processor*. BETA CAE Systems S.A. https://www.beta-cae.com/ansa.htm

Fasoulas, S., Munz, C.-D., Pfeiffer, M., Beyer, J., Binder, T., Copplestone, S., Mirza, A., Nizenkov, P., Ortwein, P., & Reschke, W. (2019). Combining particle-in-cell and direct simulation Monte Carlo for the simulation of reactive plasma flows. *Physics of Fluids*, *31*(7), 072006. https://doi.org/10.1063/1.5097638

Ferrer, E., Rubio, G., Ntoukas, G., Laskowski, W., Mariño, O. A., Colombo, S., Mateo-Gabín, A., Marbona, H., Lara, F. M. de, Huergo, D., Manzanero, J., Rueda-Ramírez, A. M., Kopriva, D. A., & Valero, E. (2023). HORSES3D: A high-order discontinuous Galerkin solver for flow simulations and multi-physics applications. *Computer Physics Communications*, *287*, 108700. https://doi.org/10.1016/j.cpc.2023.108700

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

Hindenlang, F. (2014). *Mesh curving techniques for high order parallel simulations on unstructured meshes* [PhD thesis, University of Stuttgart]. https://doi.org/10.18419/opus-3957

Kopper, P., Schwarz, A., Copplestone, S. M., Ortwein, P., Staudacher, S., & Beck, A. (2023). A framework for high-fidelity particle tracking on massively parallel systems. *Computer Physics Communications*, *289*, 108762. https://doi.org/10.1016/j.cpc.2023.108762

Krais, N., Beck, A., Bolemann, T., Frank, H., Flad, D., Gassner, G., Hindenlang, F., Hoffmann, M., Kuhn, T., Sonntag, M., & Munz, C.-D. (2021). FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws. *Computers & Mathematics with Applications*, *81*, 186–219. https://doi.org/10.1016/j.camwa.2020.05.004

Kurz, M., Kempf, D., Blind, M. P., Kopper, P., Offenhäuser, P., Schwarz, A., Starr, S., Keim, J., & Beck, A. (2025). GALÆXI: Solving complex compressible flows with high-order discontinuous Galerkin methods on accelerator-based systems. *Computer Physics Communications*, *306*, 109388. https://doi.org/10.1016/j.cpc.2024.109388

Pointwise, Inc. (2025). *Pointwise mesh generation software*. https://www.pointwise.com

Rueda-Ramirez, A., Schlottke-Lakemper, M., Gassner, G. J., Astanin, A., & Winters, A. R. (2017). *DGSEM for general advection-diffusion equations*. https://github.com/project-fluxo/fluxo