The Journal of Open Source Software

SOFIA: A Python Library for High-Quality 2D Triangular Mesh Adaptation

Youssef Mesri [1]

1 MINES Paris - PSL, France

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Hugo Ledoux

Reviewers:

- @inducer
- @joewallwork

Submitted: 18 November 2025
Published: unpublished

Summary

Triangular meshes are fundamental data structures in computational science and engineering, serving as the backbone for numerical simulations in fluid dynamics, structural analysis, heat transfer, and many other fields. The quality of a mesh directly impacts the accuracy, stability, and convergence of numerical solutions (Manzinali et al., 2018; Mesri, Zerguine, et al., 2008; Persson & Strang, 2004). `SOFIA` (Scalable Operators for Field-driven Iso/Ani Adaptation) is a Python library that provides robust and efficient tools for 2D triangular mesh modification, refinement, and quality improvement through local topological operations, with particular emphasis on **anisotropic mesh adaptation** and **automatic boundary preservation**.

In its current version, `SOFIA` focuses on **mesh topology modification (h-adaptation)**. That is, it changes the mesh by locally splitting/collapsing/flipping edges and inserting/removing vertices to refine or coarsen the discretisation. Limited vertex relocation is available through smoothing to improve element quality, but `SOFIA` does not currently implement full **mesh movement (r-adaptation)** driven by a PDE-based moving mesh method; such r-adaptation workflows are left to external solvers or future work.

Statement of Need

While several mesh generation tools exist (Geuzaine & Remacle, 2009; Kloeckner, 2015; Shewchuk, 1996; Zhou, 2018), there is a gap in the Python ecosystem for a lightweight, well-documented library focused specifically on **mesh adaptation** and **local modification operations**. Existing solutions like Triangle (wrapped by MeshPy) offer refinement capabilities but are primarily oriented toward initial mesh generation with Delaunay triangulation. `SOFIA` instead emphasizes **metric-based anisotropic adaptation** during simulation workflows, providing fine-grained control over mesh topology modification.

`SOFIA` addresses these needs by providing a **pure Python** implementation of:

1. **Local topological operations** (split, collapse, flip, insert, remove) enabling fine-grained mesh modification
2. **Metric-based anisotropic h-adaptation** driven by a user-supplied tensor field (see below)
3. **Automatic boundary preservation** built into edge collapse, so boundary conformity can be maintained without manual vertex "locking"
4. **Quality management** with metrics and optimisation strategies for isotropic and anisotropic meshes
5. **Reliability** supported by an extensive unit test suite

The repository README mentions a future C++ backend; however, the current release is a pure Python package and this paper describes functionality available in the present version.

The library is designed for computational scientists, researchers, and engineers who need to adaptively refine/coarsen meshes during simulations, optimize existing meshes, implement custom mesh adaptation strategies, or generate boundary layer meshes for high-Reynolds number flows.

Functionality

Core Operations

`SOFIA` implements the fundamental local operations for triangular mesh modification:

- **Edge operations**: Split edges at midpoint or custom locations, collapse edges, and flip edges
- **Vertex operations**: Insert and remove vertices using cavity re-triangulation
- **Pocket filling**: Fill holes in meshes after vertex removal or other operations
- **Boundary operations**: Safely manipulate boundary edges and vertices while maintaining domain conformity
- **Automatic boundary detection**: Edge collapse operations automatically detect boundary vertices and preserve domain geometry.

Anisotropic Mesh Adaptation

A key feature of `SOFIA` is its support for anisotropic mesh adaptation, crucial for capturing directional features (Alauzet, 2010; Loseille & Alauzet, 2011; Mesri, Alauzet, et al., 2008; Mesri, Zerguine, et al., 2008; Mesri et al., 2016):

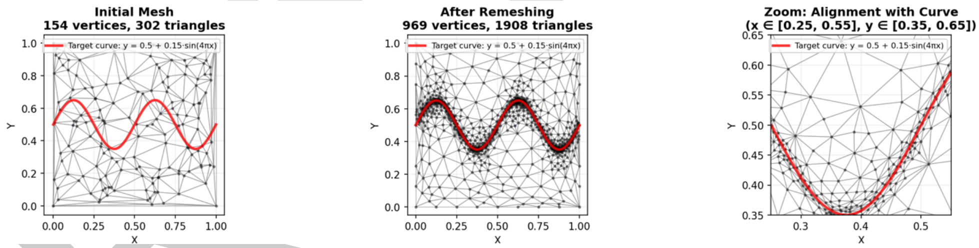

Figure 1: Example of an anisotropic adapted triangular mesh produced with `SOFIA`. The adaptation is driven by a user-supplied metric tensor field, resulting in strongly stretched elements aligned with the target features while preserving the domain boundary.

- **Metric tensor field**: User-defined symmetric positive-definite tensor field specifying desired mesh resolution and anisotropy
- **Metric-based edge lengths**: Operations use metric edge length $L_M(e) = \sqrt{(p_2 - p_1)^T M (p_2 - p_1)}$ instead of Euclidean distance
- **Boundary layer support**: Natural support for highly anisotropic elements near boundaries (high aspect ratios)
- **Smooth transitions**: Metric fields can specify smooth transitions from anisotropic to isotropic regions

Here, p_1 and p_2 are the endpoints of an edge e in physical coordinates, M is the symmetric positive-definite metric tensor (or a suitable edge-averaged metric), and $L_M(e)$ is the target length measure used to decide whether an edge should be split or collapsed. This is a **metric-based h-adaptation** approach (in the spirit of local mesh modification governed by a metric) rather than a hierarchical refinement strategy.

Key Innovation: Boundary-aware Edge Collapse

Traditional edge collapse operations often collapse to the midpoint of an edge, which can deform domain boundaries. In `SOFIA`, boundary preservation is built into the collapse decision:

boundary vertices are detected from topology (boundary edges have only one incident triangle), and when collapsing an edge touching the boundary the operation collapses to the boundary vertex instead of the midpoint. This avoids manual boundary "protection" and helps maintain geometric fidelity during aggressive coarsening near boundaries (Mesri et al., 2012).

This innovation is particularly important for anisotropic remeshing, where many edges near boundaries need to be collapsed while maintaining domain geometry (Mesri et al., 2012).

Boundary Layer Insertion and Adaptation

For high-Reynolds number flow simulations and other applications requiring fine resolution near boundaries, `SOFIA` provides boundary layer mesh generation capabilities:

- **Structured layer insertion**: Insert vertices at geometric progression distances from boundaries (e.g., $y_i = y_0 \cdot r^i$)
- **Direction-aware metrics**: Compute normal and tangential directions to boundaries for proper metric alignment
- **Progressive refinement**: Smooth transition from very fine resolution at walls to coarse resolution elsewhere

The workflow combines explicit boundary layer construction with metric-driven adaptation for high-fidelity simulations.

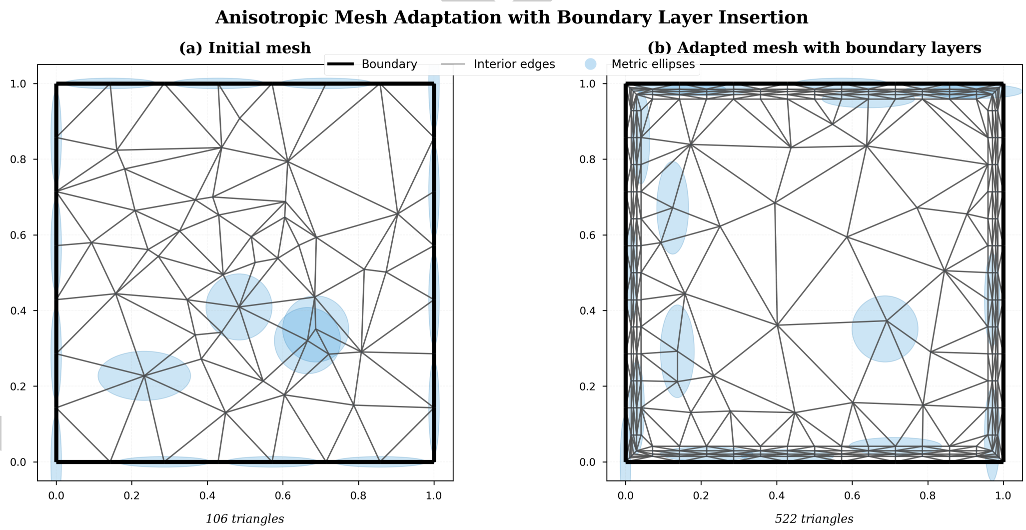

Figure 2: Anisotropic mesh adaptation with boundary layer insertion. Panel (a) shows the initial mesh with 106 triangles, while panel (b) demonstrates the adapted mesh with 522 triangles after inserting 4 boundary layers and performing metric-based adaptation. The boundary layers use a geometric progression with first layer height of 0.015 and growth ratio of 1.4, creating highly anisotropic elements near the boundary while maintaining smooth transitions to isotropic regions in the interior.

Quality Management

The library provides quality assessment and improvement tools with separate handling for isotropic/anisotropic meshes:

- **Quality metrics**: Minimum angle, area ratios, aspect ratios, and shape measures for triangles
- **Metric-based validation**: For anisotropic meshes, quality assessed using metric edge length distribution rather than geometric angles
- **Quality-driven refinement**: Adaptive refinement based on element size, quality thresholds, or user-defined criteria

- **Optimization strategies**: Vertex smoothing with boundary protection, edge flipping cascades, and iterative quality improvement
- **Validation framework**: Incremental topology checking and optional strict validation mode

Adaptive Refinement Workflows

SOFIA supports multiple adaptation strategies for both isotropic/anisotropic meshes:

- **Isotropic refinement**: Uniform or area-based refinement maintaining equilateral triangles
- **Anisotropic refinement**: Metric-driven adaptation with split/collapse/smooth cycles
- **Edge-based refinement**: Selectively refine specific edges (e.g., boundary refinement)
- **Quality-based refinement**: Target low-quality elements for improvement
- **Combined workflows**: Multi-criteria refinement for complex adaptation scenarios
- **Mesh coarsening**: Reduce element count while preserving geometric features and quality
- **Boundary layer generation**: Create highly stretched elements near boundaries for viscous flow simulations

Integration and Extensibility

The modular architecture allows easy integration into existing simulation pipelines:

- **Simple API**: Intuitive mesh editor interface with clear operation semantics
- **Batch operations**: Efficient processing of multiple mesh regions
- **Visualization tools**: Built-in plotting with Matplotlib for analysis and debugging
- **Extensible framework**: Easy to add custom quality metrics or adaptation strategies

Examples

The repository includes comprehensive examples demonstrating various use cases. Each example includes visualization and quantitative metrics. For instance, the simple anisotropic remeshing achieves perfect boundary preservation (max deviation = 0.00e+00) while reducing approximation error by 3×, demonstrating the effectiveness of automatic boundary detection in edge collapse operations (Manzinali et al., 2018; Mesri et al., 2012).

Performance and Testing

SOFIA prioritizes reliability and correctness:

- **Comprehensive test suite**: Over 100 unit tests covering all operations and edge cases
- **Continuous validation**: Tests run on every commit to ensure stability
- **Topology verification**: Built-in checks for mesh conformity and validity
- **Benchmarking**: Performance tests for batch operations and large-scale refinement

The library handles meshes ranging from tens to thousands of elements. The repository also includes benchmarks focused on the split/collapse/flip kernels used during anisotropic remeshing.

State of the Field and Comparison

SOFIA complements and extends existing tools in the Python ecosystem:

- **Triangle/MeshPy** (Kloeckner, 2015; Shewchuk, 1996): Provides Delaunay triangulation and refinement; SOFIA specialises in metric-based anisotropic adaptation and modification with automatic boundary preservation.

- **PyMesh** (Zhou, 2018): Requires substantial C++ dependencies; `SOFIA` aims to remain lightweight with a pure-Python implementation that is easily extensible for research and prototyping.
- **FEniCS/Firedrake mesh tools** (Logg et al., 2012): Useful within FEM frameworks; for metric-based adaptation workflows connected to Firedrake via PETSc/ParMmg see e.g. (Wallwork et al., 2022). `SOFIA` remains framework-agnostic and can be used upstream of different PDE codes.
- **MMG/BAMG and related remeshers** (Dapogny et al., 2014; Hecht, 2012): High-performance anisotropic remeshers; `SOFIA` provides a Python-native workflow and emphasises boundary-aware collapse.

`SOFIA`'s main differentiators are automatic boundary preservation during edge collapse, quality-driven operations with anisotropic-aware checks, and a readable pure-Python codebase that is easy to extend for research and teaching.

Availability

The source code for `SOFIA` is available on GitHub at https://github.com/youssef-mesri/sofia-mesh and is archived on Zenodo (Mesri, 2024) with DOI: 10.5281/zenodo.18492172.

Acknowledgements

We thank the JOSS reviewers for their detailed feedback and constructive suggestions that improved this paper. Development of `SOFIA` has benefited from discussions within the mesh adaptation research community, particularly regarding metric-based anisotropic methods.

References

Alauzet, F. (2010). Parallel anisotropic 3D mesh adaptation by mesh modification. *Engineering with Computers*, *26*(3), 247–258. https://doi.org/10.1007/s00366-005-0009-3

Dapogny, C., Dobrzynski, C., & Frey, P. (2014). Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems. *Journal of Computational Physics*, *262*, 358–378. https://doi.org/10.1016/j.jcp.2014.01.005

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

Hecht, F. (2012). New development in FreeFem++. *J. Numer. Math.*, *20*(3-4), 251–265. https://freefem.org/

Kloeckner, A. (2015). MeshPy: Simplicial mesh generation from python. In *Website repository*. Mathema. https://mathema.tician.de/software/meshpy/

Logg, A., Mardal, K.-A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method. *Springer*. https://doi.org/10.1007/978-3-642-23099-8

Loseille, A., & Alauzet, F. (2011). Continuous mesh framework Part I: Well-posed continuous interpolation error. *SIAM Journal on Numerical Analysis*, *49*(1), 38–60. https://doi.org/10.1137/090754078

Manzinali, G., Hachem, E., & Mesri, Y. (2018). Adaptive stopping criterion for iterative linear solvers combined with anisotropic mesh adaptation, application to convection-dominated problems. *Computer Methods in Applied Mechanics and Engineering*, *340*, 864–880. https://doi.org/10.1016/j.cma.2018.06.025

Mesri, Y. (2024). *SOFIA: Scalable operators for field-driven iso/ani adaptation* (Version v0.1.0). Zenodo. https://doi.org/10.5281/zenodo.18492172

Mesri, Y., Alauzet, F., Loseille, A., Hascoët, L., Koobus, B., & Dervieux, A. (2008). Continuous mesh adaptation models for CFD. *Computational Fluid Dynamics Journal*, *16*(4), 346–355.

Mesri, Y., Guillard, H., & Coupez, T. (2012). Automatic coarsening of three dimensional anisotropic unstructured meshes for multigrid applications. *Applied Mathematics and Computation*, *218*, 10500–10519. https://doi.org/10.1016/j.amc.2012.04.014

Mesri, Y., Khalloufi, M., & Hachem, E. (2016). On optimal simplicial 3D meshes for minimizing the Hessian-based errors. *Applied Numerical Mathematics*, *109*, 235–249. https://doi.org/10.1016/j.apnum.2016.07.007

Mesri, Y., Zerguine, W., Digonnet, H., Silva, L., & Coupez, T. (2008). Dynamic parallel adaption for three dimensional unstructured meshes: Application to interface tracking. In *Proceedings of the 17th international meshing roundtable* (pp. 195–212). Springer. https://doi.org/10.1007/978-3-540-87921-3_12

Persson, P.-O., & Strang, G. (2004). A simple mesh generator in MATLAB. *SIAM Review*, *46*(2), 329–345. https://doi.org/10.1137/S0036144503429121

Shewchuk, J. R. (1996). Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. *Applied Computational Geometry Towards Geometric Engineering*, 203–222. https://doi.org/10.1007/BFb0014497

Wallwork, J. G., Knepley, M. G., Barral, N., & Piggott, M. D. (2022). Parallel metric-based mesh adaptation in PETSc using ParMmg. *International Meshing Roundtable*. https://doi.org/10.48550/arXiv.2201.02806

Zhou, Q. (2018). PyMesh: Geometry processing library for python. In *GitHub repository*. GitHub. https://github.com/PyMesh/PyMesh