Proceedings of the 5th International Conference on Fluid Flow and Thermal Science (ICFFTS 2024)
Lisbon, Portugal- November 21 - 23, 2024
Paper No. 152
DOI: 10.11159/icffts24.152

Gradient Weighted Embedded Error Estimator for Mesh Adaptation

Kunal Ghosh
Indian Institute of Science
C V Raman Avenue, Bengaluru, India
kunalghosh@iisc.ac.in

Abstract - Mesh adaptation is essential for accurate computational fluid dynamics (CFD) simulations, especially when computational resources are limited. In simulating the advection-diffusion equation using the finite volume method (FVM), mesh adaptation is often necessary to accurately capture sharp gradients and complex flow features. Although the metric-based mesh adaptation used in this study is highly effective with true errors, these are typically unavailable, requiring the use of error estimators. However, adjoint-based error estimators, though effective, are computationally expensive as they require solving the larger adjoint system. To address this, an inexpensive error estimator based on the gradient of the numerical solution and the embedded method is proposed. Specifically, the central difference and upwind difference schemes are utilized for error estimation within this framework. Since errors are predominantly influenced by sharp gradients in the scalar boundary layer, a target functional based on the product of the embedded error and the solution gradient to compute the error estimator is proposed. Consequently, these proposed error estimators generate meshes closely resembling those produced using true error values, effectively resolving the scalar boundary layer. As a result, the L^2 and L^∞ norms of the error typically decrease with successive adaptation cycles. Furthermore, the L^1, L^2, and L^∞ norms of error per element also reduce with successive adaptation cycles. In conclusion, the proposed error estimator facilitates mesh adaptation without needing true errors or computationally expensive adjoint-based error estimators, thus ensuring accurate numerical solutions, especially in scenarios with sharp boundary layers.

***Keywords*:** Metric-based mesh adaptation, Embedded methods, CFD, FVM, Scalar Boundary Layer

1. Introduction

NASA's CFD Vision 2030 report [1] identifies mesh generation and adaptation as critical challenges to advance CFD methodologies. Addressing these challenges is essential to accurately estimate the functional outputs, which requires resolving all significant flow features. While uniform grid refinement can achieve this, it becomes prohibitively expensive for complex geometries. In contrast, heuristic-based mesh refinement, which focuses on high-gradient areas, is more cost-effective but does not guarantee convergence. Alternatively, manually generating grids tailored to specific flow features is labour-intensive and suitable only for simple cases where feature locations are known in advance.

The fluid flow phenomena of interest typically involve complex anisotropic characteristics. These can be effectively addressed through mesh adaptation. Specifically, metric field-based mesh adaptation offers a robust mathematical framework for managing both isotropic and anisotropic cases [2]. Moreover, anisotropic meshes can be naturally created by designing a unit mesh in a Riemannian metric space and transforming it into Euclidean space, as demonstrated by Hecht and Mohammad [3]. This method has proven effective in generating highly anisotropic meshes, particularly within boundary layers in 2D and 3D steady-state CFD simulations for the HDG method [4]. Building on this, Dolejsi [5] developed a comprehensive framework to generate metric fields for high-order hp-adaptive methods by minimizing interpolation error in the L^q-norm. Balan et al. [6] further advanced this approach by applying an adjoint-based hp-adaptation methodology for high-order DG schemes in nonlinear convection-diffusion problems, deriving optimal anisotropy based on Dolejsi's method [5] and mesh size using dual-weighted residuals. Additionally, Rangarajan et al. [7] developed a framework for anisotropic h-adaptation in 2D for high-order methods, optimizing the mesh globally with respect to the L^q-norm of interpolation error. This framework was later extended to hp-adaptation [8] and 3D cases [9], achieving optimal convergence orders in 2D and 3D simulations. These methods by Rangarajan et al. [7 and 9] were further extended to goal-oriented mesh adaptation [10 and 11], demonstrating improved convergence compared to h- and hp-adaptation alone. Furthermore, Rangarajan et al. [12] established analytical formulas for determining mesh density and mesh anisotropy using error estimators, which forms the basis for this work. For an extensive review of various error estimates for mesh adaptation and their convergence properties across different flow problems, Balan et al. [13] provide an extensive analysis. On the other hand, Alauzet et al. [14] offer a comprehensive review of mesh adaptation techniques.

Despite the advancements in mesh adaptation, current methodologies face significant limitations, primarily due to the high computational cost of solving adjoint equations. While these methods have proven effective, their computational expense can be prohibitive. Moreover, computationally cheaper alternatives, such as those based on embedded methods, have yet to be thoroughly explored for this specific mesh adaptation approach [12].

In this work, the possibility of developing an error estimator that is both computationally inexpensive and capable of effectively resolving critical flow features is explored. To achieve this, a novel target functional aimed at creating a cost-effective error estimator is proposed. This functional leverages the embedded method in combination with the gradient of the numerical solution, enabling accurate error estimation without the need for solving the computationally expensive adjoint system of equations. This error estimation is then used for mesh adaptation. Thus, this approach offers a more efficient alternative for mesh adaptation within the framework proposed by Rangarajan [12].

2. Methodology

In this work, the improvement in the mesh adaptation is demonstrated for the scalar boundary layer case in the advection-diffusion equation (Eqn. 1).

$$\nabla \cdot \left(\boldsymbol{c}\phi\right) - \nabla \cdot \left(\mu \, \nabla \phi\right) - S^{\phi} = 0 \tag{1}$$

In Eqn. 1, S^{ϕ} represents the source term, ϕ denotes the state variable, $\boldsymbol{c}$ is the advection speed, μ is the diffusion coefficient and the set of constants in the equation is $\mathbf{P} = \{\mathbf{c}, \mu\}$.

Let's rewrite the governing equation (Eqn. 1) as follows:

$$\mathcal{N}\left(\phi, \boldsymbol{P}\right) = 0 \tag{2}$$

Where, $\mathcal{N}\left(\phi, \boldsymbol{P}\right) = \nabla \cdot \left(\boldsymbol{c}\phi\right) - \nabla \cdot \left(\mu \, \nabla \phi\right) - S^{\phi}$

Assume the objective is to minimize a cost function $\mathcal{J}(\phi, \boldsymbol{P})$.

So, the problem can be formulated using a Lagrange multiplier as follows:

$$\mathscr{L}\left(\phi, \boldsymbol{P}, \lambda\right) = \mathcal{J}(\phi, \boldsymbol{P}) + \lambda \mathcal{N}(\phi, \boldsymbol{P}) \tag{3}$$

In Eqn. 3, $\mathscr{L}\left(\phi, \boldsymbol{P}, \lambda\right)$ represents the Lagrangian, $\mathcal{J}(\phi, \boldsymbol{P})$ denotes the function to be minimized, λ is the Lagrange multiplier, and $\mathcal{N}(\phi, \boldsymbol{P})$ is the equality constraint. Here, the goal is to minimize $\mathcal{J}(\phi, \boldsymbol{P})$ subject to the constraint $\mathcal{N}\left(\phi, \boldsymbol{P}\right) = 0$, which represents the governing equation.

On differentiating the Eqn. 3, with respect to λ and setting the derivative equal to 0, the governing equation can be recovered.

$$\frac{\partial \mathscr{L}}{\partial \lambda} = 0 \rightarrow \mathcal{N}(\phi, \boldsymbol{P}) = 0 \tag{4}$$

Similarly, on differentiating the Eqn. 3, with respect to ϕ and setting the derivative equal to zero yields:

$$\frac{\partial \mathscr{L}}{\partial \phi} = \frac{\partial \mathcal{J}}{\partial \phi} + \lambda \frac{\partial \mathcal{N}}{\partial \phi} = 0 \rightarrow \frac{\partial \mathcal{J}}{\partial \phi} = -\lambda \frac{\partial \mathcal{N}}{\partial \phi} \tag{5}$$

This results in the adjoint equation, which can be expressed in its discretized form as follows:

$$\frac{\mathcal{J}(\phi,\boldsymbol{P}) - \mathcal{J}(\phi_h,\boldsymbol{P})}{\phi - \phi_h} = -\lambda\left(\frac{\mathcal{N}(\phi,\boldsymbol{P}) - \mathcal{N}(\phi_h,\boldsymbol{P})}{\phi - \phi_h}\right) \tag{6}$$

So,

$$\mathcal{J}(\phi,\boldsymbol{P}) - \mathcal{J}(\phi_h,\boldsymbol{P}) = -\lambda\left(\mathcal{N}(\phi,\boldsymbol{P}) - \mathcal{N}(\phi_h,\boldsymbol{P})\right) \tag{7}$$

In this equation, ϕ represents the true solution, while ϕ_h denotes the numerical solution.
As the governing equation is $\mathcal{N}(\phi,\boldsymbol{P}) = 0$, it follows that:

$$\mathcal{J}(\phi,\boldsymbol{P}) - \mathcal{J}(\phi_h,\boldsymbol{P}) = \lambda\mathcal{N}(\phi_h,\boldsymbol{P}) \tag{8}$$

So,

$$\lambda = \frac{\mathcal{J}(\phi,\boldsymbol{P}) - \mathcal{J}(\phi_h,\boldsymbol{P})}{\mathcal{N}(\phi_h,\boldsymbol{P})} \tag{9}$$

Let's assume that the cost function is

$$\mathcal{J}(\phi_h,\boldsymbol{P}) = |\phi - \phi_h| \tag{10}$$

So,

$$\mathcal{J}(\phi,\boldsymbol{P}) = |\phi - \phi| = 0 \tag{11}$$

Thus, using the cost function (Eqn. 11), the adjoint variable (Eqn. 9) can be expressed as:

$$\lambda = -\frac{\mathcal{J}(\phi_h,\boldsymbol{P})}{\mathcal{N}(\phi_h,\boldsymbol{P})} = -\frac{|\phi - \phi_h|}{\mathcal{N}(\phi_h,\boldsymbol{P})} \tag{12}$$

So, the adjoint weighted residual error estimate η_κ across the computational domain Ω is

$$\eta_\kappa = \left|\int_\Omega \lambda\,\mathcal{N}(\phi_h,\boldsymbol{P})d\Omega\right| \tag{13}$$

Thus, by applying Eqn. 12, it follows that

$$\eta_\kappa = \left|\int_\Omega -\frac{|\phi - \phi_h|}{\mathcal{N}(\phi_h,\boldsymbol{P})}\mathcal{N}(\phi_h,\boldsymbol{P})d\Omega\right| \rightarrow \eta_\kappa = \left|\int_\Omega -|\phi - \phi_h|d\Omega\right| \tag{14}$$

For the purpose of error estimation, this study assumes that the true solution is approximated using a second-order accurate central difference scheme (CDS), while the numerical solution is approximated using a first-order accurate upwind difference scheme (UDS). Consequently, $|\phi - \phi_h| \approx |\phi_{CDS} - \phi_{UDS}|$, where ϕ_{CDS} and ϕ_{UDS} are the numerical solutions computed using the CDS and UDS, respectively. However, this cost function struggled to refine the regions of very sharp gradients, potentially due to the smaller difference in the solutions in the embedded method, [24]. This limitation might be addressed by employing higher-order FVM schemes, which provide a more accurate numerical solution. Nevertheless, extending widely used FVM methods [23] to arbitrary higher orders is challenging compared to methods such as the finite element method (FEM) [25], discontinuous Galerkin (DG) [26], or hybridized discontinuous Galerkin (HDG) [12], which can more easily accommodate higher-order extensions.

So, a new cost function for the FVM method is proposed as follows:

$$\mathcal{J}\left(\phi_{h'}\boldsymbol{P}\right) = \left|\phi - \phi_h\right| \left|\nabla \phi_h\right| \tag{15}$$

This is because in this problem the error is primarily due to the sharp gradient of the scalar boundary layer. So, the new adjoint variable (Eqn. 9) will be

$$\lambda = \frac{\mathcal{J}\left(\phi,\boldsymbol{P}\right) - \mathcal{J}\left(\phi_{h'}\boldsymbol{P}\right)}{\mathcal{N}\left(\phi_{h'}\,\boldsymbol{P}\right)} \tag{16}$$

So,

$$\lambda = \frac{|\phi - \phi|\left|\nabla \phi\right| - \left|\phi - \phi_h\right|\left|\nabla \phi_h\right|}{\mathcal{N}\left(\phi_{h'}\,\boldsymbol{P}\right)} = -\left(\frac{\left|\phi - \phi_h\right|\left|\nabla \phi_h\right|}{\mathcal{N}\left(\phi_{h'}\,\boldsymbol{P}\right)}\right) \tag{17}$$

So, the adjoint weighted residual error estimate (η_κ) (Eqn. 13) is

$$\eta_\kappa = \left|\int_\Omega - \frac{\left|\phi - \phi_h\right|\left|\nabla \phi_h\right|}{\mathcal{N}\left(\phi_{h'}\boldsymbol{P}\right)}\mathcal{N}\left(\phi_{h'}\boldsymbol{P}\right)d\Omega\right| \rightarrow \eta_\kappa = \left|\int_\Omega -\left|\phi - \phi_h\right|\left|\nabla \phi_h\right| d\Omega\right| \tag{18}$$

So,

$$\eta_\kappa \approx \left|\int_\Omega -\left|\phi_{CDS} - \phi_{UDS}\right|\left|\nabla \phi_{CDS}\right| d\Omega\right| \tag{19}$$

Once the adjoint-based error estimator is evaluated, the isotropic mesh adaptation procedure described in [12] is followed with any necessary adjustments. This approach results in improved mesh adaptation compared to relying solely on the embedded error. For concise notation, this error estimator will be referred to as the gradient-weighted embedded error estimator or GWEEE.

In this study, the solution is taken from [12] as follows:

$$\phi = \left(x + \frac{e^{\beta_1 x/\mu} - 1}{1 - e^{\beta_1/\mu}}\right)\left(y + \frac{e^{\beta_2 y/\mu} - 1}{1 - e^{\beta_2/\mu}}\right) \tag{20}$$

where, $\boldsymbol{c} = \beta_1\hat{x} + \beta_2\hat{y}$, with β_1 and β_2 representing the advection speeds in the x and y directions, respectively.

Substituting this ϕ into the advection-diffusion equation yields an expression for the source term, S^ϕ. This expression for S^ϕ is then used to solve the advection-diffusion equation using FVM [16, 18, 19, 20, 21, 22 and 24]. This particular expression is chosen for the advection-diffusion equation because it produces a sharp scalar boundary layer for smaller values of μ, as illustrated in Fig. 1. All the plots presented in this work are generated using [17] with parameters set to $\beta_1 = 10$, $\beta_2 = 10$ and $\mu = 0.4$.

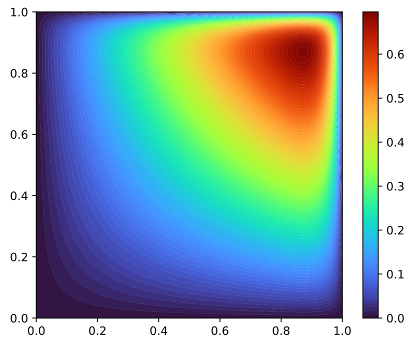

Fig. 1: True Solution

3. Results

The mesh generated by the proposed error estimator (GWEEE) closely resembles the one produced using true error values, which are typically unavailable in practical scenarios. This similarity is evident in Fig. 2 and 3, which illustrate the meshes and demonstrate the effectiveness of the proposed approach in accurately capturing essential flow features.

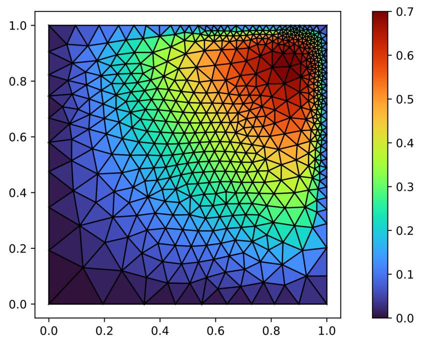

Fig. 2: Mesh generated using GWEEE

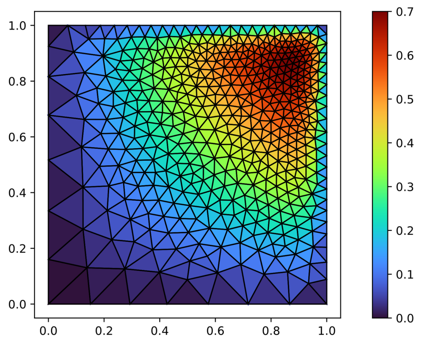

Fig. 3: Mesh generated using true error

Additionally, the proposed target functional effectively resolves the scalar boundary layer. This is demonstrated in the contour plots presented in Fig. 4 and 5.

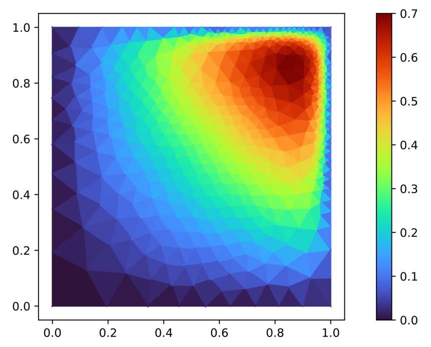

Fig. 4: Contour of the solution for GWEEE

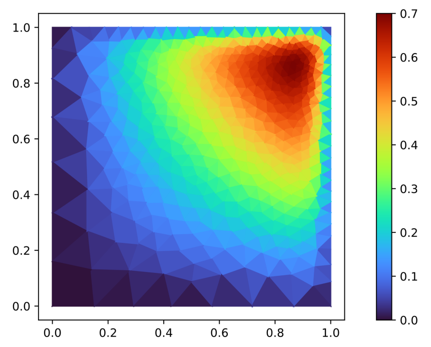

Fig. 5: Contour of the solution for true error

Moreover, the L^2, and L^∞ norms of the error typically decrease before plateauing over successive iterations of adaptation for GWEEE. This trend is clearly illustrated in Fig. 6 and 7. Additionally, the error norms for the numerical solution on the mesh generated using the proposed target functional are usually lower than those on the mesh produced with true error values.

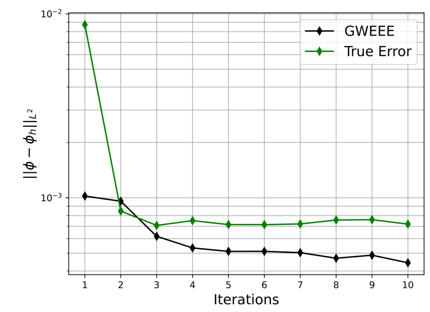

Fig. 6: L^2-Norm of the error

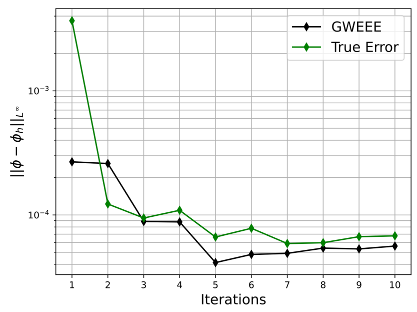

Fig. 7: L^∞-Norm of the error

Notably, despite using the same target degrees of freedom in terms of the total number of elements, there is a slight variation between the total number of elements in true error case and the proposed error estimator. This discrepancy arises because the metric-based mesh generator, BAMG [15], constructs meshes that are optimal only in a least-squares sense. In addition to this, the average error norms over the elements for both cases were also evaluated. The analysis revealed that the average L^1, L^2, and L^∞ norms of error per cell decrease before eventually plateauing with successive mesh adaptation cycles. This trend is illustrated in Fig. 8, 9, and 10. Furthermore, the error norms per cell for the mesh generated using the proposed target functional are lower compared to those for the mesh generated using the true error values.

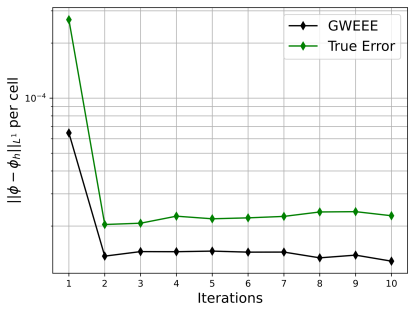

Fig. 8: L^1-Norm of the error per cell

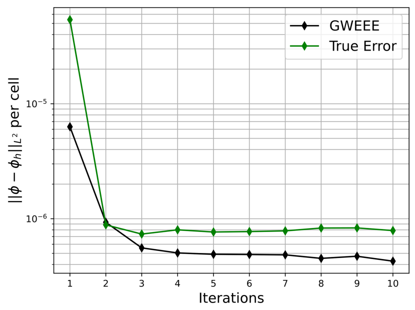

Fig. 9: L^2-Norm of the error per cell

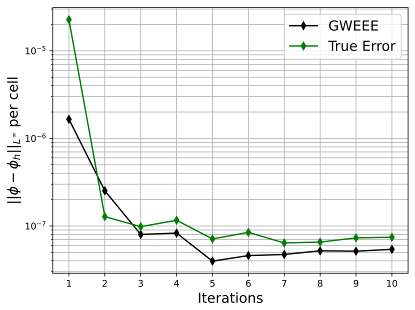

Fig. 10: L^∞-Norm of the error per cell

4. Conclusion

In conclusion, the proposed error estimator facilitates efficient mesh adaptation with significantly reduced computational costs, achieving accurate numerical solutions. This efficiency arises from bypassing the need to solve the larger adjoint system typically required for adjoint-based error estimators, as the proposed methodology utilizes the embedded method instead. For the scenarios considered in this study, particularly those involving sharp boundary layers, the proposed estimator delivers superior results compared to the true error. Therefore, this error estimator is well-suited for applications involving sharp boundary layers.

References

[1] J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie and D. J. Mavriplis, "CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences," Hampton, VA, Langley Research Center, 2014.

[2] P. J. Frey and F. Alauzet, "Anisotropic mesh adaptation for CFD computations," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 48-49, pp. 5068-5082, 2005.

[3] F. Hecht and B. Mohammadi, "Mesh adaption by metric control for multi-scale phenomena and turbulence," in *35th Aerospace Sciences Meeting and Exhibit*, 1997, pp. 859.

[4] A. Balan, M. A. Park, S. Wood and W. K. Anderson, "Verification of Anisotropic Mesh Adaptation for Complex Aerospace Applications," in *AIAA Scitech 2020 Forum*, pp. 0675, 2020.

[5] V. Dolejší, "Anisotropic hp-adaptive method based on interpolation error estimates in the Lq-norm," *Applied Numerical Mathematics*, vol. 82, pp. 80-114, 2014.

[6] A. Balan, M. Woopen and G. May, "Adjoint-based hp-adaptivity on anisotropic meshes for high-order compressible flow simulations," *Computers & Fluids*, vol. 139, pp. 47-67, 2016.

[7] A. M. Rangarajan, A. Balan and G. May, "Mesh Optimization for Discontinuous Galerkin Methods Using a Continuous Mesh Model," *AIAA Journal*, vol. 56, no. 10, pp. 4060-4073, 2018.

[8] V. Dolejší, G. May and A. M. Rangarajan, "A continuous hp-mesh model for adaptive discontinuous Galerkin schemes," *Applied Numerical Mathematics,* vol. 124, pp. 1–21, 2018.

[9] A. M. Rangarajan, A. Chakraborty, G. May and V. Dolejší, "A continuous-mesh optimization technique for piecewise polynomial approximation on tetrahedral grids," in 2018 *Fluid Dynamics Conference*, 2018, pp. 3246.

[10] A. M. Rangarajan, G. May and V. Dolejší, "Adjoint-based anisotropic hp-adaptation for discontinuous Galerkin methods using a continuous mesh model," *Journal of Computational Physics*, vol. 409, pp. 109321, 2020.

[11] A. M. Rangarajan, A. Chakraborthy and G. May, "A goal oriented optimization technique for tetrahedral grids using a continuous-mesh model," in *AIAA Scitech 2019 Forum*, 2019, pp. 0349.

[12] A. M. Rangarajan, "Metric Based *hp*−Adaptation Using a Continuous Mesh Model for Higher Order Schemes," Ph.D. dissertation, Dept. Mech. Eng., RWTH Aachen Univ., Aachen, NRW.

[13] A. Balan, M. A. Park, S. L. Wood, W. K. Anderson, A. M. Rangarajan, D. P. Sanjaya and G. May, "A review and comparison of error estimators for anisotropic mesh adaptation for flow simulations," *Computers & Fluids*, vol. 234, pp. 105259, 2022.

[14] F. Alauzet and A. Loseille, "A decade of progress on anisotropic mesh adaptation for computational fluid dynamics," *Computer-Aided Design*, vol. 72, pp. 13-39, 2016.

[15] F. Hecht, "BAMG: bidimensional anisotropic mesh generator," *User Guide. INRIA, Rocquencourt*, vol. 17, 1998.

[16] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Rio, M. Wiebe, P. Peterson,

P. G. Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke and T. E. Oliphant, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357-362, 2020.

[17] J. D. Hunter, “Matplotlib: A 2D graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90-95, 2007.

[18] The pandas development team, “pandas-dev/pandas: Pandas,” *Zenodo*, 2020.

[19] W. McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 56-61.

[20] N. Bell, L. N. Olson, J. Schroder and B. Southworth, “PyAMG: Algebraic Multigrid Solvers in Python,” *Journal of Open Source Software*, vol. 8, no. 87, 2023.

[21] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, I. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. Ribeiro, F. Pedregosa, P. van Mulbregt and SciPy 1.0 Contributors, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, pp. 261-272, 2020.

[22] S. K. Lam, A. Pitrou and S. Seibert, “Numba: A llvm-based python jit compiler,” in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 2015, pp. 1-6.

[23] C. R. Maliska, *Fundamentals of Computational Fluid Dynamics*. Cham, Switzerland: Springer Nature Switzerland *AG*, 2023.

[24] J. R. Dormand, *Numerical Methods for Differential Equations: A Computational Approach*, Boca Raton. CRC Press, 1996.

[25] M. G. Larson and F. Bengzon, *The Finite Element Method: Theory, Implementation, and Applications*, Heidelberg. Springer Berlin, 2013.

[26] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, NY, Springer New York, 2008.