



Efficient multi-stage aerodynamic topology optimization using an operator-based analytical differentiation

Ali Ghasemi¹ · Ali Elham²

Received: 20 July 2021 / Revised: 17 November 2021 / Accepted: 19 February 2022 / Published online: 31 March 2022
© The Author(s) 2022

Abstract

A high-performance density-based topology optimization tool is presented for laminar flows with focus on 2D and 3D aerodynamic problems via OpenFOAM software. Density-based methods are generally robust in terms of initial design, making them suitable for designing purposes. However, these methods require relatively fine resolutions for external flow problems to accurately capture the solid-fluid interfaces on Cartesian meshes, which makes them computationally very expensive, particularly for 3D problems. To address such high computational costs, two techniques are developed here. Firstly, an operator-based analytical differentiation (OAD) is proposed, which efficiently computes the exact partial derivatives of the flow solver (simpleFOAM). OAD also facilitates a convenient development process by minimizing hand-coding and utilizing the chain-rule technique, in contrast to full hand-differentiation, which is very complex and prone to implementation errors. Secondly, a multi-stage design process is proposed to further reduce the computational costs. In this technique, instead of using a fixed refined mesh, the optimization processes are initiated with a coarse mesh, and the converged solutions are projected to a locally refined mesh (as an initial guess) for a secondary optimization stage, which can be repeated to obtain a sufficient accuracy. A set of 2D and 3D laminar aerodynamic problems were studied, which promisingly confirmed the utility of the present approach, which can be adopted as a starting point for developing a design tool for large-scale aerodynamic engineering applications. In addition, the 3D problems indicated that less than 3% of total optimization CPU-time is devoted to OAD, and multi-staging up to 45% has reduced the overall costs.

Keywords Topology optimization · Operator-based analytical differentiation (OAD) · Multi-stage design · Discrete adjoint · Laminar aerodynamics · OpenFOAM

1 Introduction

Topology optimization (TO) is a powerful computational tool for enhancing or designing engineering systems with wide-ranging applications. TO fundamentally features no limitation in modifying the design configuration, during the optimization process, in contrast to size or shape optimization techniques. Consequently, TO process could be robust in terms of initial design guess, facilitating an optimization

process, starting from scratch. Such strength is significantly useful for the designing processes, which time-consumingly rely on trial-and-error or the designer's intuition. Hence, TO can be particularly useful for conceptual designs in applications where limited or no intuition are available.

Initially, TO was developed for structural optimization using a homogenization technique (Bendsøe and Kikuchi 1988), searching for an optimal void-solid material distribution. This approach (so-called the density-based method) was later applied to Stokes flow problems (Borrvall and Petersson 2003), by introducing a porosity field with continuously variable impermeability as the design parameter. Since then, TO techniques have been evolved in various aspects and applied to problems such as unsteady flows (Kreissl et al. 2011), turbulent flows (Yoon 2016), heat convection (Dede 2009), fluid-structure interaction (Yoon 2010), with many different approaches (Alexandersen and Andreasen 2020). It is noticeable that the majority of the previous works have

Responsible Editor: Anton Eyrafov.

✉ Ali Ghasemi
a.ghasemi@tu-braunschweig.de

¹ Institute of Aircraft Design and Lightweight Structures, Technische Universität Braunschweig, Braunschweig, Germany

² University of Southampton, Southampton SO16 7QF, UK

mainly focused on internal (channel) flows. State-of-the-art TO developments for external flow problems are limited to two-dimensional cases and/or very low Reynolds numbers, such as (Kondoh et al. 2012; Garcke et al. 2018; Lundgaard et al. 2018). The present work focuses on two and three-dimensional laminar aerodynamic TO, which can be utilized for optimization of small-sized flying objects such as fixed-wing micro air vehicles (MAV) (Pornsin-Sirirak et al. 2001).

Level-set and density-based methods are two main approaches of TO, each with advantages and disadvantages. The primary strength of level-set approaches is the clear-cut description of solid-fluid boundaries, if they are equipped with interface capturing techniques such as CutFEM (Villanueva and Maute 2017) or explicit (body-fitted) boundary tracking (Zhou et al. 2018; Feppon et al. 2020). This maintains a high level of model accuracy, particularly for capturing the boundary layers, during the optimization process. However, a critical drawback of such methods is that they mostly fail to create new isolated zones or split an existing one, if needed during the optimization process. This implies that the number of solid zones in final design would be less than or equal to the number of zones in initial design. As a result, such methods greatly depend on the initial guess (Jenkins and Maute 2015), and may not satisfyingly fulfill the important factor of designing from scratch. Besides, providing an appropriate initial guess can potentially be difficult, particularly for high-speed external flows, where the flow steadiness and the behavior of boundary layers are sensitive to the geometry. In contrast, density-based methods indicate no fundamental dependency on initial designs, in the sense that they are able to unlimitedly create, split or merge solid zones during optimization process (although the initial guess can affect the final solution (local minima) in gradient-based optimization). As an example, density-based method has been used together with level-set method for enabling a hole-seeding feature during the optimization (Barrera et al. 2020). Hence, density-based method is a promising approach for both design and optimization purposes.

In density-based methods, solid materials are mostly modeled by techniques such as Brinkman volume penalization (BVP) on fixed meshes. The caveat is that such methods may suffer from relatively inaccurate approximation of solid-fluid interfaces. Such inaccuracy can be distinguished by modeling and discretization (meshing) errors. On a body-fitted mesh, it is suggested that the modeling error could be satisfyingly small $\mathcal{O}(\epsilon)$ (ϵ as the penalization parameter) compared to classical no-slip boundary conditions (Hester et al. 2021) (compare Fig. 1a, b). On a non-conforming (Cartesian) mesh, however, the solid geometry lacks a smooth solid-interface description (Fig. 1c). This problem can be greatly improved by local mesh refinements, as depicted in Fig. 1d. A block mesh refinement of the design space features less re-meshing costs, compared to adaptive interface tracking used in level-set methods, however, it raises the problem size, as the number of state/design variables increases. To this end, an efficient computational tool with a refinement strategy can play an important role for achieving an accurate density-based TO, particularly for external flow problems.

The computational costs are divided into two main parts: the numerical flow (primal) solver, and numerical computation of sensitivities, as TO is a gradient-based, numerical optimization process. For the primal solver, the finite volume method (FVM) based OpenFOAM software is used, which is efficient and parallelized. For computing sensitivities of a given cost function, adjoint methods are widely used, which effectively demonstrate computational costs, independent of the number of design variables. Continuous and discrete adjoint methods are two different adjoint approaches (see Nadarajah and Jameson (2000); Evgrafov et al. (2011) for comparisons). Continuous adjoint sensitivities are obtained by numerically solving PDEs, which may introduce some discretization errors and mesh-dependent accuracies (Carnarius et al. 2011; Yu et al. 2020). Conversely, discrete adjoint methods, independent of mesh size, can provide exact sensitivities, which are consistent to the value of cost function, obtained from primal solver. Exact sensitivities

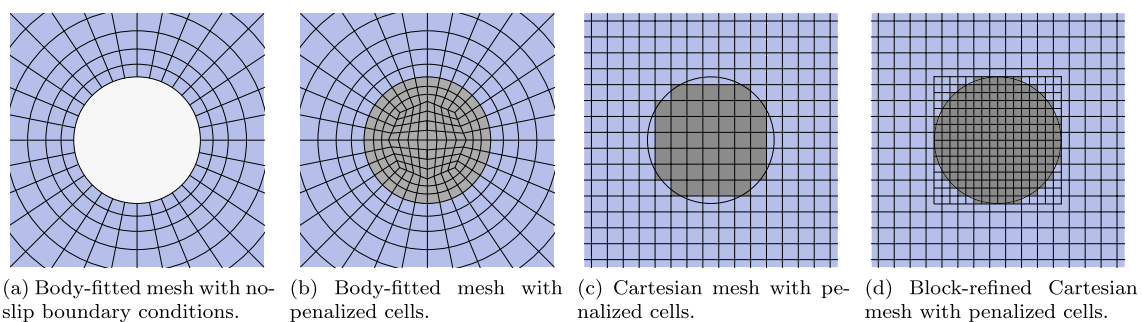


Fig. 1 Schematic of solid-fluid interface descriptions using volume penalization on different meshes, compared to classical no-slip boundary conditions

are sometimes vital, particularly for external or turbulent flows (Dilgen et al. 2018), and most of the modern adjoint-based optimization tools utilize the discrete adjoint methods (Peter and Dwight 2010; Kenway et al. 2019); hence, discrete adjoint method is utilized in this work, as well.

For computation of discrete adjoint sensitivities, two steps are required: computing derivatives of residuals $\mathbf{R}(\mathbf{U}, \boldsymbol{\gamma})$ with respect to state \mathbf{U} and design variables $\boldsymbol{\gamma}$, and solving the adjoint linear system of equations. Regarding the adjoint problem (second step), a suitable iterative linear solver (e.g. preconditioned Krylov subspace method), via the efficient and parallelized PETSc library (Balay et al. 2021), can provide accurate results with reasonably fast convergence speed. However, effectively (and precisely) deriving the partial derivatives of \mathbf{R} (step two) is a challenging task, particularly for the sophisticated CFD solvers. For this task, two different approaches can be used: analytical, or automatic differentiation (AD) (Griewank and Walther 2008). Either of approaches feature a trade-off between the implementation efforts and overall performance, although the first one could be faster and less memory consuming (Nørgaard et al. 2017; Kenway et al. 2019). This is because once the mathematical expression of \mathbf{R} partial derivatives are explicitly available, then calculation of derivative values would be very fast. As a result, the analytical approach is employed in this work, as the computational efficiency is of primary concern.

Analytical approach, via hand differentiation (HD), has been used for some early shape optimizations, such as (Nielsen and Anderson 1999), and been implemented limitedly, for few closed-source codes, such as DLR-Tau-code (Dwight and Brezillon 2006). However, HD has some drawbacks: it is a non-trivial task (with difficulties, proportional to the complexity of solvers), and prone to implementation (human) errors (Brezillon and Dwight 2005). As a result, in recent years AD has been preferred for advanced solvers, such as OpenFOAM (He et al. 2020), and SU2 (Economon et al. 2016). As an alternative to HD, symbolic differentiation (SD) has been recently developed for an FEM-based Euler solver for an error-prove, and fast discrete adjoint developments (Elham and van Tooren 2021). However, for SD the code has to be rewritten symbolically before differentiation, which is not trivial for all CFD solvers. For density-based TO, analytical differentiation has been previously used for solvers with comparatively simpler mathematical formulations, such as lattice Boltzmann (LB) (Pingen et al. 2007), and pseudo-spectral (Ghasemi and Elham 2021). Another example is hand-coded discrete adjoint, developed for TO using a staggered-based FVM solver (Marck et al. 2013), which, in favor of a convenient analytical differentiation, has a simpler formulation compared to collocated FVM. Nevertheless, such solvers are nowadays outdated, and modern FVM-based CFD software (e.g. OpenFOAM) are collocated-based (Moukalled et al. 2016). In collocated-based

FVM solvers (using SIMPLE algorithm), the pressure corrector (Poisson) equation is constructed (via Rhie-Chow interpolation) by discretization coefficients of the velocity equations (including the penalization term), and solved iteratively. This formulation is rather complex, due to the nested dependency of solution states, which has to be untangled for analytical differentiation process. Besides, the discretization formulation varies based on the type of FVM scheme, which can lead to a very slow and case-dependent implementation process. Consequently, despite the advantages of analytical differentiation, it can still be a highly complicated task, since no general approach has been proposed so far to alleviate its complexity.

To address the aforementioned issues, an operator-based analytical differentiation (OAD) is developed here, which has three advantages. Firstly, OAD untangles the complex residuals formulations, by expressing them in matrix forms using a set of linear operators, which clarifies the dependency of residuals on state and design variables, and facilitates the differentiation process by maximizing the use of chain-rule and minimizing hand-coding. More precisely, the derivatives are simply computed at discretization level, which greatly reduces the risk of implementation errors. Secondly, for different FVM schemes simply the corresponding operator needs to be updated with minimal effort (instead of restarting differentiation from scratch). Thirdly, OAD preserves a reasonably high performance, due to the low cost of the matrix operators for highly sparse matrices, in favor of more efficient discrete sensitivity computations.

Lastly, to further reduce the overall cost of the TO process a multi-stage procedure is employed. As mentioned before, a sufficiently fine mesh is required for an improved accuracy of solid-fluid interfaces. However, using a fine mesh on the entire simulation domain, especially in aerodynamic problems, from the beginning of optimization process would be either very expensive or waste of computational efforts. Also, the geometry of design, which needs finer meshing, is not easily predictable in advance. To address this issue, TO can be initiated on a coarser mesh, and then optimized design can be projected onto a new mesh with refined design space. Generally, the secondary optimization stages converge relatively faster, as their starting states are an optimized solution on a coarser mesh. As a result, this technique can reduce the overall computational costs, particularly for the design problems, starting from scratch. Finally, a set of 2D/3D optimization problems, such as drag minimization or lift maximization, are solved to better demonstrate the potential utilities of the present approach, which can benefit a vast amount of applications and future progresses, especially for aerodynamic topology optimizations.

Thus, the novelties are summarized as: (1) proposing an operator-based approach (OAD) for a high performance and exact (analytical) differentiation of simpleFOAM solver in

OpenFOAM (collocated FVM flow solver), which facilitates rapid and convenient implementations for further developments, (2) utilizing a multi-stage (multi-resolution) procedure for achieving an improved model accuracy with less computational efforts, (3) exploring the potential ability of density-based topology optimization technique for 2D/3D laminar aerodynamic problems, via the proposed efficient and parallelized open-source design tool. The remainder of the paper is organized as follows: in Sect. 2 the flow governing equations as well as the numerical solve procedure is presented; in Sect. 3 the topology optimization formulation and algorithm is discussed together with the sensitivity analysis; a set of 2D and 3D optimization examples are solved in Sect. 4 and discussed in 5; finally, Sect. 6 summarized the achievements of the present work.

2 Fluid flow problem

2.1 Penalized Navier–Stokes

The steady-state incompressible Navier-Stokes (NS) equations equipped with Brinkman volume penalization (BVP) are

$$\nabla \cdot (\mathbf{u}\mathbf{u}) - \nabla \cdot (\nu \nabla \mathbf{u}) + \frac{1}{\epsilon} \chi \mathbf{u} = -\frac{1}{\rho} \nabla p, \quad (1)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (2)$$

where \mathbf{u} is the velocity vector, p is the dynamic pressure, ρ is the fluid density, ν is the kinematic viscosity, and ϵ and χ are the penalization intensity (permeability) and material identifier parameters, respectively. At any point \mathbf{x} , solid and fluid materials are then modeled using penalization effect which is controlled by χ , such that

$$\mathbf{x} \in \begin{cases} \text{Solid zones } (\Omega_S) & \text{if } \chi = 1 \\ \text{Fluid zones } (\Omega_F) & \text{if } \chi = 0. \end{cases} \quad (3)$$

It is shown that the solution of penalized NS equations converges to the NS solutions with classical boundary conditions (zero velocities on $\partial\Omega_F$) (Carbou and Fabrie 2003) with

$$\|\mathbf{u} - \mathbf{u}_{\text{penalized}}\| \leq C\epsilon^{1/2}, \quad (4)$$

which indicates smaller the ϵ , better the penalization (modeling) accuracy.

2.2 Flow (primal) solver

A standard finite volume method (FVM) is used for discretization of penalized incompressible Navier-Stokes equations, which is widely utilized for CFD applications. The

Rhie-Chow interpolation method (Rhie and Chow 1983) is used for an oscillation-free computation of pressure gradients on collocated meshes, and the flow solutions are obtained, iteratively, via the segregated SIMPLE algorithm (Patankar and Spalding 1983). In this study, the native simpleFoam solver of OpenFOAM CFD software is modified to incorporate the Brinkman penalization term in the left-hand-side of the momentum equation, similar to Eq. (1).

As it will be discussed in Sect. 3.3, for the analytical differentiation of the flow solver residuals, a basic knowledge of the discretization process, as well as the SIMPLE algorithm, is essentially required. Therefore, such details are discussed sufficiently in Appendix A.

3 Topology optimization

3.1 Interpolation of materials

Motivated by Borrvall and Petersson (2003), a density-based approach is adopted, in which a porosity field is utilized to describe the design. More precisely, a continuous interpolation function is used for χ in Eq. (1), such that the binary solid-fluid model is replaced by a porous material with variable permeability within the flow domain. The material interpolation function χ is then defined as

$$\chi(\gamma) = \frac{\gamma}{1 + q(1 - \gamma)}, \quad (5)$$

where $\gamma \in [0, 1]$ is the material parameter, and q is the interpolation parameter which controls the interpolation curvature. Now Eq. (3) becomes

$$\mathbf{x} \in \begin{cases} \text{Solid material} & \text{if } \gamma = 1 \\ \text{Fluid material} & \text{if } \gamma = 0 \\ \text{Porous (permeable) material} & \text{if } 0 < \gamma < 1, \end{cases} \quad (6)$$

such that the material zones are exclusively controlled by γ . Therefore, a γ field is assigned to all cells, and the vector of γ is adopted as the design variables for optimization.

As shown in Fig. 2, γ 's are defined on cell centers, where the momentum equations are solved and the penalization is applied. It should be noted that velocities at faces (ϕ fluxes) are not directly penalized, as they are intended to satisfy mass continuity, therefore, the solid-fluid boundary crosses cell centers (not faces).

A proper interpolation curvature helps to better control the behavior of gray designs ($0 < \gamma < 1$) during optimization, and also to eliminate them at local minima. Figure 3 illustrates the interpolation function of 5 with $q = \{0.1, 1, 10\}$ and $\epsilon = 10^{-7}$. As the curve intersections suggest, a certain impermeability level is achieved at larger γ values when q is larger. Therefore,

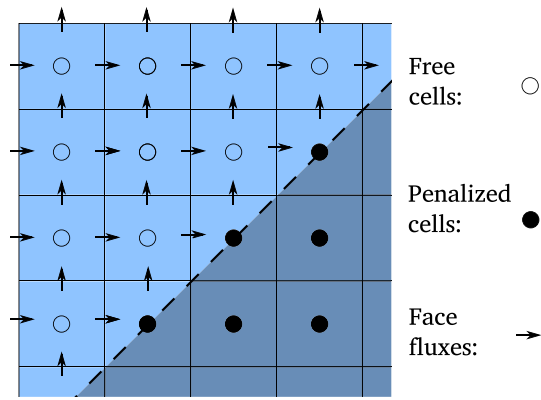


Fig. 2 Solid-fluid modeling using the penalization field. The dashed line, crossing cell centers, indicates the solid-fluid interface

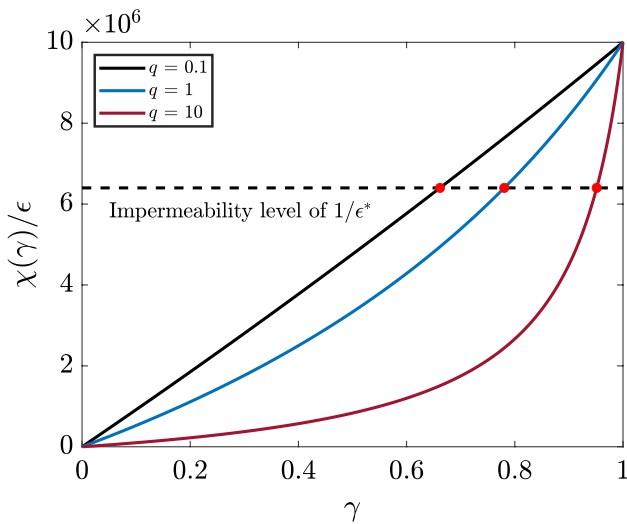


Fig. 3 Material interpolation function for various curvature parameters q

for a given problem, this effect forces the optimizer to push γ to 1 anywhere a solid zone is needed, or otherwise eliminate it by setting γ to 0, if a volume constraint is imposed. Accordingly, different q values are used for different optimization cases, in the present work.

3.2 Optimization formulation

A generic density-based topology optimization problem is formulated here as

$$\begin{cases} \text{minimize}_{\gamma} & J(\mathbf{U}, \gamma) \\ \text{subject to} & C_j(\mathbf{U}, \gamma) \leq 0 \\ & \mathbf{R}(\mathbf{U}, \gamma) = 0 \\ & 0 \leq \gamma_i \leq 1 \quad \forall \gamma_i \in \Omega_d, \end{cases} \quad (7)$$

where, J and C are the objective and constraint functions;

$$\mathbf{U} = [u_x \ u_y \ u_z \ p \ \varphi]^T \quad (8)$$

and

$$\mathbf{R} = [r^{u_x} \ r^{u_y} \ r^{u_z} \ r^p \ r^\varphi]^T \quad (9)$$

are the vectors of state variables and residuals, respectively; as mentioned earlier, γ is the vector of design variables, and $\Omega_d \subseteq \Omega$ is the design space, which can be defined as a part of the physical domain (Ω).

3.3 Sensitivity analysis

A gradient-based optimization is employed, and the accurate gradients (sensitivities) are obtained via a discrete adjoint method. Adjoint methods are effective techniques for computing sensitivity of a given function, which depends on both state and design variables. For sensitivity of function J , a Lagrangian function is first defined as

$$\mathcal{L} = J + \underbrace{\lambda^T \mathbf{R}}_{\approx 0}, \quad (10)$$

where, λ is the vector of adjoint variables; and by differentiating it with respect to design variable γ it becomes

$$\frac{d\mathcal{L}}{d\gamma} = \frac{\partial J}{\partial \gamma} + \frac{\partial J}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\gamma} + \lambda^T \left(\frac{\partial \mathbf{R}}{\partial \gamma} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\gamma} \right). \quad (11)$$

By rearranging (11), it becomes

$$\frac{d\mathcal{L}}{d\gamma} = \frac{\partial J}{\partial \gamma} + \lambda^T \frac{\partial \mathbf{R}}{\partial \gamma} + \left(\frac{\partial J}{\partial \mathbf{U}} + \lambda^T \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right) \frac{d\mathbf{U}}{d\gamma}. \quad (12)$$

To avoid computing $\frac{d\mathbf{U}}{d\gamma}$, which is a matrix of N_s (number of state variables) by N_d (number of design variables), alternatively, the adjoint problem of

$$\frac{\partial \mathbf{R}^T}{\partial \mathbf{U}} \lambda = -\frac{\partial J^T}{\partial \mathbf{U}} \quad (13)$$

is solved, and then the total derivatives are obtained by substituting λ in (12).

Solving the linear system of (13) for N_s adjoint unknowns has favorably a computational load, which is independent of the number of design variables. But, in order to solve the adjoint problem (and compute sensitivities) four partial

derivative matrices, namely $\frac{\partial J}{\partial \gamma}$, $\frac{\partial J}{\partial \mathbf{U}}$, $\frac{\partial \mathbf{R}}{\partial \gamma}$ and $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ are needed, where the first two depend on the given function and the latter two on the primal solver residuals. For this task, the discretized residuals are differentiated fully analytically. This approach provides solver-consistent and exact derivatives, since the governing equations are differentiated after the discretization. In addition, computing such derivatives would be significantly efficient, once their explicit close-forms are available. However, differentiating a sophisticated CFD-code such as OpenFOAM, analytically, is a non-trivial task, since it is not always easy to obtain the residuals in a closed-form.

To mitigate this problem, the primal solver is reformulated using several PDE and numerical operators, such as divergence, Laplace, gradient and interpolation operators (equivalent to the built-in OpenFOAM ones), in order to obtain residuals, represented in algebraic expressions. This approach untangles the dependency of the residuals on the state or design variables, without explicitly providing their mathematical expressions. The differentiation would accordingly be considerably simplified by applying the chain-rule of differentiation, where it is needed.

The details of the developed operator-based analytical differentiation (OAD) process is discussed in Appendix B. This differentiation process efficiently provides the exact partial derivatives of \mathbf{R} with respect to state variables, \mathbf{U} ,

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} = \begin{bmatrix} \mathbf{r}_{u_x}^{u_x} & 0 & 0 & \mathbf{r}_p^{u_x} & \mathbf{r}_\varphi^{u_x} \\ 0 & \mathbf{r}_{u_y}^{u_y} & 0 & \mathbf{r}_p^{u_y} & \mathbf{r}_\varphi^{u_y} \\ 0 & 0 & \mathbf{r}_{u_z}^{u_z} & \mathbf{r}_p^{u_z} & \mathbf{r}_\varphi^{u_z} \\ \mathbf{r}_{u_x}^p & \mathbf{r}_{u_y}^p & \mathbf{r}_{u_z}^p & \mathbf{r}_p^p & \mathbf{r}_\varphi^p \\ \mathbf{r}_{u_x}^\varphi & \mathbf{r}_{u_y}^\varphi & \mathbf{r}_{u_z}^\varphi & \mathbf{r}_p^\varphi & \mathbf{r}_\varphi^\varphi \end{bmatrix}_{(4N_C+N_F)^2}, \tag{14}$$

and with respect to design variables, γ ,

$$\frac{\partial \mathbf{R}}{\partial \gamma} = \begin{bmatrix} \mathbf{r}_\gamma^{u_x} \\ \mathbf{r}_\gamma^{u_y} \\ \mathbf{r}_\gamma^{u_z} \\ \mathbf{r}_\gamma^p \\ \mathbf{r}_\gamma^\varphi \end{bmatrix}_{(4N_C+N_F) \times N_D}, \tag{15}$$

for the adjoint problem of Eq. (10)–(13). Here, N_C , N_F , and N_D denote to the total number of cells, faces (with variable fluxes), and design variables, respectively.

3.4 Implementation notes

Figure 4 illustrates the flowchart of the present topology optimization framework. The method of moving asymptotes (MMA) (Svanberg 1995) is used as the gradient-based optimizer, which is particularly developed for problems with large number of design variables. As the flowchart

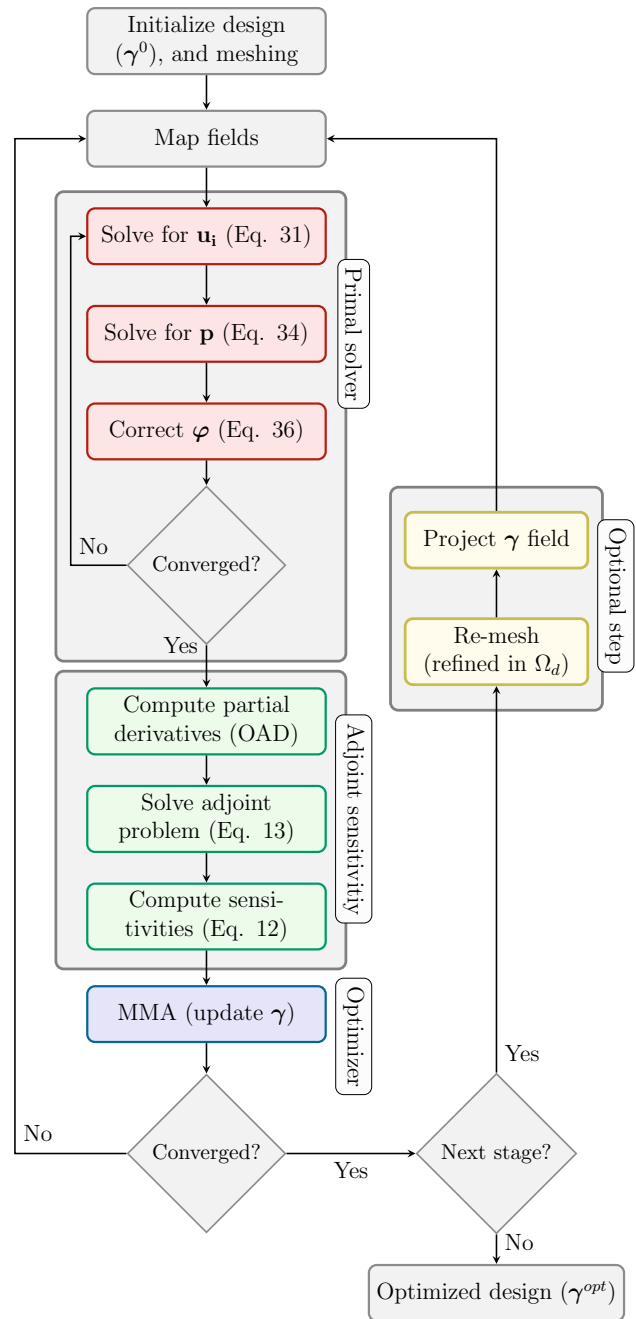


Fig. 4 Flowchart of the multi-stage discrete adjoint topology optimization

indicates, the optimization process has two main modules: the primal solver (OpenFOAM) and sensitivity computation (via discrete adjoint) parts. Once the flow solver (SIMPLE algorithm) converges, the required partial derivatives for the objective and constraints are computed via OAD, and the adjoint problem is solved to obtain the sensitivities for the optimizer. For the sparse matrix operations, which are required in OAD, the highly efficient and parallel Armadillo C++ library (Sanderson and Curtin 2016, 2018) is used,

which is a user-friendly and open-source tool for linear algebra problems, with a syntaxes similar to MATLAB. For code compilations, the `g++` compiler with `-O3` optimization option is used. For the solution of discrete adjoint problem, two methods are used, namely: a direct linear solver (LU factorization) for relatively small-sized problems (e.g. 2D cases), or the flexible GMRES iterative method with block Jacobi preconditioning and SOR sub-preconditioning for larger problems (e.g. 3D cases). For the first option the built-in MATLAB solver is used, but for the second one, the efficient and parallel PETSc library is used. As a result, all pieces of the present tool are efficiently parallelized (OpenFOAM, Armadillo, and PETSc). In this work, mainly the Phoenix computing resources (from TU Braunschweig) are utilized.

As mentioned earlier, a relatively higher mesh resolution is needed for better accuracy of the solid-fluid interface on Cartesian meshes. As a result, it is possible to perform the optimization problem on a fine mesh from the beginning; however, this can sometimes be noticeably expensive, particularly for 3D problems. Alternatively, a multi-stage procedure can be used. In this manner, the optimization is initially performed on a coarser mesh, and then the converged solutions are mapped onto the fine mesh, to be used as an initial starting point for a new optimization process on the fine mesh. The multi-stage process has two important advantages. Firstly, the overall optimization performance can be improved, as the secondary stages tends to converge faster with a good initial guess. Secondly, the new design space, which will be locally refined, can be adjusted according to the dimensions of the coarser design (note that in aerodynamic problems, the simulation space is considerably larger than the design, and a proper design space cannot always be easily estimated in advance). Consequently, unnecessary declaration of extra state and design variables will be avoided by limiting the mesh refinement to the necessary zones, which again leads to a reduced computational costs.

The multi-stage procedure can be used for both internal and external flow problems, but for external flow problems it is more beneficial, particularly if the design space size is reduced. Therefore, before re-meshing, the block of design space is adjusted with nearly extra 5% margin to the coarser design. Then, for the next stage a new mesh is generated with locally refined zone, coinciding the new design space. For re-meshing, a mesh grading strategy with gradually expanding cells (towards the boundaries) is used, although the mesh remains uniform within the refined (design) zones. For projecting the design (penalization) field onto the refined mesh, the OpenFOAM built-in `mapFields` tool with linear interpolation is used. Lastly, it should be highlighted again that as the flowchart of Fig. 4 indicated, the multi-stage procedure is optional, and its utility depends on the problem type (refer to Sect. 4 for examples).

4 Optimization results

In this section, first the accuracy of the BVP method is analyzed as it is the basis for density-based TO, and the derived sensitivities are examined in terms of accuracy and performance. Next, a series of 2D and 3D flow problems are studied to demonstrate the utilities of the present TO framework for a broad range of application. An emphasis is placed on drag and lift optimizations of external flow problems, to investigate the potentials of density-based TO for the design of flying objects.

4.1 Preliminary case studies

4.1.1 Accuracy of BVP method and mesh study

Introduced by Borrvall and Petersson (2003), penalization technique is the basis of the density-based flow TO approach, and is fundamentally different from level-set methods, in which the solid-fluid boundaries are clearly captured during the optimization process. Indeed application of Dirichlet (no-slip) boundary conditions on solid surfaces with a body-fitted mesh is more accurate than a solid structure model represented by nearly impermeable porous media on a non-matching mesh. But, a model analysis is essentially needed to quantify such inaccuracies. Therefore, a set of preliminary cases are studied for this purpose.

Firstly, flow around a cylinder is simulated with three settings: (a) body-fitted mesh with classical boundary conditions, (b) body-fitted mesh with overlapping penalized zone, and (c) non-overlapping penalization zone on Cartesian meshes (Fig. 1). Case (a) is considered as the reference solution here. Case (b) with various penalization parameters ($\epsilon = \{10^{-5}, 10^{-6}, 10^{-7}\}$) is intended to indicate BVP model

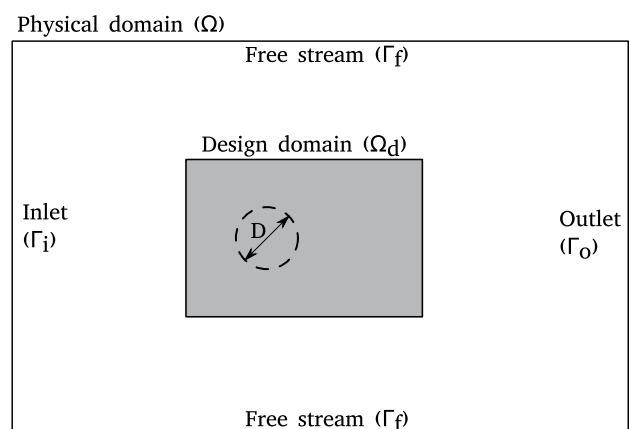


Fig. 5 Schematic of the 2D external flow problem with specified boundary conditions

accuracy, as the internal mesh (penalization zone) matches the cylinder surface. In case (c), penalization is applied on cells, of which centers laid inside the cylinder. This case is refined two times to examine accuracy achievements with higher mesh resolutions.

Figure 5 illustrates schematic of a typical 2D external flow optimization problem, with physical and design domains and boundary conditions. Here, a fixed circular design, which is a challenging geometry for Cartesian mesh, is considered; but later, optimized structures will be designed during a optimization process, inside Ω_d . The drag force is studied as a measure of accuracy, which will be later used as an objective or a constraint for optimization. Exerted fluid forces on solid structures are conveniently computed by volume integration of the Brinkman penalization term (in Eq. 1) over a given domain (see Kondoh et al. (2012)) via

$$\mathbf{F} = \frac{1}{\epsilon} \int_{\Omega_d} \chi(\boldsymbol{\gamma}) \mathbf{u} \, d\Omega. \tag{16}$$

The Table 1 lists the computed drag coefficients, $C_D = \frac{2F_x}{\rho U_\infty^2 A}$, for flow at $Re = \frac{U_\infty D}{\nu} = 36$, where A and D are the frontal cross-section area and the cylinder diameter, respectively. The flow domain is 12-by-5, where a $D = 1$. Cases (a) and (b) are meshed with sufficient resolutions (301k and 389k cells, respectively), with higher resolutions near cylinder surface. For case (c), the domain is discretized uniformly, and refined two times such that mesh resolutions are $\Delta x = \{D/64, D/128, D/256\}$.

The overlapping penalized zones indicate negligible errors (less than 0.1%), which is slightly improved by reducing ϵ ; however, very small ϵ slows down the convergence rate, due to numerical stability issues. Hence, $\epsilon = 10^{-6}$ is used for a compromise between accuracy and performance. The results of the Cartesian meshes are less accurate, due to the combination of penalization and discretization errors. It is because capturing a curved geometry on a Cartesian mesh leads to non-matching solid interfaces (Fig. 1c). It can be

Table 1 Comparison of drag force for 2D cylinder using BVP method on body-fitted and Cartesian meshes

Case	C_D	Rel. error
Body-fitted (no-slip)	1.65420	
Body-fitted (BVP, $\epsilon = 10^{-5}$)	1.65311	-0.066%
Body-fitted (BVP, $\epsilon = 10^{-6}$)	1.65327	-0.056%
Body-fitted (BVP, $\epsilon = 10^{-7}$)	1.65332	-0.052%
Cartesian, coarse (BVP, $\epsilon = 10^{-6}$)	1.64776	-0.389%
Cartesian, medium (BVP, $\epsilon = 10^{-6}$)	1.64964	-0.275%
Cartesian, fine (BVP, $\epsilon = 10^{-6}$)	1.65079	-0.213%

conceived that the BVP modeling error is less than the discretization error, although is can be considerably improved by mesh refinement (Fig. 1d). As a result, it should be better analyzed how much mesh refinement can improve the accuracy.

Accordingly, a symmetric convex design (with aspect ratio of $L_y/L_x = 1/8$, and $A = 1$) is studied for various mesh resolutions at three different Reynolds numbers, $Re = \{50, 200, 500\}$ (or $Re_a = \{150, 600, 1500\}$ by adopting cord as characteristic length). A series of Cartesian meshes with resolutions between 50 to 400 cells per unit characteristic length (D) are considered, and selectively shown in Fig. 6. The computed drag forces are shown in Fig. 7, for all cases. It is observable that at higher Reynolds number, a finer mesh is needed for a better accuracy. By assuming the boundary layer size be proportional to $\delta^* = L_x/\sqrt{Re_x}$ and performing analyzing the computed forces, it can be estimated that 15 cells per δ^* length would approximately provide error of less than $\sim 1\%$, which for instance is the case for mesh of Fig. 6c at $Re = 500$. Similarly, minimum 12 or 7 cells per δ^* for errors less than $\sim 2\%$ and $\sim 4\%$, respectively. Nevertheless, such errors could simply be ignored for some applications or preliminary (conceptual) designs, where a high accuracy level is not a primary concern.

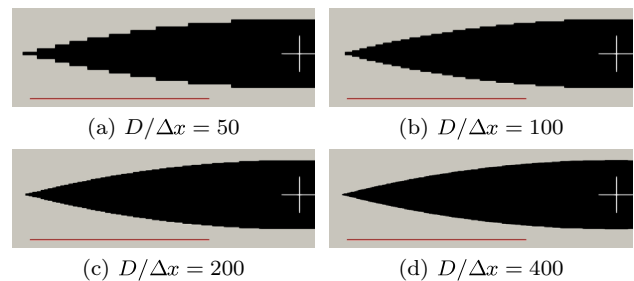


Fig. 6 Symmetric convex design on various Cartesian meshes (note that figures are cropped)

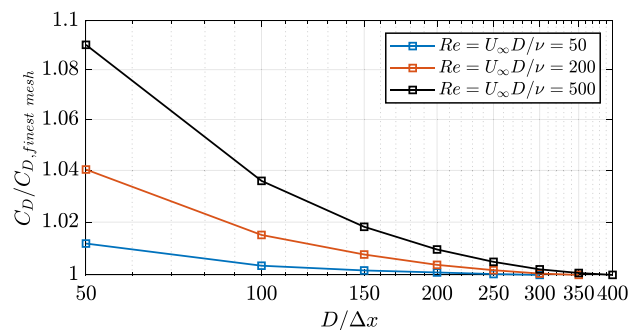


Fig. 7 Drag and mesh convergence study for the symmetric convex shape, at various Reynolds numbers

4.1.2 Sensitivity verification

Discrete adjoint with analytically differentiated partial derivatives provides exact sensitivities. Nevertheless, it is crucial to verify the developed adjoint problem and particularly its implementation, prior to performing optimizations. For this task, the sensitivity of the problem in Sect. 4.1.2, as an example, is computed via analytical discrete adjoint, and compared with the sensitivities derived numerically by finite-difference method (FDM). Hence, drag force partial derivatives, as required in Eq. 12–13 ($J = F_x$), are simply obtained from Eq. 16 by

$$\frac{\partial J}{\partial u_{x,i}} = \frac{1}{\epsilon} \chi_i v_i, \tag{17}$$

and

$$\frac{\partial J}{\partial \gamma_i} = \frac{1}{\epsilon} u_{x,i} v_i \frac{\partial \chi_i}{\partial \gamma_i}, \tag{18}$$

where v and i are cell volume and index, respectively. The numerically approximated sensitivities are obtained using central-differencing scheme via

$$\frac{dJ}{d\gamma_i}^{FDM} = \frac{J(\gamma_i + \delta) - J(\gamma_i - \delta)}{2\delta}, \tag{19}$$

and then, the relative errors are computed via

$$E_i = \frac{dJ/d\gamma_i^{FDM} - dJ/d\gamma_i^{Analytic}}{dJ/d\gamma_i^{FDM}}. \tag{20}$$

Figure 8 indicates the relative errors for all design variables of a coarse meshed domain. FDM is perturbed with various δ , such that the acceptable range between round-off and truncation error of FDM is captured. The matching results

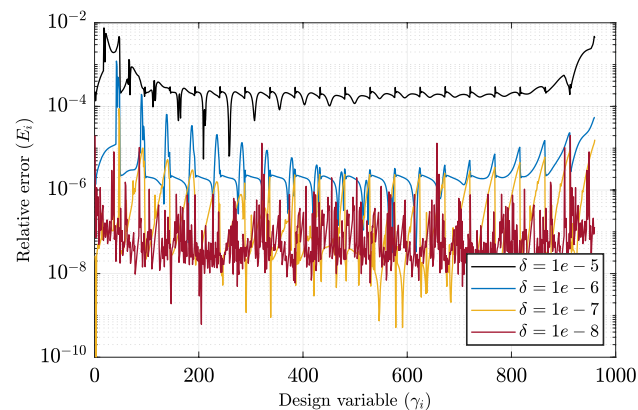


Fig. 8 Sensitivity verification using central-scheme finite-difference method with various perturbation sizes

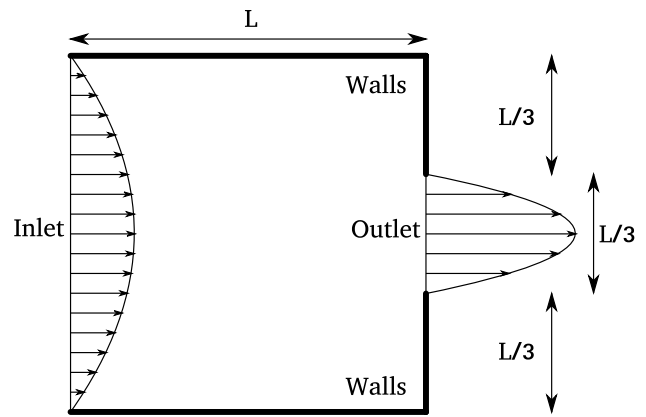


Fig. 9 Schematic of the diffuser TO problem

($E < 10^{-6}$) satisfyingly confirm the accuracy as well as the correct implementation of the analytical adjoint. Same results can be obtained for other functions such as lift or pressure drop.

4.1.3 Diffuser problem (2D) and performance analysis

First example is the classical diffuser topology optimization problem, first investigated by Borrvall and Petersson (2003).

Figure 9 shows the geometry as well as the boundary conditions imposed for this internal flow problem. A parabolic velocity profile with $u_x^{max} = 10$ is applied at inlet, $L = 3$, and $Re = u_x^{max} L / \nu$ is either 10, 100 or 1000. In this case the design and physical domains are identical $\Omega_d = \Omega$. The objective is minimization of the total pressure drop, hence

$$J = \int_{\Gamma_{in}} \left(p + \frac{1}{2} \rho \mathbf{u}^2 \right) d\Gamma - \int_{\Gamma_{out}} \left(p + \frac{1}{2} \rho \mathbf{u}^2 \right) d\Gamma, \tag{21}$$

with minimum 25% volume constraint. Since velocity and pressure values are fixed in inlet and outlet, respectively, the derivatives of 21 become

$$\frac{\partial J}{\partial p_i} = |\mathbf{S}_f^i| \quad (\text{for } f \in \Gamma_{in}), \tag{22}$$

$$\frac{\partial J}{\partial u_{j,i}} = -\rho |\mathbf{S}_f^i| u_{j,i} \quad (\text{for } f \in \Gamma_{out}, j = \{x, y\}), \tag{23}$$

where i is the cell index, $|\mathbf{S}_f|$ is the cell face area, and $\partial J / \partial \gamma_i = 0$. It is important to note that J requires pressure and velocity at cell faces, but in Eq. 22–23 cell center values are used. It is simply because at such boundaries, face values equal their cell center values, due to the applied zero normal gradient boundary conditions.

Figure 10 shows the optimized topologies (started from empty designs) on a uniform mesh of 480^2 size. The total

Fig. 10 Results of diffuser topology optimization with 480^2 design variables (cells)

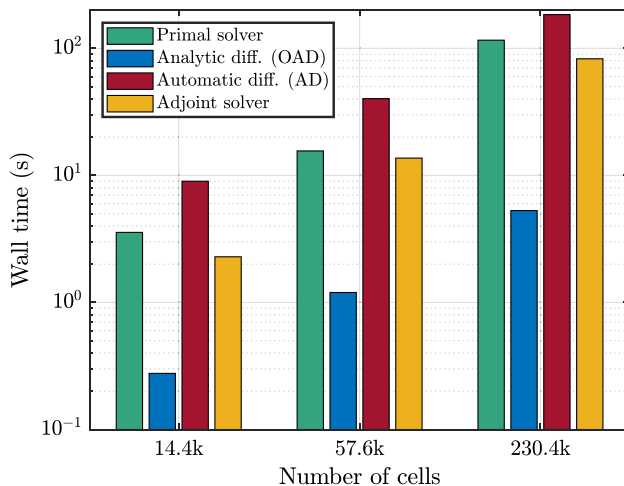
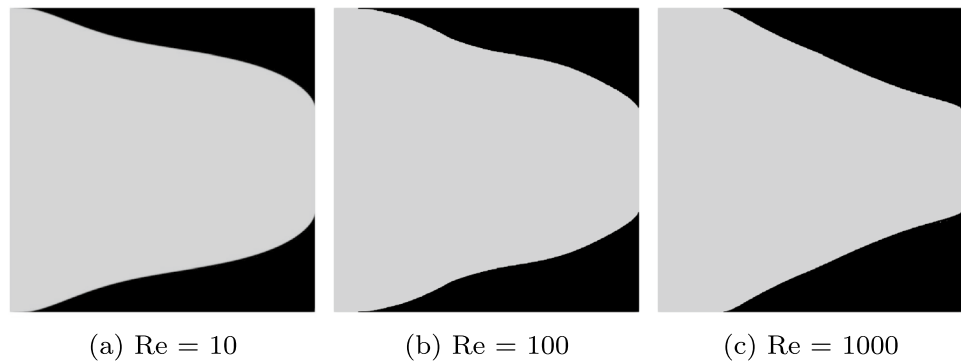


Fig. 11 CPU-time details of diffuser problem with primal solver, analytical differentiation (OAD), reverse-mode AD (DAFoam), and adjoint solver parts, for three resolutions 120^2 , 240^2 , 480^2 , computed on a 14-core processor (Intel® Xeon® E5-2697 v3). DAFoam (v2.2.3) is used for as AD tool and matrix coloring is excluded for timing

pressure loss is changed by + 13.2%, - 5.2% and - 16.6%, respectively for $Re = 10, 100, 1000$. For $Re = 10$, the objective value has been increased to fulfill the volume constraint. Since the empty design is an infeasible state, the optimizer tends to rapidly expand the volume, which leads to a jump of objective value. Hence, the interpolation parameter of $q = 5$ was used for a more gradual increase of objective function value and an oscillation-free process. However, for $Re = 100$ and 1000 cases, the objective function reduces by volume increase (without oscillation). As a result, $q = 0.1$ was used, which leads to almost a linear curve. Nonetheless, it should be highlighted that the results are barely sensitive to the interpolation parameter, and $q = 1$ is found to be working properly for most of the cases.

It is worth noting to the computational performance of the present topology optimization example, particularly the analytical differentiation part. Figure 11 indicates a breakdown CPU-time for three resolutions of 120^2 , 240^2 and 480^2 cells, performed on a 14-core processor. Noticeably, less than 11%

of total adjoint sensitivity parts for these cases are dedicated to differentiation by OAD, whereas it is more than 67% for the reverse-mode AD, although both approaches provide exact derivatives. This can advantageously reduce the total computational time, particularly for large 3D problems. Also, it is important to note that the exact derivatives can be computed on-the-fly by OAD. More precisely, OAD doesn't need any information (e.g. cell connectivities or mesh topology) to be previously stored. Therefore, by changing mesh after multiple refinements, no extra process is required for the new mesh.

4.2 Drag minimized (2D) topologies

Aim of this example is finding optimum shape of structures with minimized drag forces, under a minimum volume (cross-section area) constraint. The problem setting is similar to the one studied in Sect. 4.1.1, and Fig. 5 indicates its geometry and boundary conditions. Area of a circular cross-section with unit diameter $D = 1$ is considered as the lower bound for the design area, i.e. $A_{\min} = (D/2)^2\pi$. Size of the design domain is set to $\Omega_d = 6D \times D$, which properly overlaps the final geometries. The reference Reynolds number is defined as $Re_r = U_\infty D/\nu$, nevertheless, the actual values could be calculated by $Re_a = (L_c/D)Re_r$, where L_c is the characteristic length of the optimized shapes. A square with area of $\sim A_{\min}/100$ is adopted as the initial design γ^{init} . It is because the solver doesn't converge with fully empty design space. The interpolation parameter is set to $q = 1$ for a smooth (oscillation-free) optimization process. A sufficiently fine mesh (545.1k cells, with $\Delta x = D/200$) is used here to provide 15 cells per δ^* length, for an accuracy level of approximately less than 1% of drag for flows up to $Re_r \leq 500$ (and less than $\sim 3\%$ for $Re_r = 1000$). A single-stage optimization process is considered here, because the problem is relatively inexpensive (compared to 3D cases).

Figure 12 illustrates the optimized shapes (topologies) for Re_r numbers ranging from 10 to 1000. The optimized design of $Re_r = 10$ has a blunt convex-symmetric shape, similar to the introduced rugby-ball shapes by Borrvall and Petersson

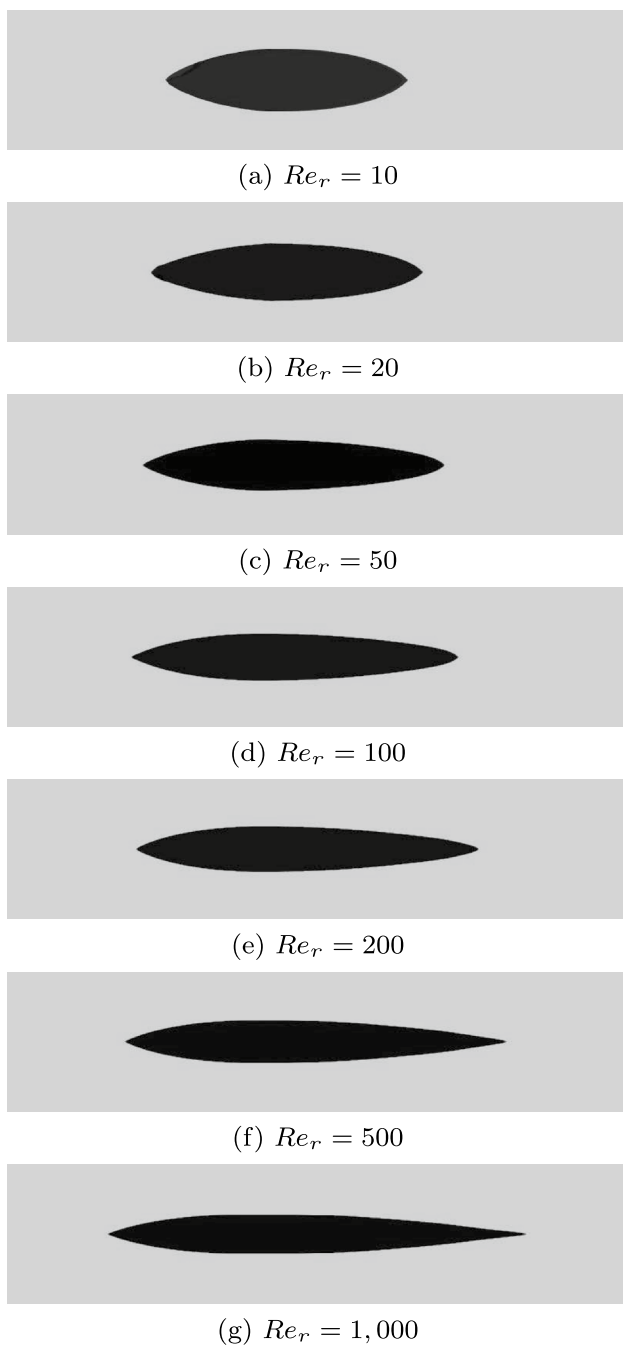


Fig. 12 Drag minimized (2D) topology optimization results under volume constraint

(2003). By increasing Reynolds number, aspect ratio of optimized designs increases. Also, leading and trailing edges become gradually sharper, as pressure drag dominates viscous drag for higher Reynolds numbers. The symmetrical profile disappears at larger Re numbers to suppress flow separation and formation of a wake region, which increase the drag, as it is the case for case $Re_r = 1000$ for instance. The optimized drag coefficients are plotted in Fig. 13 with

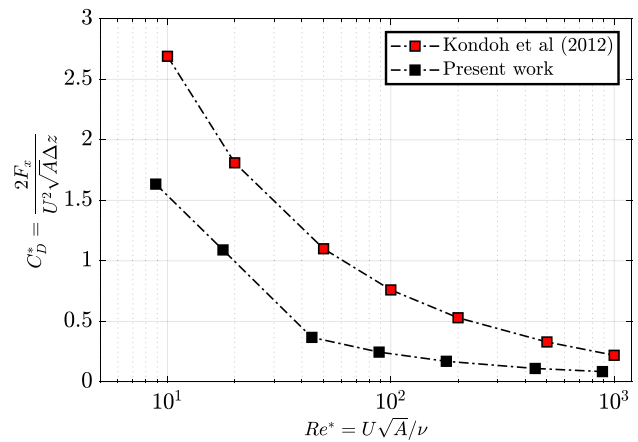


Fig. 13 Drag coefficients of the 2D volume-constrained topology optimization problem for various Reynolds numbers. The reference length is changed here to \sqrt{A} in accordance to the literature

respect to Reynolds numbers, and compared to the similar work of Kondoh et al. (2012) (note that reference length is replaced by $\sqrt{A_{min}}$ for similar definition of Re and C_D), which tend to produce comparatively less drag forces.

Before proceeding, it is practical to examine the effect of initial guess on the final design. For this purpose, the optimization case of $Re_r = 200$ is solved again with different initial geometries. Figure 14 shows the initial designs as well as the final solutions. As Fig. 15 indicates, all cases converged to local minima, although with different convergence speed, and final drag values. All cases, no matter the number of initial elements, converged to an airfoil-shaped geometry (identical in terms of topology). Nevertheless, it is observable that the initial guess can influence the final solution, at least in terms of convergence behavior. In addition, starting from a single-element design can be a suitable initial guess to start the optimization with, and is adopted as the initial guess in the following examples as well.

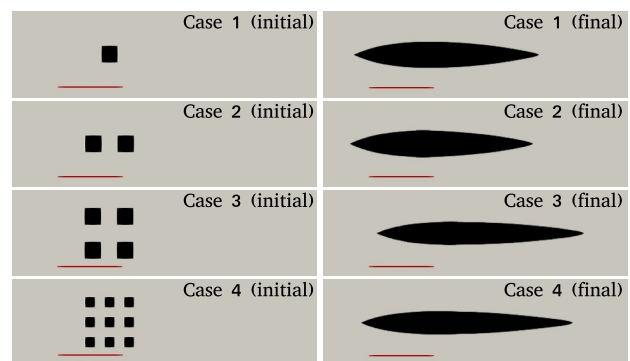


Fig. 14 Initial and optimized designs for $Re_r = 200$

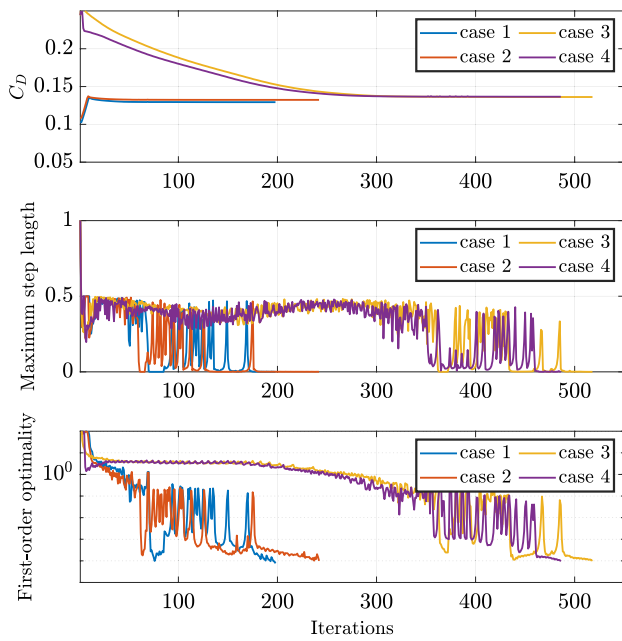


Fig. 15 Influence of initial guess on final solutions

4.3 Lift maximized (2D) topologies, with constrained drag

In this example, the goal is to investigate the topologies, which maximize lift force with constrained drag force. This problem formulation can be considered for some applications, in which the goal is to maximize payload for a fixed thrust force, for instance. A set of problems with different flow and optimization settings are considered to better study the utility of the present approach. In all cases, drag force is constrained such that $C_D/C_D^0 \leq \alpha$, where C_D^0 is the minimized drag coefficient obtained from Sect. 4.2. Each case is defined with a unique combination of three different

aspects, namely: Reynolds number $Re_r = \{50, 200, 500\}$, permissible drag force $\alpha = \{1, 1.4, 1.8\}$, and enabled or disabled minimum volume constraint (18 cases in total). Due to the flow physics of the lift problem, the simulation domain is expanded to $\Omega = 12D \times 12D$, which embodies a design domain of $\Omega_d = 7D \times 2D$. In addition to the enlarged physical domain, the adjoint problem should be solved twice her (for both lift and drag sensitivities), which increases the computational load compared to the drag minimization case. Therefore, a two-stage optimization process is employed to reduce the overall cost of these 18 optimization cases. Similar to the previous example, optimizations cases are initiated with a small square with area of $A = 0.01D^2$.

Figures 16, 17 and 18 illustrate the optimized topologies for $\alpha = 1, 1.4$ and 1.8 , respectively. The design of 16a is simply a curved thin structure, which attempts to guide the upstream flow to downward direction in order to generate lift force. It is predictable that for higher Re_r (or U_∞) the flow stream above the upper design surface may separate and form an unsteady wake region behind, which may increase lost of momentum that are supposed to generate lift. It is worth noting that the optimized topologies avoid such phenomena interestingly in three ways: firstly, the angles of attack are reduced to minimize the area of the potential wake, and to avoid early flow detachments; secondly, two sharp corners are often created to control flow detachments at the upper and lower surfaces of the trailing edges; and thirdly, a limited number of isolated elements (trailing structures) are created in the wake region to guide and stabilize the flow. Therefore, most of the optimized topologies consist of more than one element.

The importance of such trailing structures can be investigated better by a closer look at the flow stream lines depicted in Fig. 19, corresponding to the optimized design of Fig. 18f. It can be seen that the upper-surface flow is partially guided through the training elements, which stabilizes the flow in

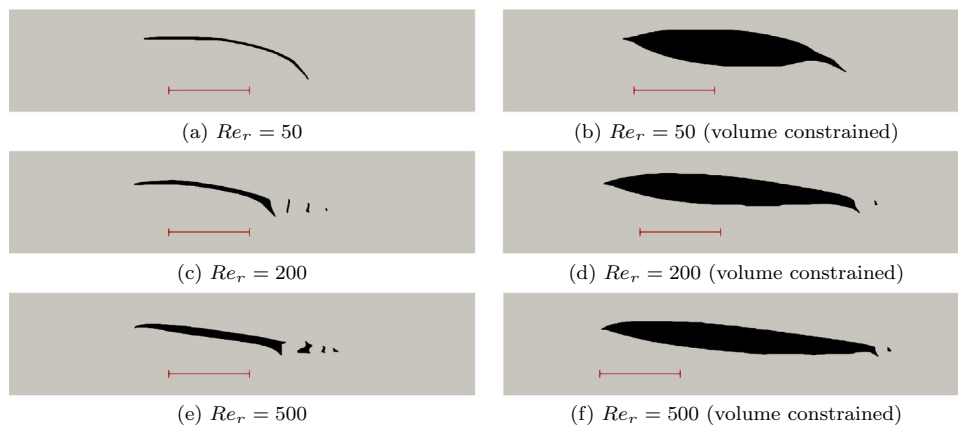


Fig. 16 Maximized lift with drag constraint of $C_D \leq C_D^0$

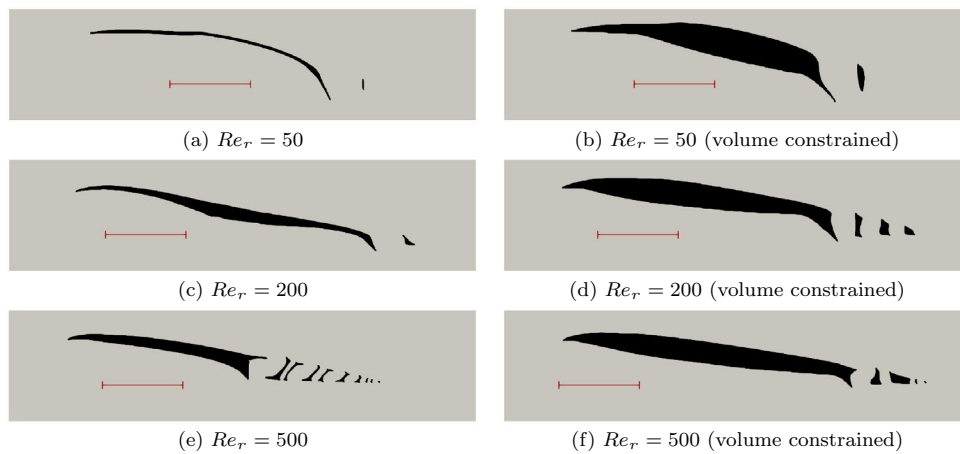


Fig. 17 Maximized lift with drag constraint of $C_D \leq 1.4C_D^0$

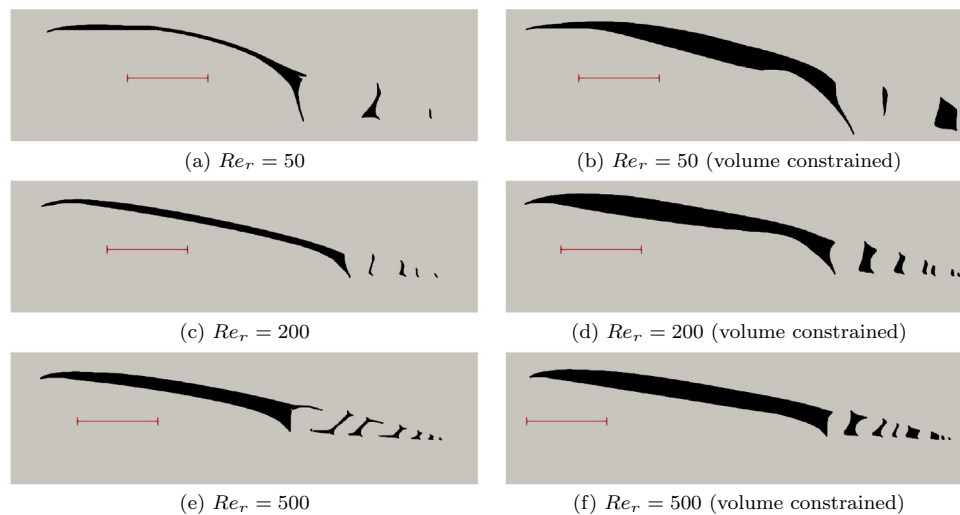


Fig. 18 Maximized lift with drag constraint of $C_D \leq 1.8C_D^0$

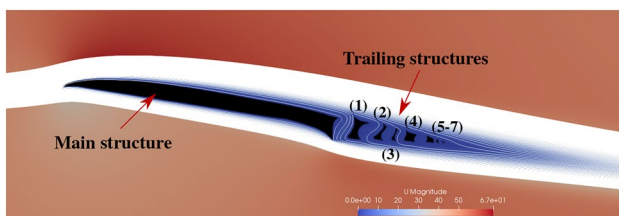


Fig. 19 Plot of streamlines for the case of Fig. 18f, to reveal the role of trailing elements in the wake

the wake. The post-simulation analysis of the same structure without the trailing elements indicates a largely unsteady flow in the wake, and confirms the stabilization role of such elements. A two-stage design process of this structure is plotted together with its optimization history in Fig. 20.

It is worth noting that the intermediate designs contain areas of gray (porous) materials in the wake, with damping flow velocity effects, which better clarifies the importance of wake stabilization during the optimization process. A detailed analysis of the exerted forces on trailing elements is listed in the Table 2. It is interesting that the elements produce negative drag and lift forces, which the latter is negligible in magnitude. Here the drag force is a limiting design constraint, and a negative drag force is advantageous, as it allows the main structure to reach to a larger drag. As a result, the trailing elements play two effective roles, wake stabilization and producing negative drag forces.

Figure 21 indicates the optimization results, C_L/C_D , for different Reynolds and α numbers, with and without volume constraints. It should be highlighted that the actual Reynolds number are much larger the the length of the

Fig. 20 Design process and optimization history of case of Fig. 18f

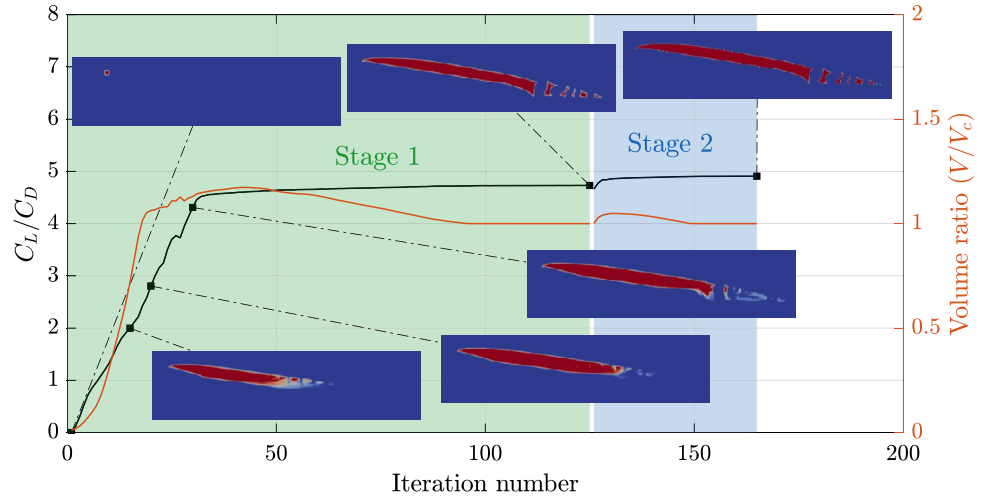


Table 2 Relative drag $C_{D,i}/C_D$ and lift $C_{L,i}/C_L$ forces of the trailing structures, for case of Fig. 18f

Trailing structures	1	2	3	4	5-7
Drag ratio (%)	-3.209	-0.814	-0.361	-0.319	-0.070
Lift ratio (%)	-0.066	-0.005	-0.015	-0.031	-0.029

main structure is adopted as the characteristic length. For example, $Re_a \approx 1900$ for the design in Fig. 18f. Relaxing the permissible drag force by increasing α , expectedly facilitates achieving to higher maximum lift forces in all Reynolds numbers. At higher flow velocities the achieved maximum C_L/C_D 's increases, due to the larger input momentum of the upstream. It can be also been that imposing volume constraint considerable limits the obtainable maximum lift, particularly for $\alpha = 1$ cases. Larger volumes lead to less aspect-ratios which generates more drag forces. However, for larger

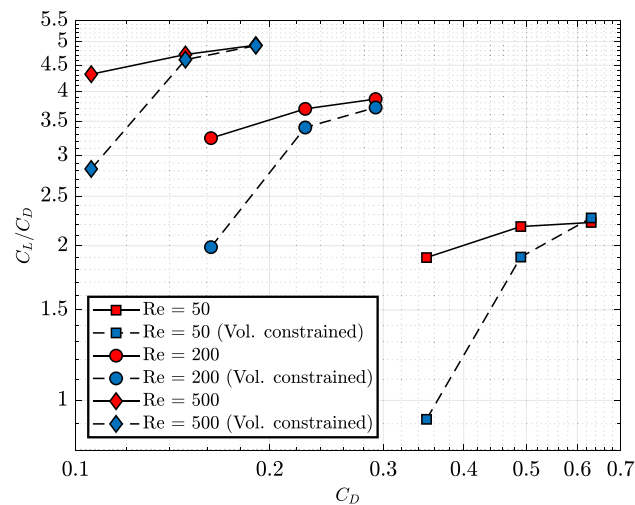


Fig. 21 Optimized C_L/C_D values versus C_D constraints

permissible drag forces, the effect of minimum volume constraint diminishes, as the design volume increases (compare Fig. 16–18 cases).

4.4 Drag minimization and lift maximization (3D)

In this part, two 3D examples are considered, namely: drag minimization with constrained (minimum) volume, and lift maximization with constrained (minimum) drag. Here, the volume of a sphere with unit diameter $D = 1$ is considered as the lower limit for design volume in drag minimization case, and similar to the previous 2D cases, the value of the optimized C_D (with a scale of α) is adopted as the maximum

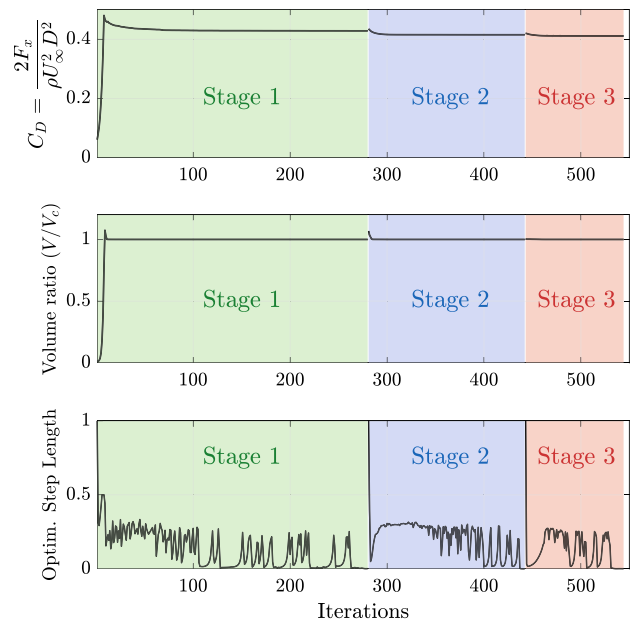
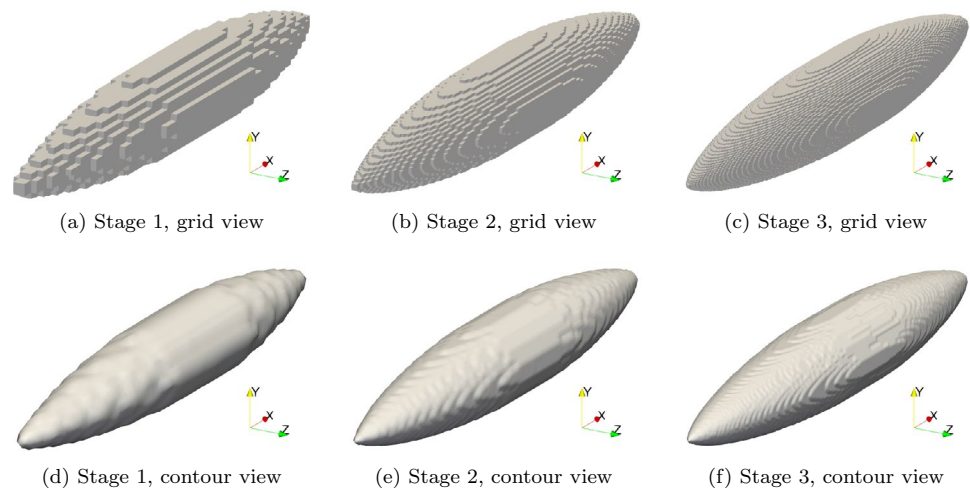


Fig. 22 Convergence histories for the three-stage drag minimization problem in Sect. 4.4

Fig. 23 Optimized solutions of the 3D drag minimization problem for three stages with $\Delta x = \{D/24, D/48, D/96\}$. Note that the graphs indicate the $\gamma = 0.5$ level-set of the 3D scalar design (γ) field



permissible drag force for the lift maximization case. The upstream velocity is $U_\infty = 20$, and the reference Reynolds number is $Re_r = U_\infty D/\nu = 200$. The topology optimization process for both cases are initiated with a relatively small cube (edge size of $D/5$) for a better convergence behavior, based on the previous findings in Sect. 4.1.1. Such an initial guess initially violates the volume constraint in the drag minimization case, however, it will demonstrate the capability of the present framework for designing almost from scratch. Accordingly, to fulfill the minimum volume, the drag force has to increase during optimization. Therefore, similar to Sect. 4.1.3 the interpolation parameter of $q = 1$ facilitates a smoother (non-oscillatory) design process.

For the drag minimization problem, a three-stage design process is utilized, with the simulation domain $\Omega = 12D \times 5D \times 5D$ and three mesh resolutions of $\Delta x = \{D/24, D/48, D/96\}$. The optimization process including the histories of C_D and design volume are shown in Fig. 22, and the final (optimized) designs in Fig. 23. It can be seen that initially the optimizer expands the design volume within the first few iterations to satisfy the initially

violated volume constraint. The process continues by minimizing the objective function with the permissible volume lower bound. Initially a conservatively large design space ($\Omega_d = 6D \times D \times D$) was used, as the final design sizes were initially unknown. Then, the location and size of the design spaces (for the second and third stages) were adjusted corresponding to the optimized topology in the first stage, to minimize the local refinement zones, as shown in Fig. 25a. The optimized drag forces are $C_D = \{0.4282, 0.4151, 0.4109\}$, respectively for each consecutive stage, with 3.1% and 1.0% improvements after each refinement. In addition, according to the mesh studies in Sect. 4.1.1, the mesh resolution in the third stage provides more than 12 cells per δ^* (at $Re_r = 200$, or $Re_a \approx 700$), which can suggest an overall error of $<2\%$ in calculation of C_D .

For the lift maximization case, a drag constraint of $C_D \leq \alpha C_D^0$ is imposed, where C_D^0 is the optimized drag coefficient from the previous case. Here, no maximum volume constraint is imposed, since the drag constraint can indirectly limit the design size. Also in this case, α is set to 8, since lower values, e.g. 1 or 2, considerably suppress the design

Fig. 24 3D lift maximization at $Re_r = 200$, with drag constraint

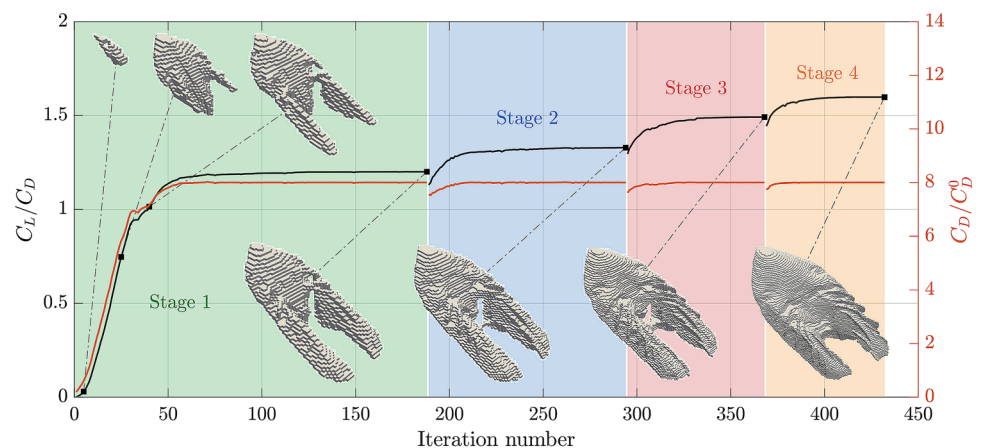
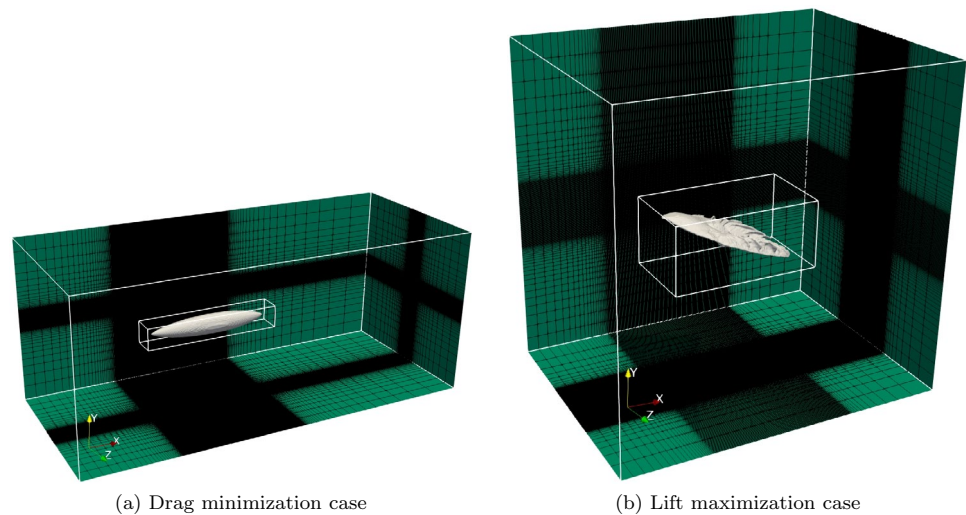


Fig. 25 Design and simulation domains for 3D drag and lift optimizations



volume to expand. Here, the initial guess does not violate the constraint, and therefore, the interpolation effect is disabled by $q = 0.1$ (almost linear curve). The simulation domain is expanded to larger domain ($\Omega = 12D \times 12D \times 9D$) compared to the drag case, to better capture the flow field. Both drag and lift values are relatively negligible initially for the small cube (initial guess), which lets the design to gradually expand its volume, in order to gain lift force, until the permissible drag is reached. For this problem a four-stage procedure is utilized, with mesh resolutions of $\Delta x = \{D/15, D/20, D/30, D/40\}$.

The optimization process is shown in Fig. 24. It can be observed that the design volume expands in the first ~ 50 iterations until the permissible drag is reached. It should also be pointed out that the optimization process, compared to the drag minimization case, requires more iterations to converge. This can be attributed to the absence of volume constraint, which sharply limited the design to change (in drag minimization cases). Figure 26 demonstrates the final design for the 4th stage, in multiple views, and Fig. 25b shows the design as well as the simulation and design domains. Considering the problem setting and the local (refined) mesh size, the final stage provides more than 7 cells per δ^* , which suggest an estimated error of $<4\%$ in computed forces. The final lift over drag ratio of $C_L/C_D = 1.59$ at $Re_r = 200$ was achieved at last stage. Unlike the drag case, the achieved maximum lift force increases at each stage, mainly because at higher mesh resolutions, a thinner design with less drag

force can be designed, which is in favor of the present drag constrained problem setting (to better manage the permissible drag force).

It is worth noting that, in contrast to the 2D lift cases, the optimized design is an one-element structure, apparently due to the different nature of the wake in 3D flow. It has a ~ 22 degree angle of attack to redirect the upstream flow downwards to increase the lift force, while maintaining the flow to be stabilized and attached in the wake region (behind the main body). Visualization of the flow streamlines in Fig. 27 can reveal the utility of the grooved surfaces for flow control (attachment) in the wake region. It is not trivial to eliminate the grooves to precisely analyze their functionality, however, an analogous behavior was previously observed in 2D optimized cases, where trailing structures suppressed wake instabilities. Design of such complex pieces can be a very challenging task for other techniques such as shape optimization, however, as it was stated earlier, density-based TO is capable of designing optimized structures from scratch.

5 Discussion

In this section, the achievements and novelties of the present work are discussed, in terms of application of density-based TO for laminar aerodynamics, and performance achievements by OAD and multi-stage framework.

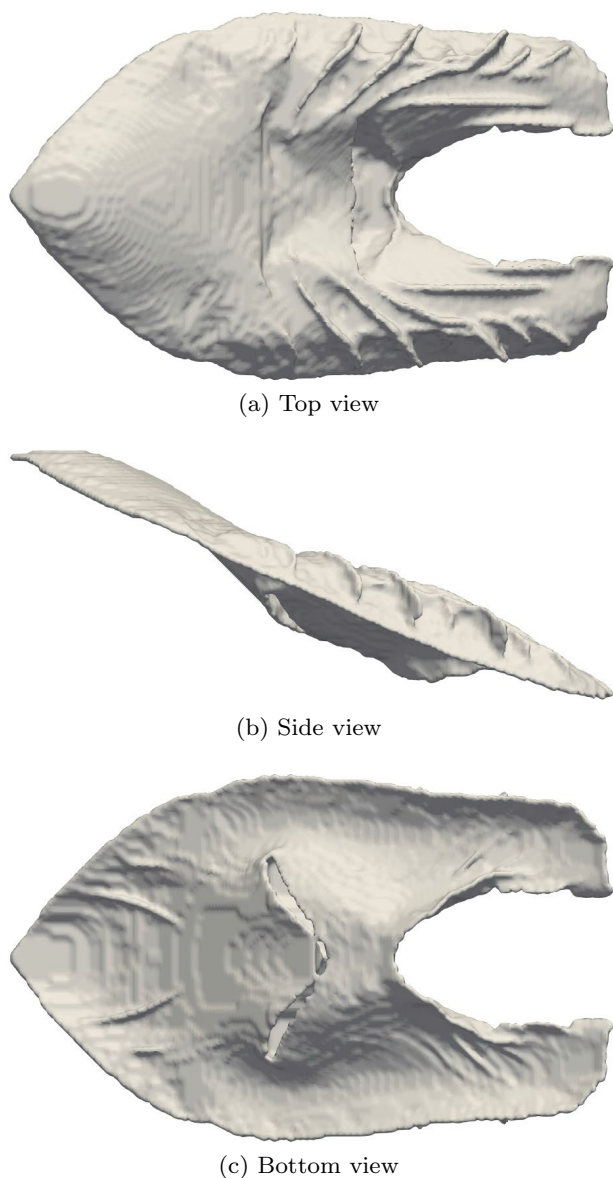


Fig. 26 3D lift maximized topology, with post-processed interface

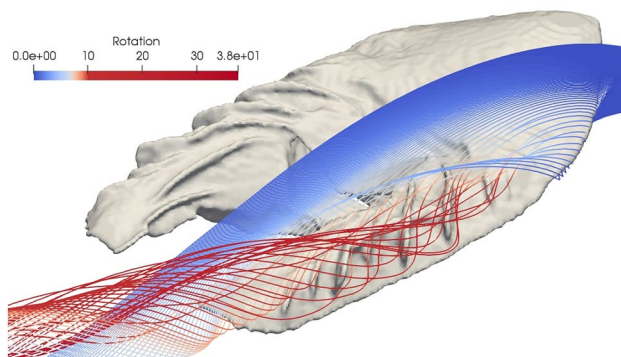


Fig. 27 Streamlines over the grooved surfaces

The optimization results demonstrated a promising utility of the density-based TO in terms of design of aerodynamic structures from scratch, for both 2D and 3D problems. In some cases, interestingly up to 8 distinct elements were designed from a very small square or cube as an initial design. This feature is a noticeable strength for density-based method, which is mostly lacked by classical level-set topology optimization approaches. In addition, the designed trailing structures or grooved surfaces demonstrated an interesting capability of the present TO framework, to maintain the flow stability during the entire optimization process, which is essentially important to minimize loss of energies and obtain high-performance flying devices. Such details are sometimes very unintuitive to be designed manually. Consequently, the present approach further underlines the designing capability of the density-based method for laminar aerodynamic applications.

Table 3 indicates the 3D problems sizes as well as their CPU-time breakdowns for each optimization stage of Sect. 4.4. It should be pointed out that the exact analytical differentiation via OAD, dedicates less than 3% of the total CPU times in these 3D cases, suggesting its high efficiency. In addition, the choice of linear solver (preconditioned flexible GMRES) for adjoint problem provides computing times comparable to the OpenFOAM (primal) solver (note that both solvers are initiated with the previous solution for a faster convergence speed). Also, the multi-stage procedure facilitated multiple local mesh refinements (withing the design space), to achieve a better model accuracy with less computational costs. Using the finest mesh for a single-stage optimization, besides its expensive computational costs, suggests less convergence speed, particularly when a large volume change occurs during the process (due to the slower volume growth with smaller cells). However, by assuming a similar convergence speed (independent of mesh size), it can be estimated that the multi-stage process can reduce the total CPU-time by at least 45%. Such a performance gain is very important, not only for sake of reduced computational time, but also for achieving higher BVP model accuracy by refined meshes.

6 Closure

An efficient and parallel multi-stage density-based topology optimization framework was presented for 2D and 3D laminar flow problems. The exact sensitivities were efficiently computed via OAD and discrete adjoint methods. For implementing OAD, Armadillo library was used, whereas OpenFOAM and PETSc library were used for flow and adjoint problems, respectively. A set of 2D and 3D problems were studied and results promisingly suggested the utility of the

Table 3 The average CPU-time per each optimization iteration for different stages of 3D problems in Sect. 4.4, performed on a 10-core Intel® Xeon® (E5-2640 v4) processor (equipped with 256 Gb RAM)

Stage	3D drag min. case			3D lift max. case				
	# Cells	Primal Sol. (min)	Adjoint Sol. (min)	OAD Diff. (min)	# Cells	Primal Sol. (min)	Adjoint Sol. (min)	OAD Diff. (min)
1	385,000	0.61	0.22	0.05	520,992	0.87	0.59	0.07
2	1,254,528	2.49	1.04	0.14	1,088,000	1.63	1.48	0.14
3	3,852,288	6.84	7.67	0.46	2,493,072	4.52	6.19	0.33
4	–	–	–	–	4,782,400	14.23	21.02	0.62

density-based for aerodynamic optimization and design problems. The key findings are:

- Solid-fluid model via Brinkman volume penalization on a Cartesian mesh is not exact but is acceptably accurate for conceptual design purposes, when the mesh is sufficiently refined.
- Solutions on locally refined meshes can be obtained by the multi-stage procedure with considerably less overall computational costs (up to 45% for 3D cases).
- OAD provides exact derivatives, with cost of less than 3% of the total optimization CPU times (in 3D cases). OAD facilitates a convenient differentiation and implementation. OAD is a general differentiation approach, which can be utilized for other complex CFD solvers as well.
- The results promisingly indicated a successful design process nearly from scratch, which suggests the potential utility of the density-based TO for an unmanned (automatic) designing tool.
- The aerodynamically optimized topologies featured interesting properties such as multi-element trailing structures or grooved surfaces to stabilize and control the flow in the wake region as well as enhancing the overall performance.
- Drag minimized airfoil geometries were obtained for Reynolds numbers up to $Re = 1000$ ($Re_a \approx 5000$). And up to $C_L/C_D = 5$ was achieved at $Re = 500$ ($Re_a \approx 1900$), supported by 7 separate trailing elements as flow stabilizers.

Appendix A: Discretization overview

By integration of Eq. (1) over a control volume V_p and application of Gauss theorem, the convection term becomes

$$\int_{V_p} \nabla \cdot (\mathbf{u}\mathbf{u}) dV = \oint_{\partial V_p} d\mathbf{S} \cdot (\mathbf{u}\mathbf{u}) \approx \sum_f \mathbf{S}_f \cdot (\mathbf{u}\mathbf{u})_f = \sum_f \varphi_f \mathbf{u}_f \tag{24}$$

where \mathbf{S}_f is face normal vector (pointing outwards) with magnitude equal to the face area, $\varphi_f = \mathbf{S}_f \cdot \mathbf{u}_f$ is the flux of fluid volume on face f , and

$$\mathbf{u}_f = \alpha \mathbf{u}_p + (1 - \alpha) \mathbf{u}_N \tag{25}$$

where subscript N stands for the neighboring cell.

For a constant ν , the diffusion term becomes

$$\int_{V_p} \nabla \cdot (\nu \nabla \mathbf{u}) dV = \oint_{\partial V_p} d\mathbf{S} \cdot (\nu \nabla \mathbf{u}) \approx \sum_f \mathbf{S}_f \cdot (\nu \nabla \mathbf{u})_f, \tag{26}$$

where

$$\mathbf{S}_f \cdot (\nabla \mathbf{u})_f = \underbrace{|\Delta_f| \frac{\mathbf{u}_N - \mathbf{u}_p}{|\mathbf{d}|}}_{\text{orthogonal}} + \underbrace{\mathbf{k}_f \cdot (\nabla \mathbf{u})_f}_{\text{non-orthogonal}}, \tag{27}$$

where $\Delta_f = \frac{S_f^2}{d \cdot S_f} \mathbf{d}$, $\mathbf{k}_f = \mathbf{S}_f - \Delta_f$, and $\mathbf{d} = \mathbf{x}_N - \mathbf{x}_p$.

By the mid-point rule approximation, the penalization term becomes

$$\int_{V_p} \frac{\chi}{\epsilon} \mathbf{u} dV \approx \frac{\chi}{\epsilon} \mathbf{u}_p V_p. \tag{28}$$

Summarizing Eq. (24)–(28), the momentum equations for u_x , u_y , and u_z become

$$a_p \mathbf{u}_p + \sum_N a_N \mathbf{u}_N = b_p - \frac{1}{\rho} \nabla p \tag{29}$$

where a denotes the resulting discretization coefficients, and b the velocity boundary conditions.

The pressure correction equation of SIMPLE algorithm is constructed by connecting continuity and momentum equations. For this, Eq. (2) is integrated over V_p and by applying Gauss theorem, it becomes

$$\int_{V_p} \nabla \cdot \mathbf{u} dV = \oint_{\partial V_p} d\mathbf{S} \cdot \mathbf{u} \approx \sum_f \mathbf{S}_f \cdot \mathbf{u}_f = 0. \tag{30}$$

To compute \mathbf{u}_f , here, Eq. (29) is first reformulated to isolate \mathbf{u} such that

$$\mathbf{u}_p = \frac{1}{a_p}(\mathcal{H}(\mathbf{u}) - \frac{\nabla p}{\rho}), \tag{31}$$

where $\mathcal{H}(\mathbf{u}) = b_p - \sum_N a_N \mathbf{u}_N$. Then, by applying the Rhie-Chow interpolation, \mathbf{u}_f is approximated by:

$$\mathbf{u}_f = \left(\frac{\mathcal{H}(\mathbf{u})}{a_p}\right)_f - \left(\frac{1}{a_p}\right)_f \left(\frac{\nabla p}{\rho}\right)_f. \tag{32}$$

Inserting Eq. (32) in (30) yields to

$$\sum_f \mathbf{S}_f \cdot \left[\left(\frac{1}{a_p}\right)_f \left(\frac{\nabla p}{\rho}\right)_f\right] = \sum_f \mathbf{S}_f \cdot \left(\frac{\mathcal{H}(\mathbf{u})}{a_p}\right)_f, \tag{33}$$

which, in a succinct form, becomes

$$\sum_f \Gamma_f^p = \sum_f \tilde{\varphi}_f, \tag{34}$$

where

$$\Gamma_f^p = \frac{1}{\rho} \left(\frac{1}{a_p}\right)_f (|\mathbf{d}_f| \frac{p_N - p_P}{|\mathbf{d}|} + \mathbf{k}_f \cdot (\nabla p)_f), \tag{35}$$

$$\text{and } \tilde{\varphi}_f = \mathbf{S}_f \cdot (\mathcal{H}(\mathbf{u})/a_p)_f.$$

Finally, the variable fluid fluxes φ_f are corrected using the updated pressure fluxes, via

$$\varphi_f \approx \tilde{\varphi}_f - \Gamma_f^p. \tag{36}$$

In SIMPLE algorithm, the flow solutions are obtained by starting with an initial guess for \mathbf{u} , and then by iteratively

solving Eqs. (31) and (34) for \mathbf{u} and p (with relaxation) and correcting φ (36), until convergence.

Appendix B: Operator-based analytical differentiation (OAD) for simpleFOAM

Residuals matrix forms: The residuals should be reformulated in algebraic (matrix) forms for the differentiation process. Therefore, Eq. (29) can be written as

$$\mathbf{r}^{\mathbf{u}_i} = \mathbf{K} \mathbf{u}_i - \mathbf{b}_i^{\mathbf{u}} + \mathbb{D}(\mathbf{v}) \mathbf{G}_i \mathbf{p} \approx 0 \tag{37}$$

where the operator $\mathbb{D}(\cdot)$ builds a diagonal matrix for a given vector, and constant $\mathbf{b}_i^{\mathbf{u}}$ vectors account for boundary conditions. For a 3D problem, three velocity residual vectors, namely $\mathbf{r}^{\mathbf{u}_x}$, $\mathbf{r}^{\mathbf{u}_y}$ and $\mathbf{r}^{\mathbf{u}_z}$ are defined. Note that pressure is scaled by inverse of density. The matrix \mathbf{K} is the sum of divergence, Laplace and Brinkman volume penalization (BVP) operators, such that

$$\mathbf{K} = \mathbf{M}_{\text{div}}^{\mathbf{u}} + \mathbf{M}_{\text{Lap}}^{\mathbf{u}} + \mathbf{M}_{\text{BVP}}^{\mathbf{u}}, \tag{38}$$

and the pressure gradient operator \mathbf{G} is defined as

$$\mathbf{G}_i = \mathbf{M}_{\text{grad},i} \mathbf{M}_{\text{cf}}^p, \tag{39}$$

where the \mathbf{M}_{cf}^p operator interpolates pressure from cell centers to the cell faces, and $\mathbf{M}_{\text{grad},i}$ operator computes its gradients in $i \in \{x, y, z\}$ directions.

Table 4 List of operators, defined to derive the residuals in matrix forms

Matrix	Cells/internal faces	Fixed-value boundaries	Fixed-flux boundaries
$\mathbf{M}_{\text{Lap}}^{\mathbf{u}}$	$M_{o,o}^f = M_{n,n}^f = v_f \frac{ S_f }{ \mathbf{d}_{no} }, M_{n,o}^f = M_{o,n}^f = -v_f \frac{ S_f }{ \mathbf{d}_{no} }$	$M_{o,o}^f = v_f \frac{ S_f }{ \mathbf{d}_{fo} }$	–
$\mathbf{M}_{\text{Lap}}^p$	$M_{o,o}^f = M_{n,n}^f = a_f \frac{ S_f }{ \mathbf{d}_{no} }, M_{n,o}^f = M_{o,n}^f = -a_f \frac{ S_f }{ \mathbf{d}_{no} }$	–	$M_{o,o}^f = a_f \frac{ S_f }{ \mathbf{d}_{fo} }$
$\mathbf{M}_{\text{div}}^{\mathbf{u}}$	$M_{o,o}^f = -M_{n,n}^f = \varphi$ if $\varphi \geq 0, M_{n,n}^f = -M_{o,o}^f = -\varphi$ if $\varphi < 0$	–	$M_{o,o}^f = \varphi$
$\mathbf{M}_{\text{div}}^{\tilde{\varphi}}$	$M_{o,f}^f = \frac{1}{v_o} \frac{S_f \cdot \mathbf{d}_{of}}{ S_f \cdot \mathbf{d}_{of} }, M_{n,f}^f = \frac{1}{v_n} \frac{S_f \cdot \mathbf{d}_{nf}}{ S_f \cdot \mathbf{d}_{nf} }$	$M_{o,f}^f = \frac{1}{v_o} \frac{S_f \cdot \mathbf{d}_{of}}{ S_f \cdot \mathbf{d}_{of} }$	$M_{o,f}^f = \frac{1}{v_o} \frac{S_f \cdot \mathbf{d}_{of}}{ S_f \cdot \mathbf{d}_{of} }$
$\mathbf{M}_{\text{grad},i}$	$M_{o,f}^f = \frac{S_{f,i}}{v_o}, M_{n,f}^f = \frac{S_{f,i}}{v_n}$	$M_{o,f}^f = \frac{S_{f,i}}{v_o}$	$M_{o,f}^f = \frac{S_{f,i}}{v_o}$
$\mathbf{M}_{\text{BVP}}^{\mathbf{u}}$	$M_{c,c} = \frac{\gamma_c}{\epsilon} \nu_c$	–	–
$\mathbf{M}_{\text{flux}}^p$	$M_{f,o} = -M_{f,n} = -a_f \frac{ S_f }{ \mathbf{d}_{no} }$	–	$M_{f,o} = -a_f \frac{ S_f }{ \mathbf{d}_{fo} }$
\mathbf{M}_{cf}^p	$M_{f,o} = 1 - M_{f,n} = \frac{\mathbf{d}_{nf}}{ \mathbf{d}_{no} }$	–	$M_{f,o} = 1$
$\mathbf{M}_{\text{cf}}^{\alpha}$	$M_{f,o} = 1 - M_{f,n} = \frac{\mathbf{d}_{nf}}{ \mathbf{d}_{no} }$	$M_{f,o} = 1$	$M_{f,o} = 1$
\mathbf{M}_{cf}^h	$M_{f,o} = 1 - M_{f,n} = \frac{\mathbf{d}_{nf}}{ \mathbf{d}_{no} }$	–	$M_{f,o} = 1$

Note that o and c are owner and neighbor global indexes of face f, respectively; c is the cell global index, and the f-superscripted M values are accumulatively summed to form the global operator matrix. The operators are based on 1st order upwind for divergence, Gauss linear for Laplace and gradient terms and linear for interpolation. For this case an orthogonal mesh is considered

For the pressure residuals of Eq. (34), \mathbf{K} is first decomposed to diagonal and off-diagonal matrices by

$$\mathbf{K} = \mathbf{A} + \tilde{\mathbf{H}}, \tag{40}$$

and the discretization coefficient vector is formed by

$$\mathbf{a}_r = \mathbf{A}^{-1} \mathbf{v} \tag{41}$$

where, \mathbf{v} stands for the vector of cell volumes. The coefficients can be interpolated to faces by

$$\mathbf{a}_{r,f} = \mathbf{M}_{cf}^{a_r} \mathbf{a}_r. \tag{42}$$

To apply Rhie-Chow interpolation, \mathbf{h} is first defined by

$$\mathbf{h}_i = \mathbb{D}(\mathbf{v})^{-1}(\mathbf{b}_i - \tilde{\mathbf{H}}\mathbf{u}_i), \tag{43}$$

where

$$\tilde{\mathbf{h}}_i = \mathbb{D}(\mathbf{a}_r) \mathbf{h}_i, \tag{44}$$

and interpolating $\tilde{\mathbf{h}}$ to faces gives

$$\tilde{\mathbf{h}}_{f,i} = \mathbf{M}_{cf}^h \tilde{\mathbf{h}}_i + \mathbf{b}_i^u. \tag{45}$$

The face flux $\tilde{\varphi}_f$ of Eq. (34) can be constructed by

$$\tilde{\varphi} = \sum_{i=1}^n \mathbb{D}(\mathbf{S}_{f,i}) \tilde{\mathbf{h}}_{f,i} \tag{46}$$

where n can be either 2 or 3 depending the problem dimensions. Now the vector of pressure residuals becomes

$$\mathbf{r}^p = -\left(\mathbf{M}_{Lap}^p \mathbf{p} - \mathbf{b}_{Lap}^p\right) - \mathbb{D}(\mathbf{v})\mathbf{M}_{div}^{\tilde{\varphi}} \tilde{\varphi} \approx 0 \tag{47}$$

where \mathbf{M}_{Lap}^p is a Laplace operator with $\mathbf{a}_{r,f}$ as diffusion coefficients; and, \mathbf{b}_{Lap}^p vanishes by assuming zero pressure value and gradients on domain boundaries.

Lastly, Eq. (36) is constructed by

$$\mathbf{r}^\varphi = \varphi - \tilde{\varphi} + \mathbf{M}_{flux}^p \mathbf{p} \approx 0 \tag{48}$$

which is counted as the fluid volume flux residual.

Analytical differentiation: Supported by the set of defined operators, the dependency of the residuals (Eq. (37), (47) and (48)) on state, intermediate and design variables are clarified. Such clarity significantly simplifies the differentiation process. Table 4 summarizes how these operators are assembled.

Beginning with momentum Eq. (37), the partial derivatives of \mathbf{r}^{u_i} are derived as

$$\mathbf{r}_{u_i}^{u_i} = \mathbf{K}, \tag{49}$$

$$\mathbf{r}_p^{u_i} = \mathbb{D}(\mathbf{v})\mathbf{G}_i, \tag{50}$$

$$\mathbf{r}_\varphi^{u_i} = \frac{\partial(\mathbf{M}_{div}^u \mathbf{u}_i)}{\partial \varphi}, \tag{51}$$

and

$$\mathbf{r}_\gamma^{u_i} = \frac{\partial(\mathbf{M}_{BVP} \mathbf{u}_i)}{\partial \gamma}, \tag{52}$$

with respect to state variables \mathbf{u}_i , \mathbf{p} and φ , and design variables γ , respectively. Note that $\mathbf{r}_{u_i}^{u_i} = 0$ if $i \neq j$.

For \mathbf{r}^p , the second term of Eq. (47) is first reformulated and simplified using Eq. (43)–(46) to obtain

$$\begin{aligned} \mathbb{D}(\mathbf{v})\mathbf{M}_{div}^{\tilde{\varphi}} \tilde{\varphi} &= \mathbb{D}(\mathbf{v})\mathbf{M}_{div}^{\tilde{\varphi}} \sum_{i=1}^n \mathbb{D}(\mathbf{S}_{f,i})(\mathbf{M}_{cf}^h \mathbb{D}(\mathbf{a}_r) \mathbf{h}_i + \mathbf{b}_i^u) = \\ & \underbrace{\mathbb{D}(\mathbf{v})\mathbf{M}_{div}^{\tilde{\varphi}} \left(\sum_{i=1}^n \mathbb{D}(\mathbf{S}_{f,i}) \mathbf{b}_i^u \right)}_{\mathbf{C}_1} + \underbrace{\sum_{i=1}^n \mathbb{D}(\mathbf{S}_{f,i}) \mathbf{M}_{cf}^h \mathbb{D}(\mathbf{v})^{-1} \mathbb{D}(\mathbf{a}_r) (\mathbf{b}_i - \tilde{\mathbf{H}}\mathbf{u}_i)}_{\mathbf{C}_{3,i}}, \end{aligned} \tag{53}$$

where \mathbf{C}_j are constant matrices. Now, \mathbf{r}^p in a succinct form becomes

$$\mathbf{r}^p = -\mathbf{M}_{Lap}^p \mathbf{p} - \mathbf{C}_1 \mathbf{C}_2 - \mathbf{C}_1 \mathbb{D}(\mathbf{v})^{-1} \mathbb{D}(\mathbf{a}_r) \sum_{i=1}^n \mathbf{C}_{3,i} (\mathbf{b}_i - \tilde{\mathbf{H}}\mathbf{u}_i). \tag{54}$$

The partial derivatives of pressure residual are then calculated as

$$\mathbf{r}_p^p = -\mathbf{M}_{Lap}^p, \tag{55}$$

$$\mathbf{r}_{u_i}^p = \mathbf{C}_1 \mathbb{D}(\mathbf{v})^{-1} \mathbb{D}(\mathbf{a}_r) \mathbf{C}_{3,i} \tilde{\mathbf{H}}, \tag{56}$$

$$\begin{aligned} \mathbf{r}_\varphi^p &= -\frac{\partial(\mathbf{M}_{Lap}^p \mathbf{p})}{\partial \mathbf{a}_{r,f}} \frac{\partial \mathbf{a}_{r,f}}{\partial \mathbf{a}_r} \frac{\partial \mathbf{a}_r}{\partial \varphi} \\ & - \mathbf{C}_1 \sum_{i=1}^n \mathbf{C}_{3,i} \left(\mathbb{D}(\mathbf{h}_i) \frac{\partial \mathbf{a}_r}{\partial \varphi} + \mathbb{D}(\mathbf{a}_r) \frac{\partial \mathbf{h}_i}{\partial \varphi} \right), \end{aligned} \tag{57}$$

and

$$\mathbf{r}_\gamma^p = -\left(\frac{\partial(\mathbf{M}_{Lap}^p \mathbf{p})}{\partial \mathbf{a}_{r,f}} \frac{\partial \mathbf{a}_{r,f}}{\partial \mathbf{a}_r} + \mathbf{C}_1 \sum_{i=1}^n \mathbf{C}_{3,i} \mathbb{D}(\mathbf{h}_i) \right) \frac{\partial \mathbf{a}_r}{\partial \gamma} \tag{58}$$

respectively for \mathbf{p} , \mathbf{u}_i , φ and γ , where chain-rule of differentiation is applied for intermediate variables such as \mathbf{a}_r . Note that from Eq. (42) $\frac{\partial \mathbf{a}_{r,f}}{\partial \mathbf{a}_r} = \mathbf{M}_{cf}^{a_r}$, and

$$\frac{\partial \mathbf{h}_i}{\partial \boldsymbol{\varphi}} = -\mathbb{D}(\mathbf{v})^{-1} \frac{\partial (\bar{\mathbf{H}} \mathbf{u}_i)}{\partial \boldsymbol{\varphi}}, \quad (59)$$

from Eq. (43).

Using the \mathbf{C}_j matrices, \mathbf{r}^φ can be simplified as

$$\mathbf{r}^\varphi = \boldsymbol{\varphi} + \mathbf{M}_{\text{flux}}^p \mathbf{p} - \mathbf{C}_2 - \mathbb{D}(\mathbf{a}_r) \sum_{i=1}^n \mathbf{C}_{3,i} \mathbf{h}_i, \quad (60)$$

and its derivatives are obtained by

$$\mathbf{r}_p^\varphi = \mathbf{M}_{\text{flux}}^p, \quad (61)$$

$$\mathbf{r}_{u_i}^\varphi = \mathbb{D}(\mathbf{v})^{-1} \mathbb{D}(\mathbf{a}_r) \mathbf{C}_{3,i} \bar{\mathbf{H}}, \quad (62)$$

$$\begin{aligned} \mathbf{r}_\varphi^\varphi = & \mathbf{I} + \frac{\partial (\mathbf{M}_{\text{flux}}^p \mathbf{p})}{\partial \mathbf{a}_{r,f}} \mathbf{M}_{\text{cf}}^{a_r} \frac{\partial \mathbf{a}_r}{\partial \boldsymbol{\varphi}} \\ & - \sum_{i=1}^n \mathbf{C}_{3,i} \left(\mathbb{D}(\mathbf{h}_i) \frac{\partial \mathbf{a}_r}{\partial \boldsymbol{\varphi}} + \mathbb{D}(\mathbf{a}_r) \frac{\partial \mathbf{h}_i}{\partial \boldsymbol{\varphi}} \right) \end{aligned} \quad (63)$$

and

$$\mathbf{r}_\gamma^\varphi = \left(\frac{\partial (\mathbf{M}_{\text{flux}}^p \mathbf{p})}{\partial \mathbf{a}_{r,f}} \mathbf{M}_{\text{cf}}^{a_r} - \sum_{i=1}^n \mathbf{C}_{3,i} \mathbb{D}(\mathbf{h}_i) \right) \frac{\partial \mathbf{a}_r}{\partial \boldsymbol{\gamma}}, \quad (64)$$

with respect to \mathbf{p} , \mathbf{u}_i , $\boldsymbol{\varphi}$ and $\boldsymbol{\gamma}$, respectively, where \mathbf{I} is the identity matrix.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflicts of interest The authors declare that they have no conflicts of interest.

Replication of results The developed open-source code is available at <https://github.com/topoptdev/aerotop>

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Alexandersen J, Andreassen CS (2020) A review of topology optimisation for fluid-based problems. *Fluids* 5(1):29
- Balay S, Abhyankar S, Adams MF, Benson S, Brown J, Brune P, Buschelman K, Constantinescu EM, Dalcin L, Dener A, Eijkhout V, Gropp WD, Hapla V, Isaac T, Jolivet P, Karpeev D, Kaushik D, Knepley MG, Kong F, Kruger S, May DA, McInnes LC, Mills RT, Mitchell L, Munson T, Roman JE, Rupp K, Sanan P, Sarich J, Smith BF, Zampini S, Zhang H, Zhang H, Zhang J (2021) PETSc Web page. <https://petsc.org/>.
- Barrera JL, Geiss MJ, Maute K (2020) Hole seeding in level set topology optimization via density fields. *Struct Multidisc Optim* 61(4):1319–1343
- Bendsøe MP, Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method. *Comput Methods Appl Mech Eng* 71(2):197–224
- Borrvall T, Petersson J (2003) Topology optimization of fluids in stokes flow. *Int J Numer Methods Fluids* 41(1):77–107
- Brezillon J, Dwight R (2005) Discrete adjoint of the navier-stokes equations for aerodynamic shape optimization. *Evolutionary and Deterministic Methods for Design (EUROGEN)*
- Carbou G, Fabrie P (2003) Boundary layer for a penalization method for viscous incompressible flow. *Adv Differ Equ* 8(12):1453–1480
- Carnarius A, Thiele F, Özkaya E, Nemili A, Gauger NR (2011) Optimal control of unsteady flows using a discrete and a continuous adjoint approach. In: *IFIP conference on system modeling and optimization*, Springer, pp 318–327
- Dede EM (2009) Multiphysics topology optimization of heat transfer and fluid flow systems. In: *Proceedings of the COMSOL Users Conference*
- Dilgen CB, Dilgen SB, Fuhrman DR, Sigmund O, Lazarov BS (2018) Topology optimization of turbulent flows. *Comput Methods Appl Mech Eng* 331:363–393
- Dwight RP, Brezillon J (2006) Effect of approximations of the discrete adjoint on gradient-based optimization. *AIAA J* 44(12):3022–3031
- Economou TD, Palacios F, Copeland SR, Lukaczyk TW, Alonso JJ (2016) Su2: an open-source suite for multiphysics simulation and design. *Aiaa J* 54(3):828–846
- Elham A, van Tooren MJ (2021) Discrete adjoint aerodynamic shape optimization using symbolic analysis with openfemflow. *Struct Multidisc Optim*. <https://doi.org/10.1007/s00158-020-02799-7>
- Evgrafov A, Gregersen MM, Sørensen MP (2011) Convergence of cell based finite volume discretizations for problems of control in the conduction coefficients. *ESAIM* 45(6):1059–1080
- Feppon F, Allaire G, Dapogny C, Jolivet P (2020) Topology optimization of thermal fluid-structure systems using body-fitted meshes and parallel computing. *J Comput Phys* 417:109574
- Garcke H, Hinze M, Kahle C, Lam KF (2018) A phase field approach to shape optimization in navier-stokes flow with integral state constraints. *Adv Comput Math* 44(5):1345–1383
- Ghasemi A, Elham A (2021) Multi-objective topology optimization of pin-fin heat exchangers using spectral and finite-element methods. *Struct Multidisc Optim* 64(4):2075–2095
- Griewank A, Walther A (2008) *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM
- He P, Mader CA, Martins JR, Maki KJ (2020) Dafoam: an open-source adjoint framework for multidisciplinary design optimization with openfoam. *AIAA J* 58(3):1304–1319

- Hester EW, Vasil GM, Burns KJ (2021) Improving accuracy of volume penalised fluid-solid interactions. *J Computat Phys* 430:110043
- Jenkins N, Maute K (2015) Level set topology optimization of stationary fluid-structure interaction problems. *Struct Multidisc Optim* 52(1):179–195
- Kenway GK, Mader CA, He P, Martins JR (2019) Effective adjoint approaches for computational fluid dynamics. *Progress Aerospace Sci* 110:100542
- Kondoh T, Matsumori T, Kawamoto A (2012) Drag minimization and lift maximization in laminar flows via topology optimization employing simple objective function expressions based on body force integration. *Struct Multidisc Optim* 45(5):693–701
- Kreissl S, Pingen G, Maute K (2011) Topology optimization for unsteady flow. *Int J Numer Methods Eng* 87(13):1229–1253
- Lundgaard C, Alexandersen J, Zhou M, Andreasen CS, Sigmund O (2018) Revisiting density-based topology optimization for fluid-structure-interaction problems. *Struct Multidisc Optim* 58(3):969–995
- Marck G, Nemer M, Harion JL (2013) Topology optimization of heat and mass transfer problems: laminar flow. *Numer Heat Transf Part B* 63(6):508–539
- Moukalled F, Mangani L, Darwish M et al (2016) The finite volume method in computational fluid dynamics, vol 113. Springer, Berlin
- Nadarajah S, Jameson A (2000) A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. In: 38th Aerospace sciences meeting and exhibit, p 667
- Nielsen EJ, Anderson WK (1999) Aerodynamic design optimization on unstructured meshes using the navier-stokes equations. *AIAA J* 37(11):1411–1419
- Nørgaard SA, Sagebaum M, Gauger NR, Lazarov BS (2017) Applications of automatic differentiation in topology optimization. *Struct Multidisc Optim* 56(5):1135–1146
- Patankar SV, Spalding DB (1983) A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. In: Numerical prediction of flow, heat transfer, turbulence and combustion. Elsevier, pp 54–73
- Peter JE, Dwight RP (2010) Numerical sensitivity analysis for aerodynamic optimization: a survey of approaches. *Comput Fluids* 39(3):373–391
- Pingen G, Evgrafov A, Maute K (2007) Topology optimization of flow domains using the lattice boltzmann method. *Struct Multidisc Optim* 34(6):507–524
- Pornsin-Sirirak TN, Tai Y, Nassef H, Ho C (2001) Titanium-alloy mems wing technology for a micro aerial vehicle application. *Sens Actuators A* 89(1–2):95–103
- Rhie C, Chow WL (1983) Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA J* 21(11):1525–1532
- Sanderson C, Curtin R (2016) Armadillo: a template-based c++ library for linear algebra. *J Open Source Softw* 1(2):26
- Sanderson C, Curtin R (2018) A user-friendly hybrid sparse matrix class in c++. In: International congress on mathematical software. Springer, pp 422–430
- Svanberg K (1995) A globally convergent version of mma without linesearch. In: Proceedings of the first world congress of structural and multidisciplinary optimization, Goslar, Germany vol 28, pp 9–16
- Villanueva CH, Maute K (2017) Cutfem topology optimization of 3d laminar incompressible flow problems. *Comput Methods Appl Mech Eng* 320:444–473
- Yoon GH (2010) Topology optimization for stationary fluid-structure interaction problems using a new monolithic formulation. *Int J Numer Methods Eng* 82(5):591–616
- Yoon GH (2016) Topology optimization for turbulent flow with spalart-allmaras model. *Comput Methods Appl Mech Eng* 303:288–311
- Yu M, Ruan S, Gu J, Ren M, Li Z, Wang X, Shen C (2020) Three-dimensional topology optimization of thermal-fluid-structural problems for cooling system design. *Struct Multidisc Optim* 62(6):3347–3366
- Zhou M, Lian H, Sigmund O, Aage N (2018) Shape morphing and topology optimization of fluid channels by explicit boundary tracking. *Int J Numer Methods Fluids* 88(6):296–313

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.