Topology Optimization of Turbulent Flows

DOI:
10.1016/j.cma.2017.11.029

Document Version
Accepted author manuscript

Topology Optimization of Turbulent Flows

Cetin B. Dilgen[1,*], Sumer B. Dilgen[1,2], David R. Fuhrman, Ole Sigmund, Boyan S. Lazarov[3]

Department of Mechanical Engineering
Technical University of Denmark
Nils Koppels Allé, Building 404
DK-2800, Denmark

Abstract

The aim of this work is to present a fast and viable approach for taking into account turbulence in topology optimization of complex fluid flow systems, without resorting to any simplifying assumptions in the derivation of discrete adjoints. Topology optimization is an iterative gradient-based design process which minimizes an objective and satisfies a set of selected design constraints by distributing material in a design domain. The gradients are obtained using adjoint sensitivity analysis which requires solutions of a forward state problem and an additional adjoint problem. In the presented article the forward solver is based on finite volume discretized Reynolds-averaged Navier-Stokes equations coupled with either one- or two-equation turbulence closure models, and the adjoint solver is obtained via automatic differentiation. The presented approach is demonstrated on the optimization of several 2D and 3D examples including a detailed comparison to designs and sensitivities obtained with different turbulence models and under a frozen turbulence assumption. The results demonstrate the importance of exact sensitivity analysis and open new possibilities for the design of large scale multiphysics problems involving turbulent flows.

Keywords: Topology Optimization, Automatic Differentiation, Turbulent Flow

1. Introduction

Computational fluid dynamics (CFD) is an extremely important tool in the design of complex flow systems. Compared to experiments it offers a relatively inexpensive method for obtaining valuable information through virtual flow simulations in a computer environment. However, even for relatively simple fluid systems, due to the inherently complex nonlinear behavior, prior knowledge of potential design improvements is not often available. In such cases the pursuit of better (optimal) designs can easily be prohibited by the inefficiency of trial-and-error approaches. To this end, topology optimization [1] can provide valuable solutions by distributing a volume of material in a selected design domain through optimization of a chosen criterion, without defining any initial shape or topology prior to the optimization process.

Topology optimization is an iterative gradient-based design process. A material distribution is found by solving an optimization problem, while satisfying a set of design constraints and governing equations that define the problem physics. Ideally, the material distribution is represented by a density which takes values of either one and zero, representing the two materials considered in the problem, i.e. if a point in the design domain is occupied with fluid the density is equal to one, and if it is occupied with solid the density field is equal to zero. In order to utilize gradient-based optimization techniques the discrete zero/one

*Corresponding author
Email addresses: cedil@mek.dtu.dk (Cetin B. Dilgen), sydil@elektro.dtu.dk (Sumer B. Dilgen)
[1]These authors contributed equally to this work.
[2]*Present address:* Department of Electrical Engineering, Technical University of Denmark, Denmark.
[3]*Present address:* School of Mechanical, Aerospace and Civil Engineering, The University of Manchester, UK.

problem is relaxed, and the density field is allowed to take intermediate values. Brinkman penalization is usually utilized to define a variable porous material which corresponds to the intermediate density values [2]. The earliest examples of utilizing topology optimization for fluid flow problems considered simplified Stokes flow models [2], with large scale examples found in [3]. More recently, [4] presented a locally cubically convergent algorithm for topology optimization of Stokes flows. Later optimizations of laminar flow problems using Navier-Stokes equations and Lattice Boltzmann methods have been demonstrated in [5, 6, 7] and [8, 9, 10], respectively. For general shape optimization, the derivation and the application of the topological sensitivities for Stokes and Navier-Stokes equations can be found in [11, 12, 13]. Considering the topology optimization of turbulent flows, [14, 15] utilized a simplified sensitivity analysis through the so-called frozen turbulence assumption, i.e, variations of the turbulence field with respect to design variables are neglected. Following the derivation of a continuous adjoint formulation for the Spalart-Allmaras turbulence model [16], recent work has demonstrated a topology optimization framework in [17].

In contrast to all previous works, the focus of the present paper is on the development of a discrete forward and adjoint Reynolds-averaged Navier-Stokes (RANS) solver for topology optimization with a scalable and computationally cheap procedure for gradient analysis where *exact* sensitivities are obtained for the considered discretization. The basic model will be coupled with different turbulence closures with minimal effort in terms of the required development time. It should be noted that, both continuous and discrete adjoint methods produce consistent sensitivities with the two approaches having their advantages/disadvantages (see [18, 19] for comparison between the two methods.). For complex CFD solvers, such as those utilizing segregated and iterative solution approaches such as SIMPLE algorithms [20], derivation of discrete adjoints by hand is a non-trivial and error-prone task. To this end, automatic differentiation (AD) [21, 22] will likewise be employed for calculating the exact gradients of the optimization objective and constraints in the topology optimization problems considered.

AD is a set of techniques enabling the exact numerical calculation of the gradients of a function implemented in a computer program. Similar to traditional sensitivity analysis, two modes can be clearly distinguished: forward and reverse modes, corresponding respectively to direct and adjoint differentiation. Depending on the capabilities of the programming language, AD can be implemented either by source code transformations or by operator overloading techniques; A detailed introduction can be found in [22]. The technique is mainly applied within shape and size optimization problems with a relatively small number of control/design variables. A source transformation AD tool is demonstrated in both reverse and forward modes to derive an exact adjoint version of the SIMPLE algorithm for shape optimization in [23]. Operator overloading is demonstrated in [24], where the OpenFOAM library [25] is differentiated in a black-box manner. The Dolfin-adjoint project [26] automatically derives the discrete adjoint given a differentiable forward model and an example of its application for topology optimization of Stokes flow problem is given in [27]. Also, both the FEniCS project [28] and COMSOL Multiphysics software [29] has automatic differentiation support. It should be pointed out that such black box differentiation brings immense memory requirements, and requires check-point algorithms [30] to store intermediate computations and to reduce the memory consumption. Alternatively, instead of differentiating the implemented code in a black box manner, the reverse mode of AD can be employed selectively, as demonstrated in [31, 32]. Herein, it is proposed to apply AD only in the formation of the exact discrete adjoint equation, thus enabling a significant reduction in memory requirements. The proposed approach requires the explicit solution of an adjoint system of equations, and the obtained adjoint solution times are similar to those of forward modes, in contrasts to other results reported in the literature. The proposed methodology produces exact consistent sensitivities and can be easily applied to any set of partial differential equations. Similar approaches for shape optimization problems can be found in [31, 33].

To the best of the authors' knowledge, topology optimization of turbulent flows has not yet been demonstrated with exact sensitivities obtained with discrete adjoint approach and without any simplifying assumptions in the derivation of the discrete adjoints. Hence, the methodology developed in this paper can be seen as a contribution to the topology optimization of complex flow problems. The present work provides the foundation for additional applications, and for further investigation, development and expansion of the method on complex multiphysics problems.

This paper is organized as follows: First, the governing equations, along with the inclusion of Brinkman

penalization, are described in section 2. Topology optimization is introduced in section 3, together with the associated adjoint sensitivity analysis. Automatic differentiation is then briefly presented in section 4, and the implementation and the verification of discrete adjoint solvers for (two equation) k-ω and (one equation) Spalart-Allmaras turbulence models are discussed in section 5. Comparison with results based on a frozen turbulence assumption is likewise included. Finally, the developed techniques and methodology are demonstrated in the optimization of several 2D and 3D turbulent fluid flow problems in section 6. Additional details on the employed finite volume discretization, the implementation and the validation of the implemented finite volume solver are presented in Appendix A-Appendix C.

2. Governing equations

The considered problems are governed by the steady-state incompressible Reynolds-averaged Navier-Stokes (RANS) equations given as

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

$$\nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = \nabla \cdot (2\nu \mathbf{S}) - \frac{1}{\rho}\nabla p + \nabla \cdot \mathbf{T}_t - \lambda \chi(\gamma)\mathbf{u} \tag{2}$$

where $\mathbf{u}$ is the mean velocity vector, p is the pressure, ν is the kinematic viscosity of the fluid, ρ is the fluid density, and λ (formally having units of inverse time) is the so-called Brinkman penalization parameter. The dimensionless function $\chi(\gamma)$ takes values ranging from zero to one, with these limits representing pure solid and fluid, respectively; The exact form of the interpolation function is discussed in section 3. The Brinkman penalization term effectively models spatially varying porosity, and is utilized for representing the solid/fluid distribution in the optimization process. The penalization term with $\chi(\gamma) = 1$, and sufficiently large enough λ, approximates a no-slip boundary condition at the fluid-solid interface. In the above, the mean strain rate tensor $\mathbf{S}$ is defined as

$$\mathbf{S} = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right) \tag{3}$$

The additional effects of turbulence are modeled through the Reynolds stress tensor $\mathbf{T}_t$, which accounts for additional normal and shear stresses on the fluid caused by turbulent eddies present in the flow. The Reynolds stress tensor is defined by invoking the so-called Boussinesq approximation as

$$\mathbf{T}_t = -\overline{\mathbf{u}' \otimes \mathbf{u}'} = 2\nu_t \mathbf{S} - \frac{2}{3}k\mathbf{I} \tag{4}$$

where ν_t is the turbulent eddy viscosity, δ_{ij} is the Kronecker delta, and

$$k = \frac{1}{2}\overline{\mathbf{u}' \cdot \mathbf{u}'} \tag{5}$$

is the turbulent kinetic energy per unit mass, with the prime superscript indicating fluctuating velocities. To achieve closure, an additional set of equations defining the eddy viscosity ν_t is required. To this end, the present work considers two different closure models, namely, the (1) two-equation k-ω model [34] and (2) the one-equation Spalart-Allmaras (SA) model [35], which are described in the following sub-sections. Implementation details are presented in Appendix A and Appendix B, and a validation of the solver is demonstrated in Appendix C.

2.1. k-ω model

The k-ω model calculates turbulent eddy viscosity ν_t as

$$\nu_t = \frac{k}{\tilde{\omega}}, \qquad \tilde{\omega} = max\left[\omega, C_{lim}\sqrt{\frac{2\mathbf{S}:\mathbf{S}}{\beta^*}}\right], \qquad C_{lim} = \frac{7}{8} \tag{6}$$

where : denotes the scalar product between two tensors (i.e. $\mathbf{a}$:$\mathbf{b} = a_{ij}b_{ij}$). The turbulent kinetic energy (per unit mass) k and the specific dissipation rate ω are obtained by solving two additional (steady state) transport equations

$$\nabla\cdot(\mathbf{u}\,k) = \mathbf{T}_t\!:\!\nabla\mathbf{u} - \beta^*\omega k + \nabla\cdot\left[\left(\nu + \sigma^*\frac{k}{\omega}\right)\nabla k\right] - \lambda\chi(\gamma)k \tag{7}$$

$$\nabla\cdot(\mathbf{u}\,\omega) = \frac{\alpha\omega}{k}\mathbf{T}_t\!:\!\nabla\mathbf{u} - \beta\omega^2 + \frac{\sigma_d}{\omega}\nabla k\cdot\nabla\omega + \nabla\cdot\left[\left(\nu + \sigma\frac{k}{\omega}\right)\nabla\omega\right] + \lambda\chi(\gamma)\left(\omega_b - \omega\right) \tag{8}$$

where $\beta = \beta_0 f_\beta$, $f_\beta = \frac{1+85\chi_\omega}{1+100\chi_\omega}$, $\chi_\omega = \left|\frac{\Omega_{ij}\Omega_{jk}S_{ki}}{(\beta^*\omega)^3}\right|$, $\mathbf{\Omega} = \frac{1}{2}\left(\nabla\mathbf{u} - \nabla\mathbf{u}^T\right)$, $\sigma_d = H\left(\nabla k\cdot\nabla\omega\right)\sigma_{d0}$ and $H\left(\cdot\right)$ is the Heaviside step function which returns a unity if the argument is positive and zero otherwise. It should be noted that χ_ω is zero for two-dimensional flows. The closure constants are $\alpha = 0.52$, $\beta_0 = 0.0708$, $\beta^* = 0.09$, $\sigma = 0.5$, $\sigma^* = 0.6$ and $\sigma_{d0} = 0.125$.

The turbulent kinetic energy k has a well defined boundary at walls given as

$$k_b = 0 \tag{9}$$

The specific dissipation rate ω, on the other hand, has a singular behavior near a wall. For smooth walls (as uniformly considered herein), the approximate (non-homogeneous Dirichlet type) boundary condition proposed by [36] will be utilized, corresponding to

$$\omega_b = \frac{60\nu}{\beta_1 y_1^2}, \qquad \beta_1 = 0.075 \tag{10}$$

where y_1 represents the distance from the wall to the cell center nearest the wall. Assuming a uniform mesh in the optimization domain, the distance y_1 is defined as half the cell height.

In the solidified regions, the penalization of wall boundary conditions for k and ω (Eqs. 9 and 10, respectively) is enforced through the Brinkman penalization term present in the last terms on the right hand sides of Eqs. 7 and 8, respectively.

2.2. Spalart-Allmaras model

The second turbulence model utilized here corresponds to the one-equation Spalart-Allmaras model [35]. In the past it has been utilized mainly for aerodynamic simulations, however, later modifications [37] have significantly broadened its range of applicability, and the model has gained popularity due to its simplicity and good numerical characteristics. In this model the eddy viscosity ν_t is obtained as follows:

$$\nu_t = \tilde{\nu} f_{v1}, \qquad f_{v1} = \frac{\widetilde{\chi}^3}{\widetilde{\chi}^3 + c_{v1}^3}, \qquad \widetilde{\chi} = \frac{\tilde{\nu}}{\nu} \tag{11}$$

with the viscosity parameter $\tilde{\nu}$ obtained via solution of the following transport equation

$$\nabla\cdot(\mathbf{u}\,\tilde{\nu}) = P - D + \frac{c_{b2}}{\sigma}\nabla\tilde{\nu}\cdot\nabla\tilde{\nu} + \frac{1}{\sigma}\nabla\cdot\left[\left(\nu + \tilde{\nu}\right)\nabla\tilde{\nu}\right] - \lambda\chi(\gamma)\tilde{\nu} \tag{12}$$

Here the production P and dissipation D terms respectively read

$$P = c_{b1}\tilde{S}\tilde{\nu}, \qquad D = c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 \tag{13}$$

where $\tilde{S} = \Omega + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2}$, $\Omega = \sqrt{2\mathbf{\Omega}\!:\!\mathbf{\Omega}}$, d is the distance from the closest wall, $r = min\left[\frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}, 10\right]$, $f_{v2} = 1 - \frac{\chi}{1+\chi f_{v1}}$, $f_w = g\left(\frac{1+c_{w3}^6}{g^6+c_{w3}^6}\right)^{1/6}$ and $g = r + c_{w2}(r^6 - r)$. The closure constants are $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $c_{v1} = 7.1$, $\sigma = 2/3$, $c_{w1} = \frac{c_{b1}}{\kappa^2} + \frac{1+c_{b2}}{\sigma}$, $c_{w2} = 0.3$, $c_{w3} = 2$ and $\kappa = 0.41$.

Since the turbulent kinetic energy k is not directly accessible in this model, the Reynolds stress tensor is calculated from $\mathbf{T}_t = 2\nu_t\mathbf{S}$. From comparison with the full Boussinesq approximation (Equation 4) it is noted that this neglects turbulent normal stresses. The eddy viscosity parameter $\tilde{\nu}$ in the Spalart-Allmaras model has a homogeneous Dirichlet boundary condition defined at walls as

$$\tilde{\nu}_{wall} = 0 \tag{14}$$

The additional penalization term, the last term on the right hand side of the Equation 12, approximates the wall boundary condition for $\tilde{\nu}$ (Equation 14) in the solidified regions.

As it can be seen from the model closure coefficients, the Spalart-Allmaras turbulence model needs the distance to the closest wall d. This is obtained by adopting a Poisson-like approach [38, 39]. For the distance calculation, the Poisson equation has a homogeneous Dirichlet boundary condition on walls. In the presence of porous material, the equation is penalized the same way as in the Equation 12 to be able to calculate the distance field d.

3. Topology optimization

A generic topology optimization problem, as utilized in the present work, can be defined as

$$\min_{\boldsymbol{\gamma}} \quad C(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma})) \tag{15}$$

$$\text{s.t.} \quad \mathbf{R}(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma})) = 0 \tag{16}$$

$$\mathbf{g}_i(\boldsymbol{\gamma}) \leq 0, \quad \forall i = 1, \ldots, N \tag{17}$$

$$0 \leq \boldsymbol{\gamma} \leq 1 \tag{18}$$

where $C(\cdot)$ is an objective function and $\boldsymbol{\gamma}$ represents the material (fluid/solid) distribution in the selected design domain ($\boldsymbol{\gamma}$ is a vector which is a discrete representation of the density γ), subject to an additional set of inequality constraints $\mathbf{g}_i$. $\mathbf{U}$ is the vector of state variables, and $\mathbf{R}$ is the vector of governing equations written in residual form. As stated earlier, the values of $\boldsymbol{\gamma}$ take value zero for cells filled with impermeable solid material and one for fluid. The intermediate values can be interpreted as porous material with different porosities. The goal of the optimization is to find the porosity distribution which minimizes the objective and fulfills the set of constraints.

The interpolation between solid and fluid is realized with the help of a function $\chi(\gamma)$ [2], which is defined as

$$\chi(\gamma) = q\frac{1 - \overline{\gamma}(\gamma)}{q + \overline{\gamma}(\gamma)} \tag{19}$$

where the parameter q controls the curvature of the above interpolation function. The interpolation functions for three different values of q are presented in Figure 1. Here it is seen that with $q = 10$ a near linear interpolation is realized. Lowering q increases the curvature of the interpolation function, resulting in a sharper transition between fluid and solid regions. In the presented results, all state fields, with the exception of ω, are penalized with $q = 0.1$. After a number of numerical experiments, it has been found that a sharper transition is needed in the ω equation for robust convergence in the presence of intermediate density/porosity regions. Thus, $q = 10^{-4}$ is used in the penalization of ω.

The values of $\overline{\gamma}$ are obtained from regularized Heaviside projection [40], controlled by a sharpness parameter β and a threshold $\eta = 0.5$. The projection is applied on a PDE filtered density field [41]. A detailed discussion of this procedure, together with other alternatives, can be found in [42]. Gradients of the objective and constraints with respect to the original design field γ are subsequently obtained by applying the chain rule, which accounts for every field transformation.

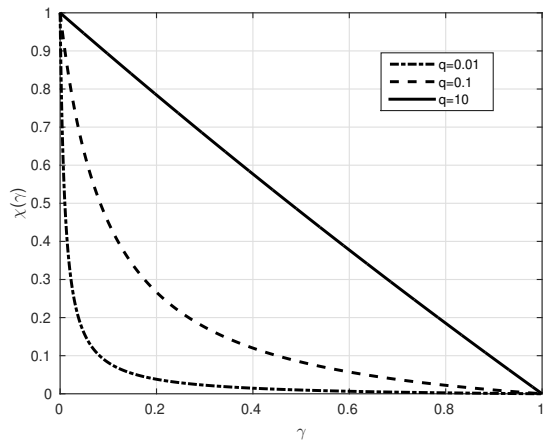


Figure 1: Interpolation function Equation 19 for different values of curvature parameter q. $\gamma = 1$ represents fluid, whereas $\gamma = 0$ represents solid. Values between 0 and 1 are referred to as "gray" or intermediate regions.

The penalization parameter λ, in all state equations, must be sufficiently high to effectively result in impermeable solid regions. If λ is relatively small, flow will diffuse through the porous media with finite velocity. On the other hand, large λ may result in slow convergence of the optimization process, convergence to poorly performing locally optimal solutions, and slow convergence of the forward state solver. To this end, the choice of λ is made based on the dimensionless Darcy number defined as:

$$Da = \frac{\nu U/L}{\lambda LU} = \frac{\nu}{\lambda L^2} \tag{20}$$

which represents the ratio of viscous forces to Darcy damping forces, where U and L are, respectively, characteristic velocity and length scales defining a given flow problem (to be clarified in what follows). Utilizing the (dimensionless) Da enables selection of (dimensional) λ in a consistent manner, regardless of problem scale. Through testing, it has been found that selecting a sufficiently small Darcy number ($Da \approx 10^{-5} - 10^{-6}$) [43] is sufficient to yield nearly impermeable solidified regions.

The solution of the optimization problem is obtained with the help of the Method of Moving Asymptotes (MMA) [44, 45]. Based on the gradients of the objective and the set of constraints, MMA updates the design field in an iterative fashion. The gradients are obtained by adjoint analysis. After obtaining the state solution, an adjoint system of equations is solved and the gradients of the objective and the constraints are computed with respect to the selected parametrization of the density field. Based on the gradients and the values of the constraints, MMA updates the design. Physically realistic designs require fine spatial resolution, resulting in a large number of degrees-of-freedom (DOFs) for representing the state solution, and high computational cost. Therefore, the state, the adjoint and the optimization solvers are parallelized, following the general strategy outlined in [46, 47], in order to obtain a final design in a reasonable amount of time.

Apart from the parallel implementation, the main difficulty in the above outlined optimization process is the calculation of the gradients of the objective and the constraints with respect to the parametrization of the material distribution. This calculation represents one of the main novel contributions of the present work. For estimations based on finite differencing, the computational cost scales proportionally to the number of design parameters. Thus, for topology optimization problems, which are characterized by a large number of design variables and relatively small number of constraints, an adjoint-based sensitivity analysis is the preferred approach i.e. for each objective or constraint function only a single state and adjoint solution is necessary to evaluate all sensitivities. Here, a discrete adjoint approach is adopted, where the state problem is first discretized, followed by the evaluation of the objective, the constraints and gradients, utilizing the discrete solution to the state problem.

3.1. Discrete adjoint

The evaluation of the gradients will now be demonstrated for a generic objective or constraint function $C(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma}))$. The state solution $\mathbf{U}(\boldsymbol{\gamma})$ is required to satisfy the governing equation, which is written in residual form $\mathbf{R}(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma})) = 0$. The objective function is augmented with the help of a Lagrange multiplier vector $\boldsymbol{\lambda}$, and the Lagrangian function $\mathcal{L}$ is written as

$$\mathcal{L} = C(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma})) + \boldsymbol{\lambda}^T \mathbf{R}(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma})) \tag{21}$$

For zero residual the Lagrangian coincides with the objective/constraint function. The derivative with respect to design parametrization can be written as

$$\begin{aligned} \frac{d\mathcal{L}}{d\boldsymbol{\gamma}} &= \frac{\partial C}{\partial \boldsymbol{\gamma}} + \frac{\partial C}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\boldsymbol{\gamma}} + \boldsymbol{\lambda}^T \left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\boldsymbol{\gamma}} \right) \\ &= \frac{\partial C}{\partial \boldsymbol{\gamma}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}} + \underbrace{\left(\frac{\partial C}{\partial \mathbf{U}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right)}_{=0} \frac{d\mathbf{U}}{d\boldsymbol{\gamma}} \end{aligned} \tag{22}$$

The term $\frac{d\mathbf{U}}{d\boldsymbol{\gamma}}$ is computationally very expensive to evaluate. Thus, to avoid its computation the indicated part of the above equation is set to zero. This requirement provides the adjoint equation for the optimization problem, which is given as

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T \boldsymbol{\lambda} = -\frac{\partial C}{\partial \mathbf{U}} \tag{23}$$

To obtain a Lagrange multiplier vector satisfying the adjoint equation, the final vector of gradients is evaluated as

$$\frac{dC}{d\boldsymbol{\gamma}} = \frac{\partial C}{\partial \boldsymbol{\gamma}} + \frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}}^T \boldsymbol{\lambda} \tag{24}$$

The adjoint procedure described above is applied to the already-discretized governing equations and a selected function. As this approach operates on the discrete level, where the boundary conditions are already accounted for, it can handle any cost function and governing equation in a robust way. Importantly, calculated sensitivities are always *exact* for the considered discretization.

4. Discrete adjoint via automatic differentiation

The Jacobian $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ in Equation 23 is not formed explicitly during the state solution. Rather, the forward solution is obtained iteratively using the SIMPLE algorithm [20]. The explicit formation of the Jacobian is an error-prone and tedious task, due to the large number of coupled state fields and applied corrections required to ensure stability of the discretization. Toward this end the presented work utilizes automatic differentiation (AD) in order to simplify, and at the same time allow for, exact gradients in the evaluation [22].

AD calculates derivatives of a function implemented as a computer code. The basic idea behind AD is the fact that every evaluation of the function can be decomposed to a set of primitive operations which are easy to differentiate. AD keeps track of all primitive operations and calculates the gradient of any complicated function by invoking the chain rule. The calculated derivatives are thus exact to machine precision. Thus, AD should be considered as a tool which simplifies the evaluation of derivatives through software techniques. AD can be implemented though source code transformations [48] or by using operator overloading features of modern programming languages such as C++ and Fortran90 [49, 50]. The advantages and disadvantages of the two approaches, as well as comparison to hand coded adjoints for several topology optimization problems, are discussed in detail in [32].

AD can differentiate a computer coded function using forward (tangent) and reverse (adjoint) modes. Forward mode provides derivatives in a single direction. Thus, the derivatives of all output variables are computed with respect to only one input variable in one forward sweep. A scalar valued function with n input arguments has to be executed n times for evaluating the gradients in forward mode. On the other hand, reverse mode can be seen as analogous to the discrete adjoint method described in subsection 3.1. The approach calculates the full set of derivatives with respect to all input variables by applying the chain rule from outputs to inputs. Thus, considering a scalar valued function, its gradient vector can be obtained in a single call to a reverse AD routine after a record (a tape) of all primitive operations. The tape records all operations during the forward evaluation of the objective/constraints functions. For a large number of design parameters and long and complex evaluations the memory requirements can become prohibitive, and special check pointing schemes are necessary to avoid excessive use of memory, as well as to reduce the number of repetitive evaluations.

The considered topology optimization problems consist of one or a few constraints and a single objective. Therefore, the natural AD differentiation mode is reverse mode. As the overloading approaches require fewer modifications to the state solution solver, and a preliminary comparison in terms of speed between [50] and [49] did not reveal significant differences, the gradients evaluation was implemented using the *Adept: automatic differentiation library* [50]. Applying AD directly in the forward solution process combined with the following objective and constraint evaluations would require check pointing and careful organization of the evaluation structure. As the linear algebra and the parallel communication libraries are not aware of the types and the structure of the AD differentiated code, the selected implementation utilizes AD selectively only for local operations. The exact process consists in forming a Jacobian, the right hand side of the adjoint equation, and finally evaluating the gradients using Equation 24.

4.1. Implementation of residual vector functions

The developed incompressible Navier-Stokes solver utilizes a collocated (cell-centered) grid arrangement. Linearization of the governing equations results in additional variables, in the form of face fluxes, which are introduced and stored in the cell face centers. Due to the iterative nature of the forward algorithm, calculating face fluxes from the interpolated face velocities does not satisfy the momentum residuals at convergence, i.e. during the iteration sequence, residuals are satisfied through both convected (nodal) and convecting (face flux) velocities. For a hexagonal cell the number of state variables is 12. These include the face fluxes and the state variables for the k-ω turbulence model. The inclusion of the SA model in the adjoint analysis follows the same steps as for the k-ω model, hence its presentation is omitted here for brevity.

The residual and the state vectors for the discrete model are given as

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 & \mathbf{p} & \mathbf{f}_1 & \dots & \mathbf{f}_6 & \mathbf{k} & \boldsymbol{\omega} \end{bmatrix}^T_{(12N_e)\times 1} \tag{25}$$

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}^{u_1} & \mathbf{R}^{u_2} & \mathbf{R}^{u_3} & \mathbf{R}^{p} & \mathbf{R}^{f_1} & \dots & \mathbf{R}^{f_6} & \mathbf{R}^{k} & \mathbf{R}^{\omega} \end{bmatrix}^T_{(12N_e)\times 1} \tag{26}$$

where N_e is the total number of cells in the domain. $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ denote the vectors of velocity components and $\mathbf{f}_i$ is the vector of face fluxes with $i = 1, \cdots, 6$ for a hexagonal cell.

Residual calculations of the velocity components ($\mathbf{R}^{u_1}, \mathbf{R}^{u_2}, \mathbf{R}^{u_3}$) and turbulence model variables ($\mathbf{R}^{k}$, $\mathbf{R}^{\omega}$) are easily differentiated, as for each cell, they reflect the governing equations and the transport equations for k and ω. The generic explicit discrete form of these equations is given for a single cell as

$$R^{\phi} = A_P \phi_P - \sum_N A_N \phi_N - S_{\phi} \approx 0$$

where ϕ stands for any variable other than pressure and fluxes (for more details see Appendix A).

The pressure residual is obtained from the iterative procedure and at convergence $p^l \approx p^{l-1}$, where the superscipt indicates the iteration stage. Thus, a residual for the pressure can be derived from its correction,

Equation B.13,

$$R^p = p_P - \left(p_P + \frac{\alpha_p}{a_P} \left[\sum_N a_N p'_N - \sum_f F^* \right] \right) \approx 0 \tag{27}$$

where F^* is calculated as in equation B.9. In the SIMPLE algorithm, the pressure correction is zero at convergence ($p' \approx 0$), therefore in the adjoint solver it is not necessary to declare the pressure correction as an independent variable, and it is hence set to zero for differentiation. With the above modification (taking out the terms with pressure correction) the residual for the pressure is calculated as

$$R^p = p_P - \left(p_P - \frac{\alpha_p}{a_P} \sum_f F^* \right) \approx 0 \tag{28}$$

Residual functions for fluxes are derived from their correction, Equation B.14, using the fact that $p' \approx 0$

$$R^{f_i} = F_i - F_i^* \approx 0 \tag{29}$$

where subscript i denotes the face number.

All of the above residual functions are implemented in a cell-wise manner. The derivative of the residual vector with respect to the state variables will be different than zero only for indices of state variables corresponding to a limited number of cells located around the considered cell. Taking into account all convective and diffusive schemes employed for the discretization, the stencil width for each row in the Jacobian is up to 33 for $3D$, and up to 13 for $2D$, problems.

4.2. Assembly of Jacobian matrices

Efficient calculation of the Jacobian matrices $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$, $\frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}}$ is crucial to yield good performance in the discrete adjoint solver. To this end, instead of directly computing the Jacobian matrices (by differentiating the whole residual vector), the implemented single-output residual functions are differentiated in an element loop. Thus, for each residual function, a row in the Jacobian is obtained through AD. This approach is beneficial, not only in term of memory requirements, but also for parallel calculation, as all differentiations are performed locally without the need of any parallel communications.

Active input variables are accompanied by a map structure containing the local and global indices of the current stencil. Thus, the calculated gradients are assembled directly into the global matrix. Algorithm 1 illustrates the general steps taken to differentiate the residual functions and assemble the Jacobian matrix.

Algorithm 1 Jacobian assembly

```
for Element n_i, i = 1, ..., N_e do
  Extract the required stencil for element based residual evaluation.
  Initialize active input variables (u_1, u_2, u_3, p, ..)
  for Every residual r = 1, ..., N_R do
    Start new recording
    Call overloaded residual function
    Reverse sweep on the recording
    for j = 1, ..., N_stencil do
      Extract the gradients
      Insert the gradients into the matrix
    end for
  end for
end for
Resulting matrices with sizes of: ∂R/∂U → (N_R × N_e) × (N_R × N_e)
∂R/∂γ → (N_R × N_e) × (N_e)
```

In the above algorithm N_R is the number of residuals per cell. For a hexahedron and the k-ω turbulence model $N_R = 12$. After differentiation and assembly, the Jacobian matrix has the following form:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} = \begin{bmatrix} \mathbf{R}_{u_1}^{u_1} & \mathbf{R}_{u_2}^{u_1} & \mathbf{R}_{u_3}^{u_1} & \mathbf{R}_{p}^{u_1} & \dots & \mathbf{R}_{f_6}^{u_1} & \mathbf{R}_{k}^{u_1} & \mathbf{R}_{\omega}^{u_1} \\ \mathbf{R}_{u_1}^{u_2} & \ddots & & & & & & \vdots \\ \mathbf{R}_{u_1}^{u_3} & & \ddots & & & & & \\ \mathbf{R}_{u_1}^{p} & & & \ddots & & & & \\ \vdots & & & & \ddots & & & \\ \mathbf{R}_{u_1}^{f_6} & & & & & \mathbf{R}_{f_6}^{f_6} & & \\ \mathbf{R}_{u_1}^{k} & & & & & & \mathbf{R}_{k}^{k} & \\ \mathbf{R}_{u_1}^{\omega} & \dots & & & & & & \mathbf{R}_{\omega}^{\omega} \end{bmatrix}_{(12N_e)\times(12N_e)}, \frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}} = \begin{bmatrix} \mathbf{R}_{\gamma}^{u_1} \\ \mathbf{R}_{\gamma}^{u_2} \\ \mathbf{R}_{\gamma}^{u_3} \\ \mathbf{R}_{\gamma}^{p} \\ \vdots \\ \mathbf{R}_{\gamma}^{\omega} \end{bmatrix}_{(12N_e)\times N_e} \tag{30}$$

Each block, i.e. $\mathbf{R}_{u_3}^{u_2}$, in the matrix has size $N_e \times N_e$ where the subscripts define with respect to which variable the differentiation is done. The Jacobian has a highly sparse structure which is exploited in terms of storage. As can be seen from Equation 23 and Equation 24, the adjoint equation requires the transpose of the matrix. In order to avoid the transpose operation on the assembled system, the assembly is therefore performed directly on the transposed matrix.

4.3. Assembly of gradient vectors

Gradient vectors in the adjoint equation are solely based on the differentiation of the cost function $C(\boldsymbol{\gamma}, \mathbf{U}(\boldsymbol{\gamma}))$ with respect to either the design variables $\boldsymbol{\gamma}$ or state variables $\mathbf{U}$. For any scalar valued cost function, first an overloaded version is implemented, where the active inputs have size corresponding to the number of elements. After recording the call to the overloaded objective function, gradients are computed and then extracted in an element loop. The process is demonstrated in algorithm 2.

Algorithm 2 Gradient vector assembly

Initialize active input variables $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{p}, ..)$
Start new recording
Call overloaded objective function
Reverse sweep on the recording
for Element n_i, $i = 1, ..., N_e$ **do**
  Extract the gradients
  Insert the gradients into the corresponding vectors
end for
Resulting vectors with sizes of: $\frac{\partial C}{\partial \mathbf{U}} \rightarrow (N_R \times N_e) \times 1$
$\frac{\partial C}{\partial \boldsymbol{\gamma}} \rightarrow (N_e) \times 1$

4.4. Adjoint solution method

The assembled adjoint equation is given as

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T \boldsymbol{\lambda} = -\frac{\partial C}{\partial \mathbf{U}} \tag{31}$$

The solution can be obtained by a direct factorization method, though such an approach does not scale well for large problems. Thus, an iterative solution technique would be preferable. The robustness of these solvers is controlled by the supplied preconditioners. Here the GMRES iterative method implemented in the PETSc library, in combination with a SIMPLER-like preconditioning scheme, ensures rapid convergence. Following the same idea behind the implemented forward flow solver, where pressure-velocity coupling has been dealt with by employing the SIMPLE algorithm, a similar preconditioner is set-up for the dual velocity-pressure

coupling in the adjoint variables, without employing any under-relaxation. The considered preconditioner is implemented by only utilizing the upper left 4×4 block of $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T$ (individual blocks of $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ can be seen from the Equation 30). Coupling to the rest of the system (adjoint fluxes and turbulent variables) is ignored, and block-wise preconditioning is implemented for these remaining adjoint variables. Hence, the following equation is considered

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix} \tag{32}$$

where each sub-block can be written as

$$\mathbf{A}_{11} = \begin{bmatrix} \left(\mathbf{R}_{u_1}^{u_1}\right)^T & 0 & 0 \\ 0 & \left(\mathbf{R}_{u_2}^{u_2}\right)^T & 0 \\ 0 & 0 & \left(\mathbf{R}_{u_3}^{u_3}\right)^T \end{bmatrix}, \qquad \mathbf{A}_{12} = \begin{bmatrix} \left(\mathbf{R}_{u_1}^{p}\right)^T \\ \left(\mathbf{R}_{u_2}^{p}\right)^T \\ \left(\mathbf{R}_{u_3}^{p}\right)^T \end{bmatrix} \tag{33}$$

$$\mathbf{A}_{21} = \begin{bmatrix} \left(\mathbf{R}_{p}^{u_1}\right)^T & \left(\mathbf{R}_{p}^{u_2}\right)^T & \left(\mathbf{R}_{p}^{u_3}\right)^T \end{bmatrix}, \qquad \mathbf{A}_{22} = \begin{bmatrix} \left(\mathbf{R}_{p}^{p}\right)^T \end{bmatrix} \tag{34}$$

The approximate Schur complement matrix is calculated as

$$\mathbf{S} = \mathbf{A}_{22} - \mathbf{A}_{21}\tilde{\mathbf{A}}_{11}^{-1}\mathbf{A}_{12} \tag{35}$$

where, $\tilde{\mathbf{A}}_{11}$ is a diagonal matrix, whose entries are the diagonal part of the $\mathbf{A}_{11}$. With the matrices defined above, the following steps define the application of the preconditioner on a vector $\left(\mathbf{r}_\mathbf{u}^*, \mathbf{r}_\mathbf{p}^*\right)$ with the resulting vector $(\mathbf{r}_\mathbf{u}, \mathbf{r}_\mathbf{p})$

$$\bar{\mathbf{r}}_\mathbf{p} = \mathbf{S}^{-1}\left[\mathbf{r}_\mathbf{p}^* - \mathbf{A}_{21}\tilde{\mathbf{A}}_{11}^{-1}\mathbf{r}_\mathbf{u}^*\right] \tag{36}$$

$$\bar{\mathbf{r}}_\mathbf{u} = \mathbf{A}_{11}^{-1}\left[\mathbf{r}_\mathbf{u}^* - \mathbf{A}_{12}\bar{\mathbf{r}}_\mathbf{p}\right] \tag{37}$$

$$\Delta\mathbf{r}_\mathbf{u} = \mathbf{A}_{11}^{-1}\left[\mathbf{r}_\mathbf{u}^* - \left(\mathbf{A}_{11}\bar{\mathbf{r}}_\mathbf{u} + \mathbf{A}_{12}\bar{\mathbf{r}}_\mathbf{p}\right)\right] \tag{38}$$

$$\Delta\mathbf{r}_\mathbf{p} = \mathbf{S}^{-1}\left[\mathbf{r}_\mathbf{p}^* - \left(\mathbf{A}_{21}\left(\bar{\mathbf{r}}_\mathbf{u} + \Delta\mathbf{u}\right) + \mathbf{A}_{22}\bar{\mathbf{r}}_\mathbf{p}\right)\right] \tag{39}$$

$$\mathbf{r}_\mathbf{u} = \bar{\mathbf{r}}_\mathbf{u} + \Delta\mathbf{r}_\mathbf{u} - \tilde{\mathbf{A}}_{11}^{-1}\mathbf{A}_{12}\Delta\mathbf{r}_\mathbf{p} \tag{40}$$

$$\mathbf{r}_\mathbf{p} = \bar{\mathbf{r}}_\mathbf{p} + \Delta\mathbf{r}_\mathbf{p} \tag{41}$$

which follows closely the steps in the SIMPLER preconditioner [51, 52]. Unlike utilizing SIMPLE-type methods as iterative solvers, above operations (Eqs. 36 to 41) are applied for preconditioning GMRES iterative solvers.

4.5. Objective and constraint functions

In the following examples the selected objective corresponds to volume-based power dissipation. An expression for power dissipation stems from the energy equation, and can be derived with the scalar multiplication of the momentum equations and the velocity vector. Taking into account the extra body force acting on the momentum equation due to the added penalization term, volume-based power dissipation is given as

$$C = \int_V \left(2\nu_{eff}\mathbf{S}{:}\mathbf{S} + \chi(\gamma)\mathbf{u}{\cdot}\mathbf{u}\right) dV \tag{42}$$

The fluid volume is bounded, and the constraint is given as

$$g_1(\boldsymbol{\gamma}) = \frac{\Delta V_i\, \gamma_i}{fV} - 1 \leq 0, \qquad i = 1, \ldots, N_e \tag{43}$$

where ΔV_i and γ_i are respectively the cell volume and design variable of the ith cell, V is the total volume of the design domain, and f is the prescribed fluid volume fraction.

For flow manifold optimization problems (subsection 6.2 and subsection 6.3) multiple outlets are constrained, such that specified mass flow rates across the outputs are obtained. The flux constraints are formulated as

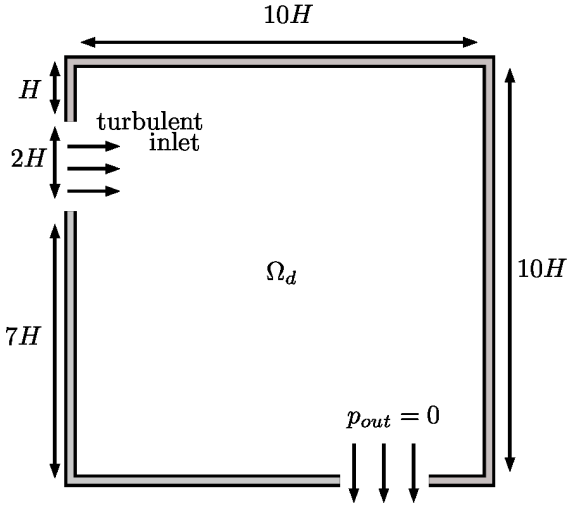
$$g_2(\mathbf{U}(\boldsymbol{\gamma})) = \frac{\int_{A_i} \mathbf{n} \cdot \mathbf{u} \, dA}{-q_i F_{in}} - 1 - \epsilon \leq 0 \qquad i = 1, m \tag{44}$$

where m is the number of constrained outlets, F_{in} is the fixed inlet flow rate calculated at the input boundary, q_i is the desired mass flux fraction at outlet i, and ϵ is the tolerance on the flux constraint. A tight tolerance corresponding to $\epsilon = 1 \times 10^{-4}$ is utilized. It should be noted that, in terms of implementation, the above flux constraint g_2 is a function of face flux F, which is a state variable. Hence, for each constrained outlet, an adjoint equation has to be solved in order to calculate the gradient of the constraint in Equation 44 with respect to the design variables. For every adjoint equation, AD is utilized only in the formation of the gradient vector $\frac{\partial g_2}{\partial \mathbf{U}}$, which is the source term of the adjoint equation. Jacobian matrices $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ in the adjoint, Equation 23, and $\frac{\partial \mathbf{R}}{\partial \boldsymbol{\gamma}}$, remain unchanged.

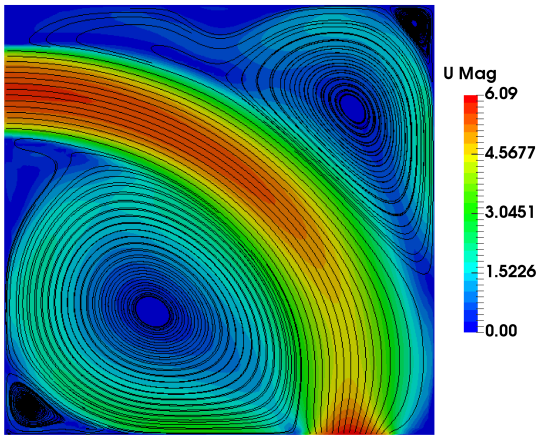
5. Gradient calculation

5.1. Sensitivity verification in a channel bend problem

The focus in this section is on verification of the adjoint sensitivity analysis. As an example, the considered computational domain for a channel bend problem is shown in Figure 2a. As seen, the domain consists of an empty box, having a single inlet and outlet. Half of the input channel height is taken to be $H = 0.1$ m. A fully developed turbulent flow profile is imposed as a boundary condition for velocities and turbulence model variables, obtained directly through an a priori simulation. The inlet bulk velocity is set to $U_b = 5$ m/s and the kinematic viscosity of the fluid is taken as $\nu = 5 \times 10^5$ m^2/s. The Reynolds number, defined with half inlet height and bulk velocity, is tailored to be $Re = U_b H/\nu = 10^4$; H and U_b are likewise taken as the characteristic scales defined in Equation 20. The computational domain consists of 13162 hexahedral cells clustered near the walls to ensure accuracy in the primal solution. The distance of the first cell to the walls in wall coordinates is kept as $y^+ = yU_f/\nu < 1$, where U_f is the friction velocity. A contour plot of the velocity magnitude from the primal solution utilizing the k-ω model, along with resulting streamlines, is presented in Figure 2b.



(a) 90 degree channel bend schematic.



(b) k-ω model, $Re = 10^4$, velocity magnitude contours [m/s] with streamlines.

Figure 2: Simulation setup and velocity contours for the channel bend problem.

The design domain, which coincides with the computational domain, is initialized with fluid only, i.e. $\gamma = 1$ is set everywhere in the domain. The interpolation function, Equation 19, is utilized in computing the Brinkman penalization with curvature parameter $q = 0.1$ and penalization parameter $\lambda = 10^3$ s^{-1}. The finite difference approximation for an arbitrary cell is obtained as

$$\frac{dC}{d\gamma_i} \approx \frac{C\left(\gamma_i + \Delta h, \mathbf{U}(\gamma_i + \Delta h)\right) - C\left(\gamma_i - \Delta h, \mathbf{U}(\gamma_i - \Delta h)\right)}{2\Delta h} \tag{45}$$

Thus, for each cell, two forward solutions are necessary for estimating the gradients of the selected objective by the central finite difference scheme. The step size is selected to be $\Delta h = 10^{-6}$.

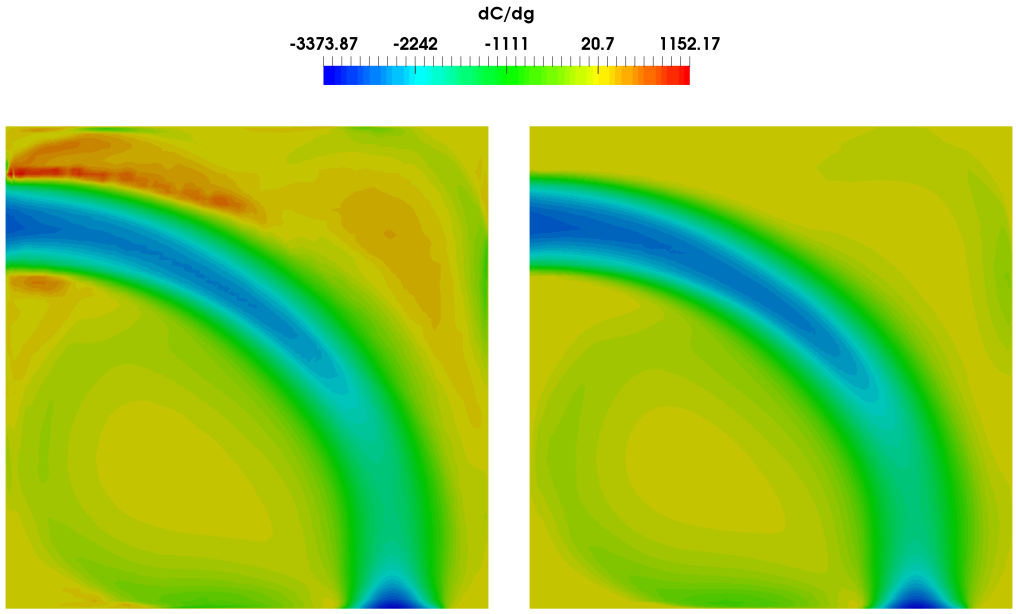
Simulations using both the SA and k-ω turbulence models have been performed. It should be stressed that throughout the present work the dependence of the distance function in the SA turbulence model on the design variables is neglected in the differentiation. The effect of this simplification on the calculated sensitivities with the adjoint SA model is illustrated in the Figure 6. For the selected verification case, the design domain is occupied only by fluid, and the distance function is not penalized. This allows for direct comparison between the gradients evaluated by finite difference and the AD adjoint for the SA model.

Derivative $\frac{dC}{d\gamma}$	Finite Difference	Adjoint k-ω	Frozen Turbulence
CV_1	-4.016668×10^{-3}	-4.016381×10^{-3}	-2.993492×10^{-3}
CV_2	-7.641710×10^{-4}	-7.649730×10^{-4}	-9.143874×19^{-4}
CV_3	$+3.366152 \times 10^{-4}$	$+3.360705 \times 10^{-4}$	-8.757109×10^{-4}
CV_4	-2.655306×10^{-2}	-2.655305×10^{-2}	-2.665780×10^{-2}

Table 1: Validation of calculated sensitivities for the channel bend problem with the k-ω adjoint solver. Cells are choosen arbitrarily.

Derivative $\frac{dC}{d\gamma}$	Finite Difference	Adjoint SA	Frozen Turbulence
CV_1	-1.488273×10^{-3}	-1.488297×10^{-3}	-2.571103×10^{-3}
CV_2	$+1.476542 \times 10^{-3}$	$+1.470075 \times 10^{-3}$	-6.637291×10^{-4}
CV_3	$+1.538834 \times 10^{-3}$	$+1.538780 \times 10^{-3}$	-6.786502×10^{-4}
CV_4	-2.530123×10^{-2}	-2.530116×10^{-2}	-2.572755×10^{-2}

Table 2: Validation of calculated sensitivities for the channel bend problem with the SA adjoint solver. Distance function is not penalized. Cells are choosen arbitrarily.

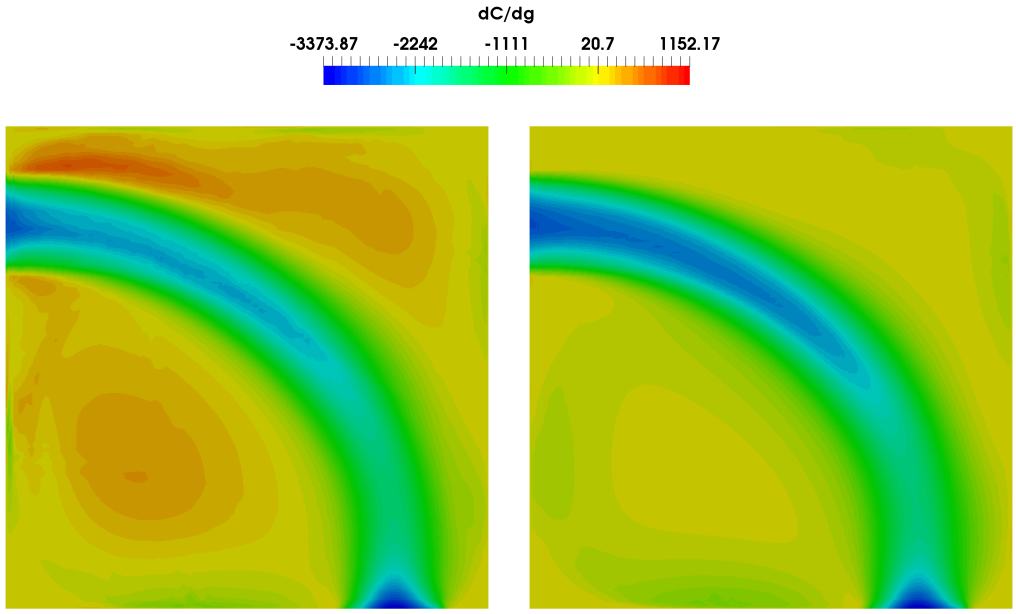


(a) Discrete k-ω adjoint solver.

(b) Discrete k-ω adjoint solver with frozen turbulence.

Figure 3: Calculated sensitivity field of the power dissipation objective function C with respect to all design variables γ for the channel bend problem. The cell values are scaled with cell volumes.

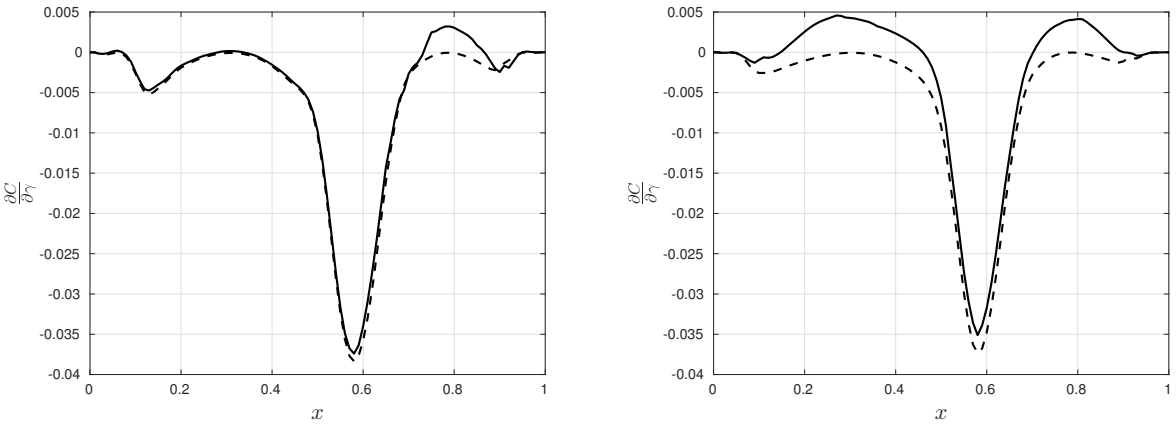
Table 1 and Table 2 provide comparisons of calculated sensitivities at randomly selected cells in the computational domain, using finite difference, full turbulence adjoint, and a frozen turbulence model. As can be seen, the estimated sensitivities agree well between the finite difference and the adjoint evaluations. As expected, utilizing a frozen turbulence assumption results in inconsistent sensitivities. Furthermore, the frozen turbulence assumption is seen to sometimes result in gradients having opposite sign, which would lead to anti-optimization steps! This effect is demonstrated in Figure 3 and Figure 4, which compare calculated sensitivity fields (with and without the frozen turbulence approximation), utilizing the k-ω and SA turbulence closure models, respectively. Although the sensitivities obtained with frozen turbulence and exact adjoint follow the same general trend, consistent evaluation clearly requires the inclusion of the turbulence model variables in the gradient evaluation. A detailed comparison can be further seen in Figure 5, where, the gradients along a line connecting the lower left corner to the upper right corner of the computational domain are presented in a one dimensional plot. Similar conclusions can be drawn from Figure 6, which depicts the sensitivities along the line (x,0.5).



(a) Discrete SA adjoint solver.

(b) Discrete SA adjoint solver with frozen turbulence.

Figure 4: Calculated sensitivity field of the power dissipation objective function C Equation 42 with respect to the design variable γ for the channel bend problem. All values are scaled with cell volumes.



(a) Solid line is full turbulence k-ω adjoint, dashed line is k-ω adjoint with frozen turbulence.

(b) Solid line is full turbulence SA adjoint, dashed line is SA adjoint with frozen turbulence.

Figure 5: Sensitivity $\frac{\partial C}{\partial \gamma}$ along a line connecting the lower left corner to the upper right corner in the channel bend problem.

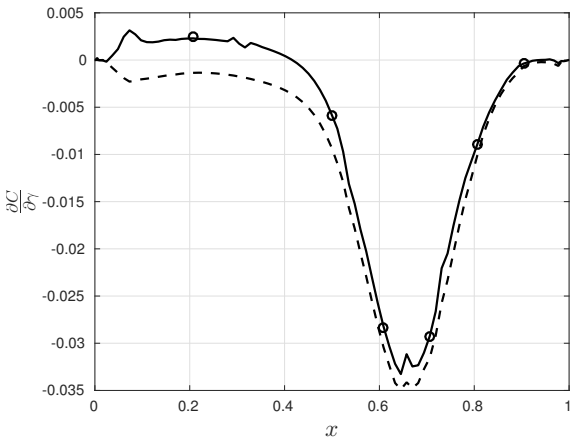


Figure 6: Sensitivity $\frac{\partial C}{\partial \gamma}$ along the line $(x, 0.5)$ in the channel bend problem, showing the effect of neglecting the density dependence on the distance function in the sensitivity analysis. The solid line represents the full turbulence SA adjoint, the dashed line represents frozen turbulence, markers - finite difference calculations.

The distance function in the SA turbulence model accounts for changes in the topology of the design by penalization techniques. However, in the adjoint analysis this dependence is neglected. As demonstrated in Figure 6, the obtained gradients agree very well with those which have taken into account changes in the topology and been computed by finite difference techniques. As the present work demonstrates optimization using exact gradients for the k-ω turbulence model, further investigations on the effect of this simplifying assumption in the SA model are left for future studies.

6. Topology optimization of turbulent flow examples

In this section the proposed sensitivity adjoint analysis will be utilized directly within several topology optimization examples. The first will consider detailed comparisons between designs obtained under the frozen turbulence assumption and with numerically exact gradients. The effect of different parameters will likewise be studied in detail. Finally, comparison to baseline designs will demonstrate the advantage of topology optimized solutions. The final examples, involving 2D and 3D manifold designs, will demonstrate the applicability of the presented techniques to more complex, and computationally intensive, problems.

6.1. 2D U-bend

The first example considers the optimal configuration of a 2D U-bend, utilizing both k-ω and SA turbulence models. The objective is given by Equation 42, i.e. the aim is to minimize the energy dissipation in a 2D channel. The effect of the frozen turbulence assumption on the final design will be investigated by comparing against the full-turbulence (numerically exact gradients) solutions. Moreover, the obtained designs will be further assessed by comparing the performance of body-fitted meshes and a baseline U-bend channel.

U_b [m/s]	H [m]	ν [m^2/s]
2	0.1	4.0×10^{-5}

Table 3: Flow properties for the 2D U-bend problem.

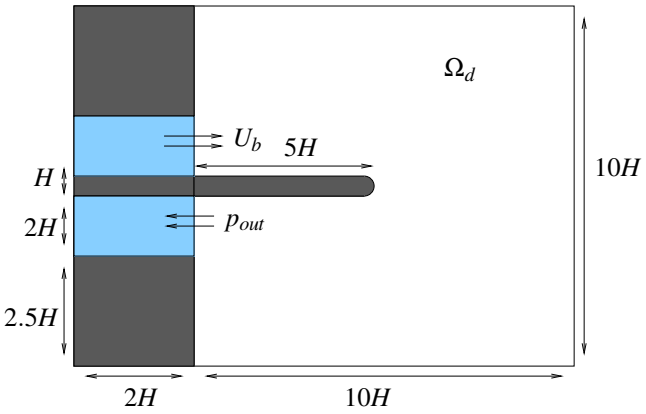


Figure 7: Schematic illustration of the 2D U-bend problem. Here Ω_d specifies the design domain. Blue and grey colors represent fixed fluid and solid regions, respectively. At the inlet, fully developed turbulent channel flow profiles are imposed for velocity and turbulent model variables. The geometry is scaled with the half inlet channel height H.

The problem setup and domain boundaries can be seen in the definition sketch, Figure 7, where, the geometry is scaled with the half-inlet channel height H. The Reynolds number in the considered optimization problem is $Re = U_b H/\nu = 5 \times 10^3$, where U_b is again the bulk velocity. Table 3 lists the properties utilized in the simulation. The considered optimization parameters are likewise shown in Table 4.

λ [s^{-1}]	q	r	β	f
2×10^3	0.1	0.01	$1.5 - 14$	0.30

Table 4: Parameter settings in the topology optimization of the 2D U-bend problem.

The total number of optimization cycles in the simulation is set to 500. During the optimization cycle, the projection parameter β is increased by a factor 1.5, starting from $\beta = 1.5$, after every 50 iterations. During the last 100 iterations, it is fixed at $\beta = 14$ which results in a relatively sharp interface between solid and fluid regions. The convergence history of the optimization process with the k-ω turbulence model is shown in Figure 8. Once the volume constraint is satisfied, the objective value starts decreasing monotonically.

The obtained topologies with both k-ω and SA turbulence models, including comparison of designs obtained under a frozen turbulence assumption, are shown in Figure 9 and Figure 10. Even though the frozen turbulence assumption results in inconsistent sensitivities, the objective functions C for the obtained designs are very close to those obtained using exact sensitivities, in this example. This is due to the confinement of the losses in the immediate vicinity of the solid/fluid boundary and to the similar length of these inter-phase boundaries. For faster flows and more confined boundary layers it might be necessary to use logarithmic transformation of the objective in order to reflect the nature of the problem on the design topology. In the current setup, a smooth curved topology is necessary to decrease the pressure difference between the entrance and outlet sections. This appears due to high inertial effects, and is clearly observed in the designs obtained with consistent sensitivities.

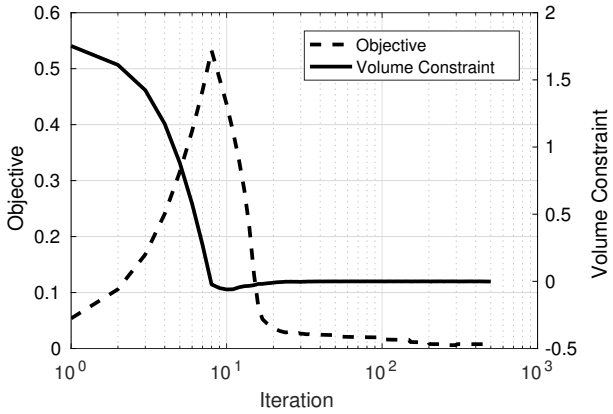
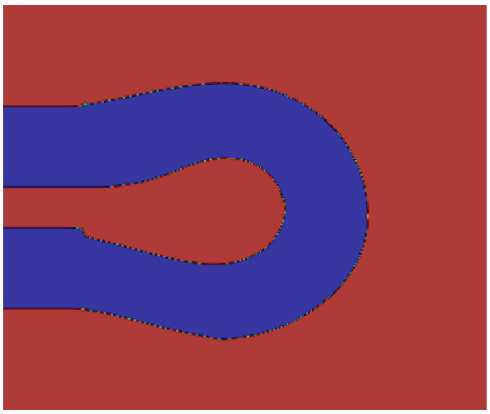
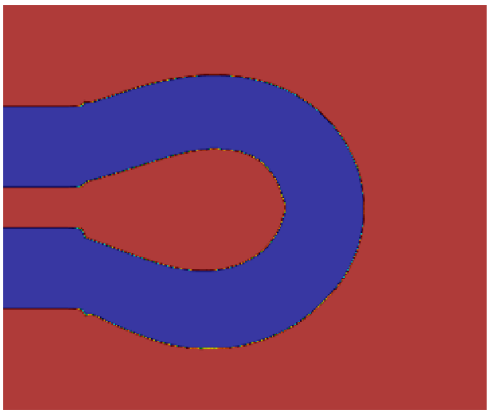


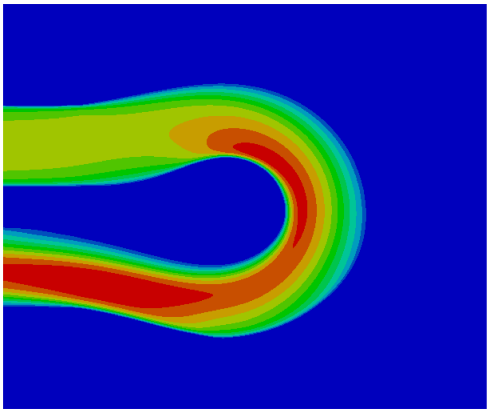
Figure 8: Convergence of the constrained optimization problem involving the 2D U-bend.



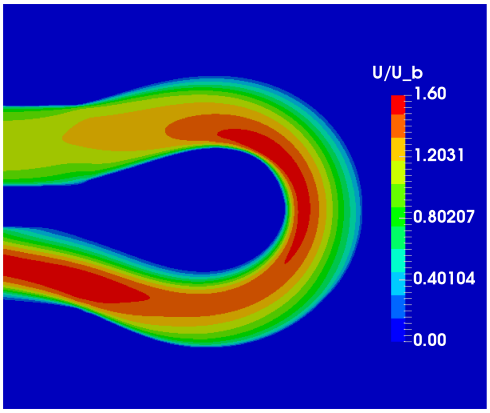
(a) k-ω model,$C = 0.00535$, full turbulence adjoint



(b) k-ω model,$C = 0.00558$, frozen turbulence adjoint

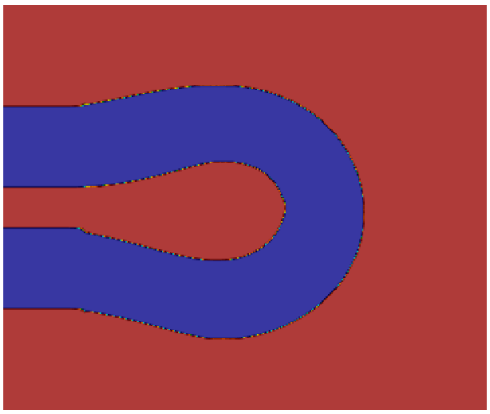


(c) Scaled velocity magnitude contour, full turbulence k-ω design

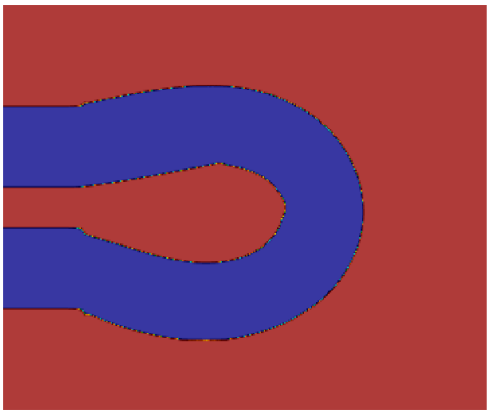


(d) Scaled velocity magnitude contour, frozen turbulence k-ω design

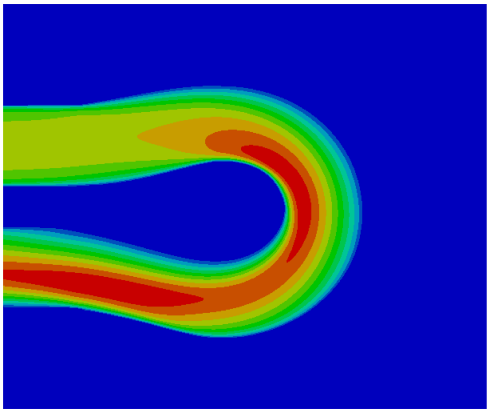
Figure 9: Topology optimization of the 2D U-bend problem for $Re = 5000$ with k-ω model. The presented velocity contours are non-dimensionalized with the bulk velocity. C is the objective function given by Equation 42 and computed for the final designs.



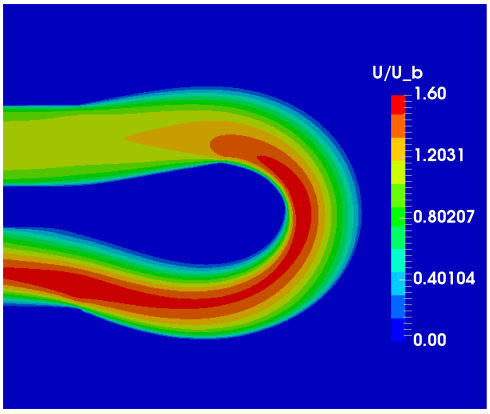
(a) SA model,$C = 0.009891$, full turbulence adjoint



(b) SA model,$C = 0.01016$, frozen turbulence adjoint



(c) Scaled velocity magnitude contour, full turbulence SA design



(d) Scaled velocity magnitude contour, frozen turbulence SA design

Figure 10: Topology optimization of the 2D U-bend problem for $Re = 5000$ with the SA model. Presented velocity contours are non-dimensionalized with the bulk velocity. C is the objective function for the optimized designs.

A cross check demonstrating the optimality of the obtained full turbulence designs is presented in Table 5. This demonstrates that the turbulence model is reflected exactly in the optimization process, and further improvement of the design performance can, in principal, only be obtained by further improving the ability of the models to describe the physical flow within the system.

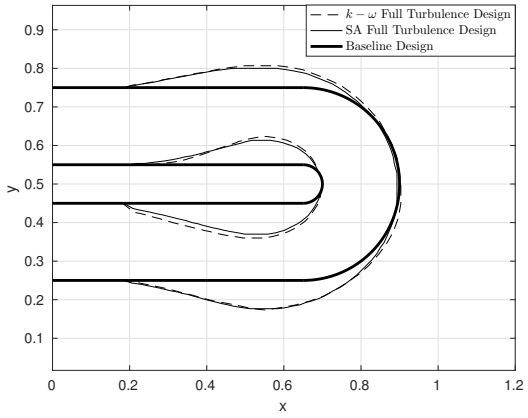
	k-ω Design	SA design
k-ω Sim.	0.00535394	0.00615163
SA Sim.	0.0100847	0.00989088

Table 5: Cross check of the optimized designs from the k-ω and SA turbulence models for the 2D U-bend problem. The blue color indicates that a specific design outperforms all others for the corresponding physical models utilized in the optimization.

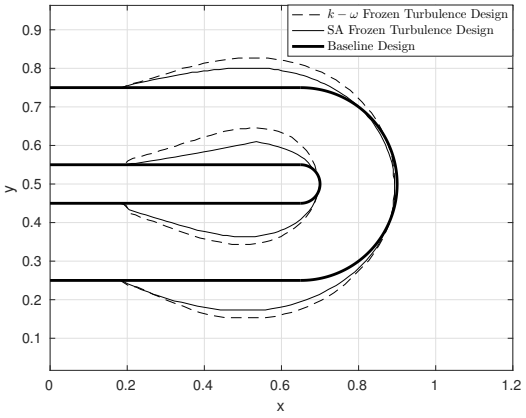
The performance gain of the optimized designs with respect to a reference design, both using a body fitted mesh, is demonstrated in Table 6 and Table 7. The body fitted profiles, shown in Figure 11, are obtained using the threshold $\eta = 0.5$. The inlet and the outlet in the simulations have been extended sufficiently in order to allow for fully-developed turbulent profiles to enter in the design domain, and to avoid back flow in the computational domain. Power dissipation objective values are calculated only in the

design domain, corresponding to the section of the optimized pipes given in Figure 11. Unstructured meshes are utilized in the simulations. For the boundary layers the distance (in dimensionless wall units) from the first cell center to the wall is kept well below unity.

The estimated performances clearly demonstrate the advantage of utilizing consistent sensitivity analysis in the optimization process. Designs obtained under a frozen turbulence assumption possess sharper corners and connections. Although, all optimized designs exhibit significant improvements over the baseline, in both tables the (two-equation) k-ω full turbulence design achieves the largest improvement (in percentage gain over the baseline). The worst of the optimized designs is that obtained with (one-equation) SA model using a frozen turbulence assumption.



(a) Optimized U-bend profiles of k-ω and SA models.



(b) Optimized U-bend profiles of k-ω and SA models with frozen turbulence assumption.

Figure 11: Comparison of topology optimized 2D U-bend profiles to the standard baseline model for $Re = 5000$. The body fitted meshes are constructed from the presented curved boundaries.

Design performance with k-ω model	Objective value	Percentage gain
k-ω Full turb. design	2.222244×10^{-3}	%66.03
k-ω Frozen turb. design	2.352015×10^{-3}	%64.05
SA Full turb. design	2.567404×10^{-3}	%60.76
SA Frozen turb. design	3.551443×10^{-3}	%45.72
Base design	6.543049×10^{-3}	–

Table 6: Performance comparison in the 2D U-bend problem of different optimized designs to the baseline model. The k-ω turbulence model and body fitted mesh are employed for simulating the responses.

Design performance with SA model	Objective value	Percentage gain
k-ω Full turb. design	1.987655×10^{-3}	%71.16
k-ω Frozen turb. design	2.101068×10^{-3}	%69.52
SA Full turb. design	2.007504×10^{-3}	%70.87
SA Frozen turb. design	2.277851×10^{-3}	%66.95
Base design	6.893569×10^{-3}	–

Table 7: Performance comparison in the 2D U-bend problem of different optimized designs to the baseline model. The SA turbulence model and body fitted mesh are employed for simulating the responses.

As expected (see again Table 6 and Table 7), the objective values calculated on the body fitted meshes are

found to be smaller than those computed with Brinkman penalization. The main reasons for this difference are the existence of the gray transition regions between the solid/fluid phases and the mesh resolution utilized for the optimization, which is $y^+ \approx 10$. For further investigation, the converge of the penalized model to the body fitted one for increasing penalization is demonstrated in Figure 12. The simulations are performed with the k-ω turbulence model and the penalization coefficients are obtained with increasing projection level, which brings the topology optimized solution near to a crisp 1-0 design. The convergence trend can be clearly observed and the predicted power dissipation agrees on the first significant digit to the body-fitted result for a near crisp 1-0 design. It should be stressed that for sharper transitions a refinement of the mesh around the transition region would be necessary in order to have a further agreement with the body fitted model. However, it is not presently computationally feasible to refine the optimization mesh to use a resolution of $y^+ \approx 1$.

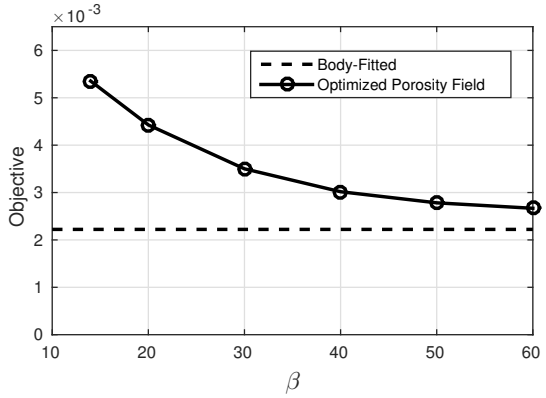
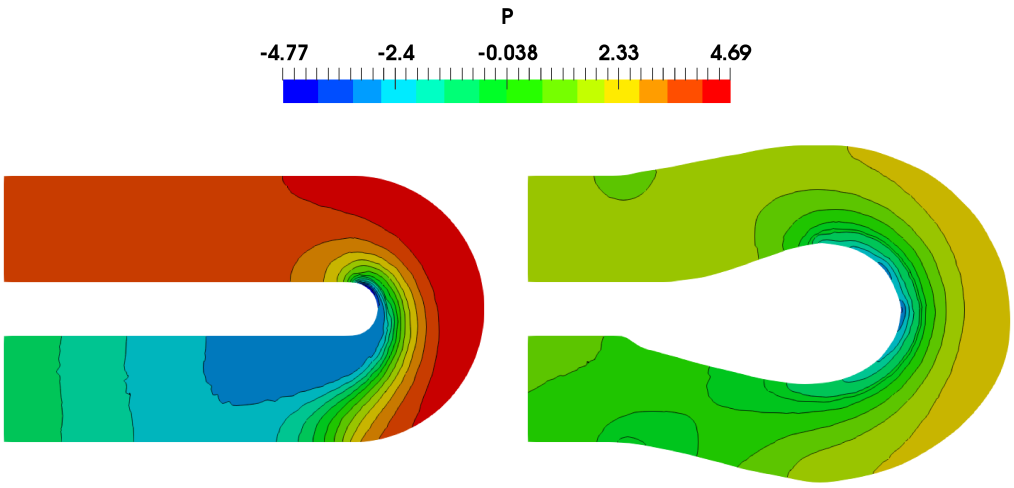


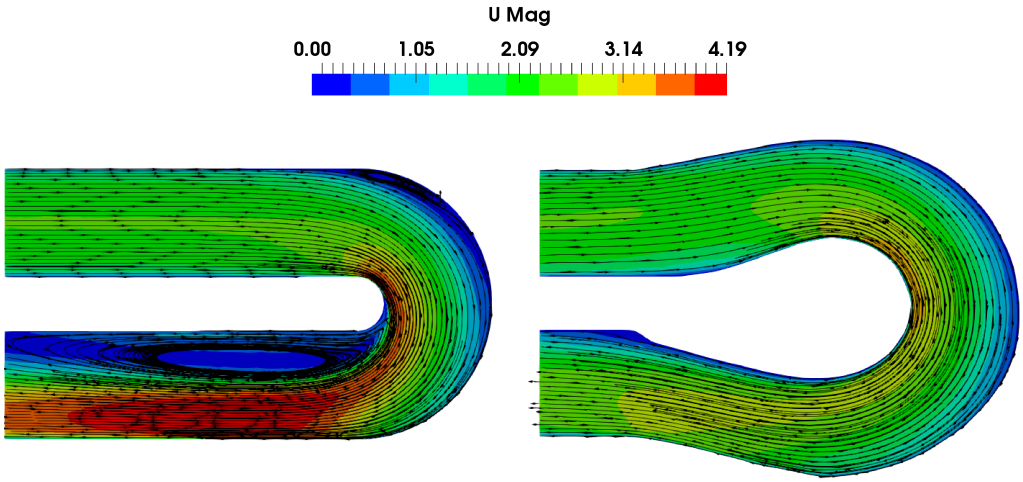
Figure 12: Objective values for the 2D U-bend problem computed for different values of the projection parameter β.



(a) Baseline design, pressure contour p $[kg/ms^2]$

(b) Optimized design with k-ω model, pressure contour p $[kg/ms^2]$

Figure 13: Optimized channel on a body fitted mesh for $Re = 5000$, showing the pressure field in comparison with that from the baseline model. The simulations are performed with the k-ω turbulence model.



(a) Baseline design, velocity magnitude u $[m/s]$ (b) Optimized design with k-ω model, velocity magnitude u [m/s]

Figure 14: Optimized channel on a body fitted mesh for $Re = 5000$, showing the velocity magnitude in comparison to that from the baseline model. This figure highlights that the main separation regions which occur in the baseline model are largely prevented in the optimized design. The simulations are performed using k-ω turbulence model.

Finally, comparisons of forward solutions of the optimized design with the k-ω model and of the baseline design are presented in Figure 13 and Figure 14. The pressure distribution demonstrates that the pressure difference between the inlet and the outlet is greatly reduced for the optimized case. The reason for the high performance gain can be explained by closely examining the flow separation presented in Figure 14, which is the main source of power dissipation.

6.2. 2D flow manifold

The second example considers the optimization of a 2D flow manifold. The objective is to minimize the power dissipation calculated using Equation 42 and restrict the mass flux at the outlets. The setup is shown in Figure 15. The k-ω turbulence model is utilized in the computations. The parameters utilized in the simulations and the optimization process are shown in Table 8 and Table 9. The Reynolds number is $Re = 3500$ and is again defined using the half inlet channel height H and the bulk velocity U_b. Fully developed turbulence profiles are set for the stream-wise velocity, turbulent kinetic energy k and the specific dissipation rate ω at the inlet. During the optimization process, the projection parameter β is increased in the same manner as in subsection 6.1.

U_b [m/s]	H [m]	ν [m^2/s]
2	0.1	5.7×10^{-5}

Table 8: Flow properties used in the 2D flow manifold problem.

λ [s^{-1}]	q	r	β	f
2×10^3	0.1	0.028	$1.5 - 14$	0.43

Table 9: Parameter settings in the topology optimization of the 2D flow manifold problem.

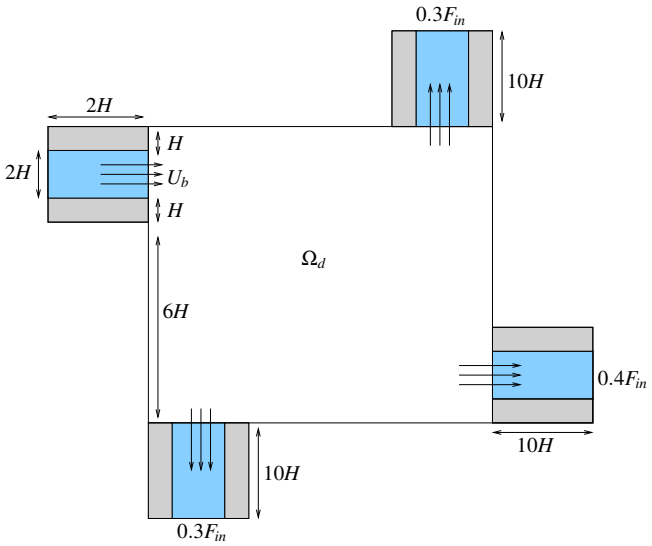


Figure 15: Schematic illustration of the 2D flow manifold problem. Here Ω_d specifies the design domain. Blue and gray colors are fixed fluid and solid regions, respectively. Fully developed turbulent channel flow profiles are imposed for velocity and turbulence model variables at the inlet. The geometry is scaled with the half inlet channel height H.

The optimized topology is shown in Figure 16. The resulting velocity and the kinetic energy distribution are shown in Figure 17. The diameter of the inner flow channels is smaller than the outlets. The optimization procedure links the inner network of channels to the outlets by minimizing the power dissipation and distributing material only in the design domain. The geometry of the outlets is fixed using Brinkman penalization, which results in rough jumps at the connection between the inner channels and the outlets. In order to ensure smooth transition a prescribed set of boundary conditions are enforced on the topology during the PDE filtering step [53]. These do not require any changes in the optimization setup, and can provide computationally cheap enforcement of topological features, accounting for physical characteristics, design, and manufacturing constraints not taken into account by the model.

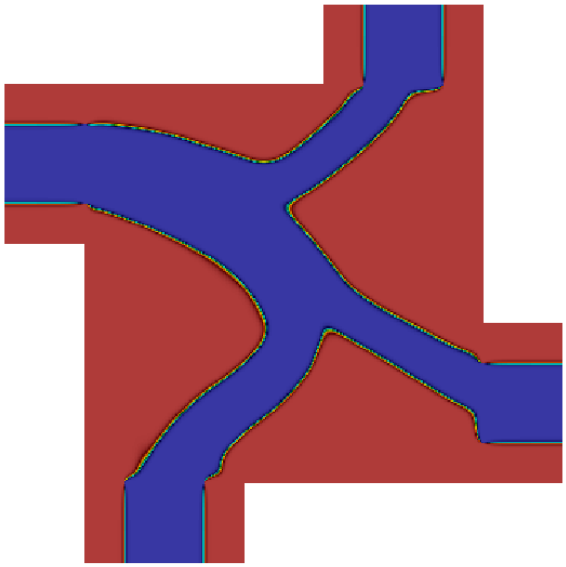
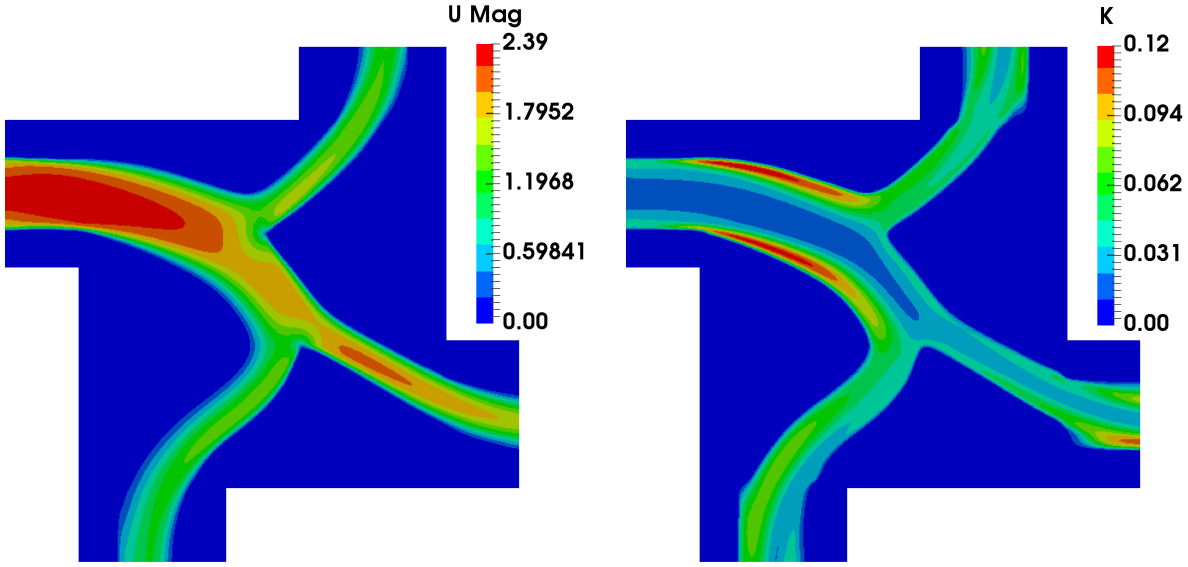


Figure 16: Topology optimization of a 2D flow manifold problem for $Re = 3500$ using the k-ω turbulence model. The objective value of the end design corresponds to $C = 2.3 \times 10^{-3}$.



(a) Velocity magnitude u [m/s] (b) Turbulent kinetic energy k [m^2/s^2]

Figure 17: Primal solutions on the topology optimization of a 2D flow manifold problem for $Re = 3500$ with the k-ω model.

6.3. 3D flow manifold

The final example considers topology optimization of a 3D manifold, as depicted in Figure 18. The aim of this example is to demonstrate the applicability of the proposed techniques on a real 3D large-scale problem. The k-ω turbulence model is utilized in the optimization. The computational domain is scaled with the input pipe radius H. The inlet length is $4H$ and the outlets are $8H$. The design domain is a cube with dimensions $10H \times 10H \times 10H$. A fully developed turbulent pipe flow profile is mapped at the inlet boundary for the stream-wise velocity and turbulence model variables. The Reynolds number considered corresponds to $Re = U_b H/\nu = 3500$, where U_b is the bulk inlet velocity. Flow properties corresponding to the turbulent pipe inlet are presented in Table 10. The optimization parameters are likewise specified in Table 11. The optimized design is shown in Figure 19.

The problem is discretized using 1.3M cells, corresponding to 16M DOFs for the forward and the adjoint solutions. Every optimization step requires one forward solution and four adjoint solutions (one for the objective function and three for the constraints). The optimized design, presented in Figure 19, is obtained overnight after 200 iteration on 400 CPU cores (Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz). The projection parameter β is multiplied by 1.5 every 20 iteration with a maximum value of 8. The design domain at the initial optimization step is filled with fluid only.

U_b [m/s]	H [m]	ν [m^2/s]
2	0.1	5.7×10^{-5}

Table 10: Flow properties for the 3D flow manifold problem.

λ [s^{-1}]	q	r	β	f
2×10^3	0.1	0.028	$1.5 - 8$	0.12

Table 11: Parameter settings in the topology optimization of 3D flow manifold problem.

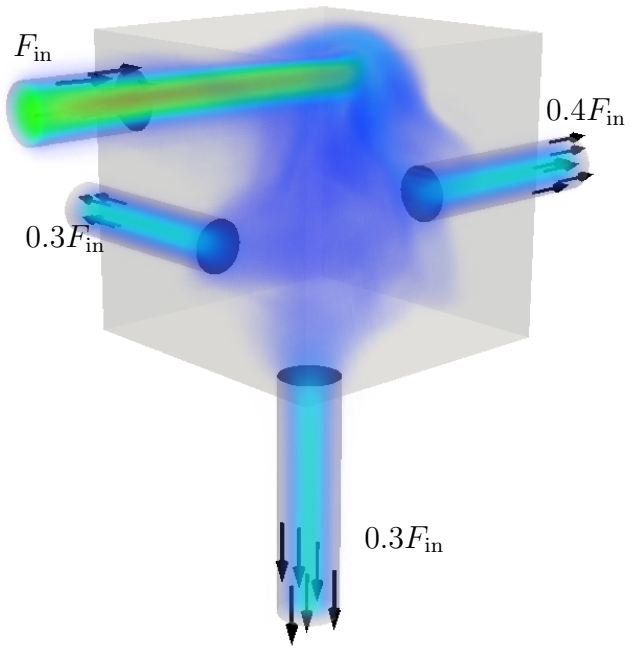


Figure 18: Computational domain for the 3D flow manifold problem. Constrained outlets are shown with the specified mass flow rates. The visible box corresponds to the design domain, whereas pipes are fixed throughout the optimization.

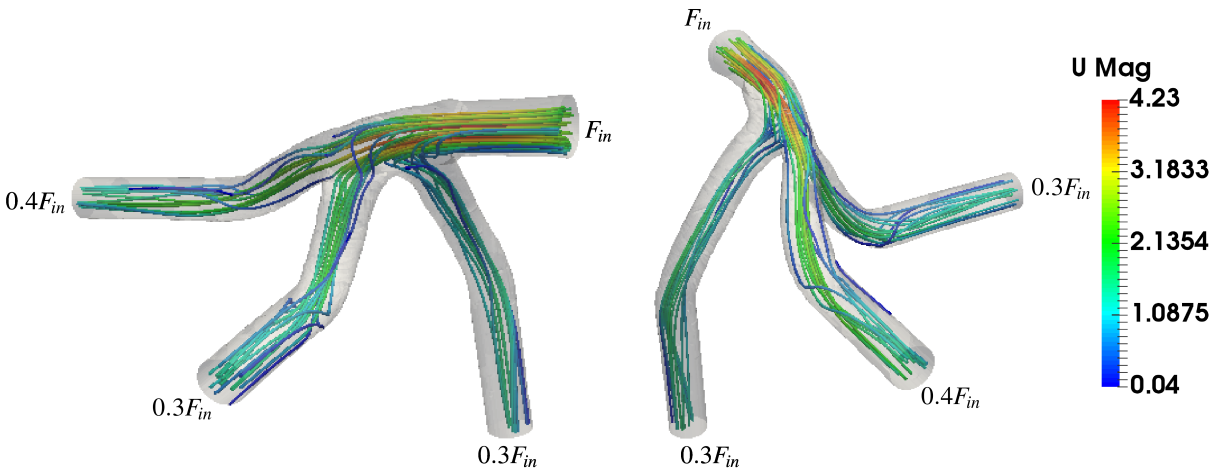


Figure 19: Iso-views of the optimized 3D flow manifold for $Re = 3500$ with the k-ω turbulence model. Streamlines are colored based on the velocity magnitude m/s. The objective value for the end design is $C = 0.83$. Constrained mass flux fractions are shown on outlets.

Similar to other large scale problems reported in the literature [41, 46, 47], most of the computational time is spent in solving the forward and the adjoint problems. The forward problem is solved iteratively using the SIMPLE algorithm. During the first several optimization iterations the topology changes significantly. Once the rough shape of the design is fixed during the initial optimization steps, the following designs are obtained by small incremental changes, mainly on the boundaries of the fluid/solid interface. These incremental changes lead to small changes of the fluid velocity and pressure distribution fields, thus requiring significantly fewer iterations. The same behavior can be observed in the adjoint solution. Thus, the computational cost

per optimization step, initially requiring 500-800 iterations for solving the forward and adjoint problems, decreases significantly towards the end of the optimization process, resulting in 25 to 50 forward and adjoint iterations. The actual iteration time is often lower than the time required for assembling the problem. These numbers demonstrate some of the advantages of the proposed schemes in terms of computational time. Furthermore, the proposed adjoint and forward algorithms are scalable. The only serial step is the construction (the action on a vector) of the Schur complement matrix given by Equation 35. Thus, larger problems can be easily optimized, provided that the implementation addresses efficiently the construction of the adjoint preconditioner.

7. Conclusions

The article demonstrates the application of Automatic Differentiation (AD) for obtaining exact sensitivities of 2D and 3D large scale turbulent flow topology optimization problems. Easy accommodation of new physics and turbulence closure models is achieved with minimal implementation effort. The developed methodology demonstrates a solution of the complex bookkeeping problem for coupled problems. The proposed adjoint solution provides a scalable and computationally cheap procedure for gradient analysis. It is shown that exact discrete gradients can differ significantly from those computed under a simplifying frozen turbulence assumption, confirming similar observations in the literature [16]. Optimized results are obtained using two different models for turbulence closure, corresponding to the one-equation Spalart-Allmeras model and the two-equation k-ω model. The optimized designs without any simplifying assumptions in the derivation of discrete adjoints outperform those optimized under a frozen turbulence assumption.

It should be pointed out that improved preconditioners are necessary for more robust behavior and even lower computational cost. Future research is required to demonstrate the applicability of the methodology to coupled fluid-heat transfer problems and more complex multiphysics.

8. Acknowledgments

The authors acknowledge the financial support received from the TopTen project sponsored by the Danish Council for Independent Research (DFF-4005-00320).

Appendix A. Discretization overview

The solution of the state equations is obtained by finite volume discretization on a non-uniform grid. The discretized Reynolds-averaged Navier-Stokes equations are solved using a segregated approach for the pressure-velocity coupling, with the help of the SIMPLE (semi-implicit method for pressure-linked equations) algorithm [20]. The SIMPLE method and its variants are widely used in computational fluid dynamics due to their low memory requirements and the ability to simulate both steady and unsteady flows. The implementation is based on the PETSc library [54, 55], which is utilized mainly for its efficient parallel sparse solvers.

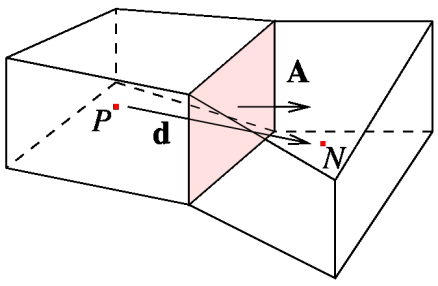


Figure A.20: General representation of a finite volume cell used in the discretization.

The discretization procedure is demonstrated for a standard transport equation. All variables are stored in cell centers (collocated grid arrangement). The fluxes are derived through a linearization procedure, and are stored in cell face centers. Figure A.20 illustrates two neighboring cells, where **d** is the vector connecting the cell centers P and N. **A** is the normal area vector of the common face f, pointing outwards from the current cell P. A generic scalar transport equation for the scalar variable ϕ is given below in vector notation, where the discretization of all spatial terms is presented.

$$\underbrace{\nabla\cdot(\mathbf{u}\phi)}_{\text{convective term}} = \underbrace{\nabla\cdot(\Gamma\nabla\phi)}_{\text{diffusion term}} + \underbrace{S_\phi}_{\text{source term}} \tag{A.1}$$

After integrating over a cell, and applying the Gauss divergence theorem, the following discrete form of the above transport equation is obtained:

$$\underbrace{\sum_f F\phi_f}_{\text{convective term}} = \underbrace{\sum_f (\Gamma\nabla\phi)_f\cdot\mathbf{A}}_{\text{diffusion term}} + \underbrace{S_\phi\Delta V}_{\text{source term}} \tag{A.2}$$

where surface integrals are realized as a sum over all faces of a cell, volume integrals are approximated as $\int_V dV = \Delta V$, and a new variable, corresponding to the face flux F, is introduced and lagged by an iteration

$$\sum_f (\mathbf{u}^{l-1}\cdot\mathbf{A})_f\ \phi_f = \sum_f F\phi_f \tag{A.3}$$

The calculation of F is based on weighted linear interpolation of nodal velocities **u** between two neighboring cell centers (P and N in Figure A.20). Although linear interpolation is second-order accurate [56], it is not bounded. Hence, using the same strategy for the calculation of ϕ_f results in non-physical oscillatory solutions for convection dominated flows. In order to ensure oscillation-free solutions, while still preserving second order accuracy, TVD (total variation diminishing) schemes are employed here. The convective term is written as [57]

$$F\phi_f = F\left[\phi_U + \frac{\psi(r)}{2}\left(\phi_D - \phi_U\right)\right] \tag{A.4}$$

where r is the ratio of the upwind to the downwind gradient, $\psi(r)$ is a limiter function that resides in the TVD monotonicity region, and subscripts U and D represent upwind and downwind cells, respectively. Selection of these cells is done based on the direction of the flow (the sign of the face flux F) between two neighboring cells (P and N in Figure A.20). The present work adopts a modified ratio r for unstructured grids [58]

$$r = \left[\frac{2\nabla\phi_P\cdot\mathbf{d}}{\phi_D - \phi_U} - 1\right] \tag{A.5}$$

and the SUPERBEE limiter [59] is utilized for the function $\psi(r)$ which reads

$$\psi(r) = max\left[0, min\left[1, 2r\right], min\left[2, r\right]\right] \tag{A.6}$$

It should be noted that, to improve numerical stability, the first term on the right hand side of the Equation A.4 is treated implicitly, while the second term is calculated explicitly and added to the source in a deferred correction manner.

The calculation of the diffusive fluxes needs attention due the non-orthogonality of the cells. They are corrected with the help of the over-relaxed approach proposed by [60] to ensure second-order accuracy.

$$\Gamma\nabla\phi_f\cdot\mathbf{A} = \underbrace{\Gamma\,|\mathbf{A}_d|\frac{\phi_N - \phi_P}{|\mathbf{d}|}}_{\text{orthogonal}} + \underbrace{\Gamma\mathbf{A}_r\cdot\nabla\phi_f}_{\text{non-orthogonal}} \tag{A.7}$$

The orthogonal part is treated implicitly, whereas the non-orthogonal part is calculated explicitly from the previous iteration as a correction and added to the source. $\mathbf{A}_d$ represents the parallel part of $\mathbf{A}$ to $\mathbf{d}$ which is given as

$$\mathbf{A}_d = \frac{\mathbf{d}}{\mathbf{d}\cdot\mathbf{A}}|\mathbf{A}|^2 \tag{A.8}$$

The non-orthogonal part $\mathbf{A}_r$ is defined as the difference between the area vector and the orthogonal contribution $\mathbf{A}_r = \mathbf{A} - \mathbf{A}_d$. The explicit calculation of $\nabla\phi_f$ in the non-orthogonal correction is done by first approximating the gradient at the cell center through the Gauss theorem as

$$\nabla\phi_P = \frac{\sum_f \phi_f \mathbf{A}}{\Delta V} \tag{A.9}$$

The resulting cell center gradients are then interpolated to the cell faces through weighted linear interpolation.

The final discrete form of the transport equation can be written as

$$A_P\phi_P = \sum_N A_N\phi_N + S_\phi \tag{A.10}$$

The implementation details for momentum, pressure correction, and turbulence model equations are presented in Appendix B.

Appendix B. Implementation details

The steady state incompressible RANS equations read

$$\nabla\cdot(\mathbf{u}\otimes\mathbf{u}) = \nabla\cdot\mathbf{T}_\nu - \frac{1}{\rho}\nabla p + \nabla\cdot\mathbf{T}_t \tag{B.1}$$

where the viscous $\mathbf{T}_\nu$ and Reynolds $\mathbf{T}_t$ stress tensors are

$$\mathbf{T}_\nu = 2\nu\mathbf{S}, \qquad \mathbf{T}_t = 2\nu_t\mathbf{S} - \frac{2}{3}k\mathbf{I}, \qquad \mathbf{S} = \frac{1}{2}\left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right) \tag{B.2}$$

The stress terms are combined, yielding an effective viscosity $\nu_{eff} = \nu + \nu_t$, as

$$\mathbf{T} = 2\nu_{eff}\mathbf{S} - \frac{2}{3}k\mathbf{I} = \nu_{eff}\left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right) - \frac{2}{3}k\mathbf{I} \tag{B.3}$$

Consequently, the total diffusion term in the RANS equations is treated as

$$\nabla\cdot\mathbf{T} = \underbrace{\nabla\cdot\nu_{eff}\nabla\mathbf{u}}_{\text{implicit}} + \underbrace{\nabla\cdot\nu_{eff}\nabla\mathbf{u}^T - \nabla\cdot\left(\frac{2}{3}k\mathbf{I}\right)}_{\text{explicit}} \tag{B.4}$$

Utilizing the above procedure gives rise to the following discrete source contributions, due to both Reynolds stresses and pressure gradients:

$$\mathbf{b}_P = \sum_f \left(\nu_{eff}\nabla\mathbf{u}^T\right)_f\cdot\mathbf{A} - \nabla\cdot\left(\frac{2}{3}k\mathbf{I}\right)\Delta V - \sum_f p_f\mathbf{A} \tag{B.5}$$

The discretization techniques given in section Appendix A can now be applied to the momentum equations to realize algebraic linear equations having the generic form:

$$A_P\mathbf{u}_P = \sum_N A_N\mathbf{u}_N + \mathbf{S}_u, \qquad \mathbf{S}_u = \mathbf{S}_P + \mathbf{b}_P \tag{B.6}$$

Here $\mathbf{S}_P$ represents the source contributions, arising from the discretization of convective and diffusive terms (explicitly treated terms in the Equations A.4 and A.7). In order to improve the convergence properties and the diagonal dominance of the system matrix, implicit under-relaxation is applied to the solution of the momentum equations as

$$\frac{A_P}{\alpha_u}\mathbf{u}_P = \sum_N A_N \mathbf{u}_N + \mathbf{S}_u, \qquad \mathbf{S}_u = \mathbf{S}_P + \mathbf{b}_P + \left[(1-\alpha_u)\frac{A_P}{\alpha_u}\right]\mathbf{u}^{l-1} \tag{B.7}$$

The discretized form of the continuity equation, given in Equation 1, reads

$$\sum_f F = 0 \tag{B.8}$$

Due to the collocated storage of velocities and pressure at cell centers, Rhie-Chow interpolation [61] is applied on the face fluxes before deriving the pressure correction equation, which directly links the face fluxes to the driving pressure difference across faces.

$$F^* = \mathbf{u}_f \cdot \mathbf{A} - \left(\frac{\Delta V}{A_P}\right)_f \left(\frac{p_N - p_P}{|\mathbf{d}|} - \nabla p_f \cdot \mathbf{e}\right)|\mathbf{A}_d| \tag{B.9}$$

In the above $\mathbf{e} = \frac{\mathbf{d}}{|\mathbf{d}|}$ is the unit vector in the direction of the central vector. It should be stressed that F^* is based on the a solution of the momentum equations, and does not satisfy the continuity Equation B.8. Hence, it needs a correction to satisfy the divergence free condition. Thus, defining a flux correction in accordance with [62] results in the following discrete continuity equation

$$\sum_f (F^* + F') = 0 \tag{B.10}$$

where the flux correction F' is defined as [20]:

$$F' = -\left(\frac{\Delta V}{A_P}\right)_f \left(\frac{p'_N - p'_P}{|\mathbf{d}|}\right)|\mathbf{A}_d| \tag{B.11}$$

Substituting the flux correction F' into the discrete continuity equation, Equation B.10, yields the pressure correction equation with a source of mass imbalance, written here in generic form as

$$a_P p'_P = \sum_N a_N p'_N + \underbrace{S_P - \sum_f F^*}_{\text{Source}} \tag{B.12}$$

where S_P is the possible source contribution due to the diffusion discretization. The mass imbalance $\sum_f F^*$ approaches zero at convergence. With the help of the pressure correction p', nodal velocities and pressures are obtained as

$$\mathbf{u}_P = \mathbf{u}^*_P - \left(\frac{\Delta V}{A_P}\right)\nabla p'_P, \qquad p_P = p^*_P + \alpha_p p'_P \tag{B.13}$$

where α_p is the under-relaxation factor for pressure. Continuity is strictly enforced on the face fluxes F at each SIMPLE iteration following the solution of the pressure correction

$$F = F^* + F' \tag{B.14}$$

Considering the two-equation k-ω turbulence model, the terms on the right hand side of Equation B.15 require special attention

$$\nabla\cdot(\mathbf{u}\,k) - \nabla\cdot\left[\left(\nu + \sigma^*\frac{k}{\omega}\right)\nabla k\right] = \underbrace{\mathbf{T}_t{:}\nabla\mathbf{u}}_{\text{Production}} - \underbrace{\beta^*\omega k}_{\text{Dissipation}} \tag{B.15}$$

In order to preserve the positiveness of the solution and increase the numerical stability of RANS models, the positive terms (production) on the right hand side are treated explicitly, while the negative terms (dissipation) are treated implicitly. The total contribution to the central coefficient and source due to the production and dissipation terms in the k equation are given as

$$\int_V \mathbf{T}_t{:}\nabla\mathbf{u}\, dV = \underbrace{2\nu_t(\mathbf{S}{:}\nabla\mathbf{u})\Delta V - \frac{2}{3}(k\mathbf{I}{:}\nabla\mathbf{u})\Delta V}_{\text{explicit}} \tag{B.16}$$

$$\int_V -\beta^*\omega k\, dV = \underbrace{-\beta^*\omega k\Delta V}_{\text{implicit}} \tag{B.17}$$

$$S_k = 2\nu_t(\mathbf{S}{:}\nabla\mathbf{u})\Delta V - \frac{2}{3}(k\mathbf{I}{:}\nabla\mathbf{u})\Delta V, \quad a_{P(k)} = \beta^*\omega\Delta V \tag{B.18}$$

Similarly, the transport equation for ω is given as

$$\nabla\cdot(\mathbf{u}\,\omega) - \nabla\cdot\left[\left(\nu + \sigma\frac{k}{\omega}\right)\nabla\omega\right] = \underbrace{\frac{\alpha\omega}{k}\mathbf{T}_t{:}\nabla\mathbf{u}}_{\text{Production}} - \underbrace{\beta\omega^2}_{\text{Dissipation}} + \underbrace{\frac{\sigma_d}{\omega}\nabla k\cdot\nabla\omega}_{\text{Cross Diffusion}} \tag{B.19}$$

the production term is handled in the same manner as above, where positiveness of the cross diffusion term is ensured through the closure parameter σ_d. The dissipation term is discretized and linearized into the semi-implicit form $\beta\omega^l\omega^{l-1}\Delta V$. The term proportional to ω^l is added to the central coefficient.

The transport equation for the SA model is given as

$$\nabla\cdot(\mathbf{u}\,\tilde{\nu}) - \frac{1}{\sigma}\nabla\cdot[(\nu+\tilde{\nu})\nabla\tilde{\nu}] = \underbrace{c_{b1}\tilde{S}\tilde{\nu}}_{\text{Production}} \underbrace{- c_{w1}f_w\left[\frac{\tilde{\nu}}{y_w}\right]^2}_{\text{Dissipation}} + \underbrace{\frac{c_{b2}}{\sigma}\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}}_{\text{Transport}} \tag{B.20}$$

Contributions to the central coefficient come from the linearization and implicit treatment of the dissipation term, whereas production and transport terms are added to the source. The overall contribution reads

$$\int_V c_{b1}\tilde{S}\tilde{\nu}\, dV = \underbrace{c_{b1}\tilde{S}\tilde{\nu}\Delta V}_{\text{explicit}}, \quad \int_V c_{w1}f_w\left[\frac{\tilde{\nu}}{y_w}\right]^2 dV = \underbrace{c_{w1}f_w\left[\frac{\tilde{\nu}^{l-1}}{y_w^2}\right]\tilde{\nu}^l\Delta V}_{\text{implicit}} \tag{B.21}$$

$$\int_V \frac{c_{b2}}{\sigma}\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\, dV = \underbrace{\frac{c_{b2}}{\sigma}\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\Delta V}_{\text{explicit}} \tag{B.22}$$

$$S_\nu = c_{b1}\tilde{S}\tilde{\nu}\Delta V + \frac{c_{b2}}{\sigma}\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\Delta V, \quad a_{P\nu} = c_{w1}f_w\left[\frac{\tilde{\nu}}{y_w^2}\right]\Delta V \tag{B.23}$$

For the momentum equations and turbulence transport equations, residuals are checked against a prescribed tolerance ϵ as:

$$R_\phi = \left|\left|\mathbf{S}_\phi - \mathbf{M}\phi^{l-1}\right|\right|_\infty \leq \epsilon \tag{B.24}$$

where ϕ stands for any state variable, before solving the discrete system of equations. If the norm of the residual is smaller than the prescribed tolerance, then the solution from the previous iteration is considered to be sufficient for the discrete system defined by the newly updated coefficients. For the pressure correction p', mass imbalance is considered for the convergence criteria (the source term of the pressure correction) Equation B.12)

$$R_p = \left|\left|\sum_f F^*\right|\right|_\infty \leq \epsilon \tag{B.25}$$

Overall steps involved in the SIMPLE algorithm can be found in [20, 56, 57].

Appendix C. Validation of the flow solver

The focus in this Appendix is on the simulation of turbulent flow in an asymmetric plane diffuser (Buice-Eaton diffuser [63]) for validation of the developed RANS solver. The general setup is shown in Figure C.21. The dimensions are scaled with the channel height H. The computational domain does not contain any sharp edges around the starting and ending sections of the diffuser ramp. A detailed description of the geometry can be found on the website provided by NASA [64].

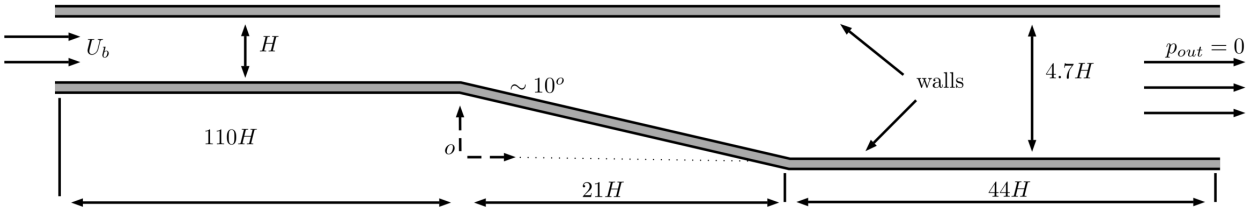


Figure C.21: Illustrative schematic for the simulation of the asymmetric plane diffuser. The origin of the coordinate system is marked as o in the figure.

U_b [m/s]	H [m]	ν [m^2/s]
19.812	0.015	1.695×10^{-5}

Table C.12: Properties used in the simulation of the asymmetric plane diffuser.

Table C.12 presents the utilized relevant parameters for defining the simulation. In the experiment made by [63], the Reynolds number is defined using the upstream channel center line velocity ($U_{cl} = 1.14U_b$) and channel height H, which results in $\mathrm{Re_{cl}} = 2 \times 10^4$. The computational grid consists of 66080 hexahedral cells. The mesh is clustered towards the walls such that the average value of the distance from the first cell center to the wall ensures that $y^+ \approx 1$ for both upper and lower walls.

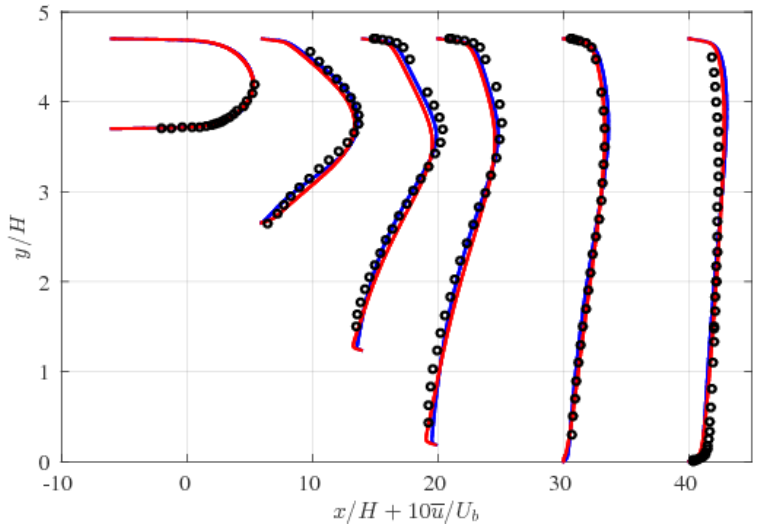


Figure C.22: Computed velocity profiles from the simulation of the asymmetric (Buice-Eaton) plane diffuser, with $Re_{cl} = 2 \times 10^4$. The blue solid lines indicate results from the k-ω model, the red solid lines indicate results from the Spalart-Allmaras model, and the circles correspond to the measurement from [63].

Both turbulence models are tested, and both perform well near the inlet of the diffuser (location $x/H = -6$ in Figure C.22), where a fully developed turbulent channel flow profile can be clearly observed. In the

diffuser ramp section (locations $x/H = 6, 14, 20$) flow separation occurs due to adverse pressure gradients. The k-ω model slightly outperforms the SA model, based on the measured mean velocity profiles. This result is expected, as the k-ω model is known to perform well in adverse pressure gradient flow cases. In the outlet (locations $x/H = 30$ and 40, both models predict similar velocity profiles, generally matching the measurements, though with small differences at location $x/H = 40$. Overall, the velocity profiles predicted with both models exhibit fine agreement with the experimental data at the presented locations of the diffuser.

References

[1] M. P. Bendsøe, O. Sigmund, Topology Optimization - Theory, Methods and Applications, Sparing Lag, Berlin Heidelberg, 2003.

[2] T. Borrvall, J. Petersson, Topology optimization of fluids in stokes flow, International Journal for Numerical Methods in Fluids 41 (1) (2003) 77–107. doi:10.1002/fld.426.

[3] N. Aage, T. H. Poulsen, A. Gersborg-Hansen, O. Sigmund, Topology optimization of large scale stokes flow problems, Structural and Multidisciplinary Optimization 35 (NA) (2008) 175–180. doi:10.1007/s00158-007-0128-0.

[4] A. Evgrafov, On chebyshev's method for topology optimization of stokes flows, Structural and Multidisciplinary Optimization 51 (4) (2015) 801–811. doi:10.1007/s00158-014-1176-x.

[5] A. Gersborg-Hansen, O. Sigmund, R. B. Haber, Topology optimization of channel flow problems 30 (2005) – 181–192.

[6] Y. Deng, Z. Liu, P. Zhang, Y. Liu, Y. Wu, Topology optimization of unsteady incompressible Navier-Stokes flows, Journal of Computational Physics 230 (17) (2011) 6688 – 6708. doi:10.1016/j.jcp.2011.05.004.

[7] S. Kreissl, G. Pingen, K. Maute, Topology optimization for unsteady flow, International Journal for Numerical Methods in Engineering 87 (13) (2011) 1229–1253. doi:10.1002/nme.3151.

[8] G. Pingen, A. Evgrafov, K. Maute, Topology optimization of flow domains using the lattice Boltzmann method, Structural and Multidisciplinary Optimization 34 (6) (2007) 507–524. doi:10.1007/s00158-007-0105-7.

[9] G. Pingen, A. Evgrafov, K. Maute, Adjoint parameter sensitivity analysis for the hydrodynamic lattice Boltzmann method with applications to design optimization, Computers & Fluids 38 (4) (2009) 910 – 923. doi:10.1016/j.compfluid.2008.10.002.

[10] S. Nørgaard, O. Sigmund, B. Lazarov, Topology optimization of unsteady flow problems using the lattice Boltzmann method, Journal of Computational Physics 307 (2016) 291 – 307. doi:10.1016/j.jcp.2015.12.023.

[11] P. Guillaume, K. Idris, Topological sensitivity and shape optimization for the stokes equations, Siam Journal on Control and Optimization 43 (1) (2004) 1–31. doi:10.1137/S0363012902411210.

[12] S. Amstutz, Topological sensitivity analysis for some nonlinear pde systems, Journal De Mathematiques Pures Et Appliquees 85 (4) (2006) 540–557. doi:10.1016/j.matpur.2005.10.008.

[13] L. F. N. Sa, R. C. R. Amigo, A. A. Novotny, E. C. N. Silva, Topological derivatives applied to fluid flow channel design optimization problems, Structural and Multidisciplinary Optimization 54 (2) (2016) 249–264. doi:10.1007/s00158-016-1399-0.

[14] C. Othmer, A continuous adjoint formulation for the computation of topological and surface sensitivities of ducted flows, International Journal for Numerical Methods in Fluids 58 (8) (2008) 861–877. doi:10.1002/fld.1770.

[15] M. Pietropaoli, R. Ahlfeld, F. Montomoli, A. Ciani, M. D'Ercole, Design for additive manufacturing: Internal channel optimization, Journal of Engineering for Gas Turbines and Power 139 (10) (2017) 102101–102101–8. doi:10.1115/1.4036358.

[16] A. S. Zymaris, D. I. Papadimitriou, K. C. Giannakoglou, C. Othmer, Continuous adjoint approach to the Spalart-Allmaras turbulence model for incompressible flows, Computers and Fluids 38 (8) (2009) 1528–1538. doi:10.1016/j.compfluid.2008.12.006.

[17] E. A. Kontoleontos, E. M. Papoutsis-Kiachagias, A. S. Zymaris, D. I. Papadimitriou, K. C. Giannakoglou, Adjoint-based constrained topology optimization for viscous flows, including heat transfer, Engineering Optimization 45 (8) (2013) 941–961. doi:10.1080/0305215X.2012.717074.

[18] A. Carnarius, F. Thiele, E. Ozkaya, N. R. Gauger, A. Carnarius, F. Thiele, E. Ozkaya, N. R. Gauger, Adjoint approaches for optimal flow control, 5th Flow Control Conference (2010) 2010–5088.

[19] A. Evgrafov, M. M. Gregersen, M. P. Srensen, Convergence of cell based finite volume discretizations for problems of control in the conduction coefficients, E S a I M: Mathematical Modelling and Numerical Analysis 45 (6) (2011) 1059–1080. doi:10.1051/m2an/2011012.

[20] S. V. Patankar, Numerical heat transfer and fluid flow, Series in computational methods in mechanics and thermal sciences, Hemisphere Pub. Corp. New York, Washington, 1980.

[21] R. Giering, T. Kaminski, Recipes for adjoint code construction, Acm Transactions on Mathematical Software 24 (4) (1998) 437–474. doi:10.1145/293686.293695.

[22] A. Griewank, Evaluating derivatives : Principles and techniques of algorithmic differentiation, SIAM, 2000.

[23] M. Auvinen, A modular framework for generation and maintenance of adjoint solvers assisted by algorithmic differentiation - with applications to an incompressible Navier-Stokes solver (2014).

[24] M. Towara, U. Naumann, A discrete adjoint model for OpenFOAM, Procedia Computer Science 18 (2013) 429–438. doi:10.1016/j.procs.2013.05.206.

[25] H. G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. 12 (6) (1998) 620–631. doi:10.1063/1.168744.

[26] P. E. Farrell, D. A. Ham, S. W. Funke, M. E. Rognes, Automated derivation of the adjoint of high-level transient finite element programs, Siam Journal on Scientific Computing 35 (4) (2013) C369–C393. doi:10.1137/120873558.
[27] P. E. Farrell, Topology optimisation of fluids in Stokes flow. Website, http://www.dolfin-adjoint.org/en/latest/documentation/stokes-topology/stokes-topology.html#topology-optimisation-of-fluids-in-stokes-flow.
[28] M. Alns, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. Rognes, G. Wells, The fenics project version 1.5, Archive of Numerical Software 3 (100). doi:10.11588/ans.2015.100.20553.
URL http://journals.ub.uni-heidelberg.de/index.php/ans/article/view/20553
[29] [link].
URL https://www.comsol.com
[30] A. Griewank, A. Walther, Algorithm 799: revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation, Acm Transactions on Mathematical Software 26 (1) (2000) 19–45. doi:10.1145/347837.347846.
[31] A. C. Marta, C. A. Mader, J. R. R. A. Martins, E. Van der Weide, J. J. Alonso, A methodology for the development of discrete adjoint solvers using automatic differentiation tools, International Journal of Computational Fluid Dynamics 21 (9-10) (2007) 307–327. doi:10.1080/10618560701678647.
[32] S. A. Nørgaard, M. Sagebaum, N. R. Gauger, B. S. Lazarov, Applications of automatic differentiation in topology optimization, Structural and Multidisciplinary Optimization (2017) 1–12doi:10.1007/s00158-017-1708-2.
[33] R. Roth, S. Ulbrich, A discrete adjoint approach for the optimization of unsteady turbulent flows, Flow Turbulence and Combustion, Flow 90 (4) (2013) 763–783. doi:10.1007/s10494-012-9439-3.
[34] D. Wilcox, Formulation of the k-ω Turbulence Model Revisited, AIAA Journal 46 (2008) 2823–2838. doi:10.2514/1.36541.
[35] P. Spalart, S. Allmaras, A one equation turbulence model for aerodynamic flows, Recherche Aerospatiale (AIAA-92-0439).
[36] F. R. Menter, Two-equation eddy-viscosity turbulence models for engineering applications, Aiaa Journal 32 (1994) 1598–1605. doi:10.2514/3.12149.
[37] S. Allmaras, T. Forrester, P. Spalart, Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model, 7th International Conference on Computational Fluid Dynamics (ICCFD07).
[38] P. Tucker, Assessment of geometric multilevel convergence robustness and a wall distance method for flows with multiple internal boundaries, Applied Mathematical Modelling 22 (4 /5) (1998) 293–311. doi:10.1016/S0307-904X(98)10007-0.
[39] P. Tucker, C. Rumsey, P. Spalart, R. Bartels, R. Biedron, Computations of wall distances based on differential equations, Aiaa Journal 43 (3) (2005) 539–549.
[40] F. Wang, B. Lazarov, O. Sigmund, On projection methods, convergence and robust formulations in topology optimization, Structural and Multidisciplinary Optimization 43 (6) (2011) 767–784. doi:10.1007/s00158-010-0602-y.
[41] B. S. Lazarov, O. Sigmund, Filters in topology optimization based on Helmholtz-type differential equations, International Journal for Numerical Methods in Engineering 86 (6) (2011) 765–781. doi:10.1002/nme.3072.
[42] B. S. Lazarov, F. Wang, O. Sigmund, Length scale and manufacturability in density-based topology optimization, Archive of Applied Mechanics 86 (1) (2016) 189–218. doi:10.1007/s00419-015-1106-4.
[43] L. H. Olesen, F. Okkels, H. Bruus, A high-level programming-language implementation of topology optimization applied to steady-state Navier-Stokes flow, International Journal for Numerical Methods in Engineering 65 (7) (2006) 975–1001. doi:10.1002/nme.1468.
[44] K. Svanberg, The method of moving asymptotesa new method for structural optimization, International Journal for Numerical Methods in Engineering 24 (2) (1987) 359–373. doi:10.1002/nme.1620240207.
[45] K. Svanberg, A class of globally convergent optimization methods based on conservative convex separable approximations, Siam Journal on Optimization 12 (2) (2001) 555–573. doi:10.1137/S1052623499362822.
[46] N. Aage, B. S. Lazarov, Parallel framework for topology optimization using the method of moving asymptotes, Structural and Multidisciplinary Optimization 47 (4) (2013) 493–505.
[47] N. Aage, E. Andreassen, B. Lazarov, Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework, Structural and Multidisciplinary Optimization 51 (3) (2015) 565–572. doi:10.1007/s00158-014-1157-0.
[48] L. Hascoët, V. Pascual, The Tapenade Automatic Differentiation tool: Principles, Model, and Specification, ACM Transactions On Mathematical Software 39 (3). doi:10.1145/2450153.2450158.
[49] CoDiPack—code differentiation package, accessed: 2016-10-18 (2016).
URL http://www.scicomp.uni-kl.de/software/codi/
[50] R. J. Hogan, Fast reverse-mode automatic differentiation using expression templates in C++, ACM Transactions on Mathematical Software 40 (4) (2014) 26. doi:10.1145/2560359.
[51] F. Verdugo, W. A. Wall, Unified computational framework for the efficient solution of n-field coupled problems with monolithic schemes, Computer Methods in Applied Mechanics and Engineering 310 (2016) 335–366.
[52] M. Ur Rehman, C. Vuik, G. Segal, Preconditioners for the steady incompressible Navier-Stokes problemdoi:10.1.1.148.4788.
[53] B. S. Lazarov, M. Schevenels, O. Sigmund, Robust design of large-displacement compliant mechanisms, Mechanical Sciences 2 (2) (2011) 175–182. doi:10.5194/ms-2-175-2011.
[54] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, PETSc Web page, http://www.mcs.anl.gov/petsc (2016).
URL http://www.mcs.anl.gov/petsc
[55] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, PETSc users manual,

Tech. Rep. ANL-95/11 - Revision 3.7, Argonne National Laboratory (2016).
URL http://www.mcs.anl.gov/petsc

[56] J. H. Ferziger, M. Peric, Computational Methods for Fluid Dynamics, Springer Berlin Heidelberg, 2001.

[57] H. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, Pearson Education Limited, 2007.

[58] M. Darwish, F. Moukalled, Tvd schemes for unstructured grids, International Journal of Heat and Mass Transfer 46 (4) (2003) 599–611. doi:10.1016/S0017-9310(02)00330-7.

[59] P. L. Roe, Some contributions to the modelling of discontinuous flows, in: B. E. Engquist, S. Osher, R. C. J. Somerville (Eds.), Large-Scale Computations in Fluid Mechanics, 1985, pp. 163–193.

[60] H. Jasak, Error analysis and estimation for the finite volume method with applications to fluid flows, Ph.D. thesis, Imperial College, University of London (1996).

[61] C. M. Rhie, W. L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation, AIAA Journal 21 (1983) 1525–1532. doi:10.2514/3.8284.

[62] S. R. Mathur, J. Y. Murthy, Pressure boundary conditions for incompressible flow using unstructured meshes, Numerical Heat Transfer, Part B: Fundamentals 32 (3) (1997) 283–298.

[63] C. U. Buice, J. K. Eaton, Experimental investigation of flow through an asymmetric plane diffuser, Transactions of the ASME. Journal of Fluids Engineering 122 (2) (2000) 433–435.

[64] J. W. Slater, NPARC Alliance Validation Archive: Buice Diffuser. Website, NASA, http://www.grc.nasa.gov/WWW/wind/valid/buice/buice.html.
URL http://www.grc.nasa.gov/WWW/wind/valid/buice/buice.html