Robust preconditioned one-shot methods and direct-adjoint-looping for optimising Reynolds-averaged turbulent flows

Nabi, Saleh; Grover, Piyush; Caulfield, Colm-cille

TR2022-031 March 01, 2022

Abstract

We compare the performance of direct-adjoint-looping (DAL) and one-shot methods in a design optimization task involving turbulent flow modeled using Reynolds-Averaged-Navier-Stokes equations. Two preconditioned variants of the one-shot algorithm are proposed and tested. The role of an approximate Hessian as a preconditioner for the one-shot method iterations is highlighted. We find that the preconditioned one-shot methods can solve the PDE-constrained optimization problem with the cost of computation comparable (about fourfold) to that of the simulation run alone. This cost is substantially less than that of DAL, which requires O(10) direct-adjoint loops to converge. The optimization results arising from the one-shot method can be used for optimal sensor/actuator placement tasks, or to provide a reference trajectory to be used for online feedback control applications.

Computers & Fluids 2022

Robust preconditioned one-shot methods and direct-adjoint-looping for optimising Reynolds-averaged turbulent flows

S. Nabi[a,*], P. Grover[b], C.P. Caulfield[c]

[a] *Mitsubishi Electric Research Labs, Cambridge, MA, USA*
[b] *Mechanical and Materials Engineering, University of Nebraska-Lincoln, NE, USA*
[c] *BP Institute and Department of Applied Mathematics & Theoretical Physics (DAMTP), University of Cambridge, UK*

Abstract

We compare the performance of direct-adjoint-looping (DAL) and one-shot methods in a design optimization task involving turbulent flow modeled using Reynolds-Averaged-Navier-Stokes equations. Two preconditioned variants of the one-shot algorithm are proposed and tested. The role of an approximate Hessian as a preconditioner for the one-shot method iterations is highlighted. We find that the preconditioned one-shot methods can solve the PDE-constrained optimization problem with the cost of computation comparable (about four-fold) to that of the simulation run alone. This cost is substantially less than that of DAL, which requires $\mathcal{O}(10)$ direct-adjoint loops to converge. The optimization results arising from the one-shot method can be used for optimal sensor/actuator placement tasks, or to provide a reference trajectory to be used for online feedback control applications.

Keywords: one-shot optimization; PDE-constrained optimization; direct-adjoint looping; boundary control

1. Introduction

Model-based optimization and control of fluid dynamical systems is an active area of research [1] in academia and industry. Applications include data assimilation in weather prediction [2], atmospheric boundary layer estimation [3], chemical process control [4], topology optimization in aerodynamics and hydrodynamics [5, 6, 7], and heating, ventilation and air-conditioning (HVAC) [8, 9, 10]. The highly nonlinear multi-scale nature of fluid dynamics necessitates use of sophisticated numerical tools to discover system parameters as well as control inputs that yield optimal performance according to application specific criteria. Such tasks can be formulated as Partial Differential Equation (PDE)-constrained optimization problems [11].
Such optimization problems are invariably solved using some variant of steepest (or gradient) descent algorithm, where computing sensitivities of the cost function with respect to design parameter(s) is a key step. For high-fidelity fluid dynamics models, the *adjoint method* [12] has long been identified as the method of choice for computing these sensitivities [13, 14, 15, 16]. This method involves an auxiliary PDE, called the adjoint PDE, whose solution yields the required sensitivities.
There are several algorithms that employ the adjoint method to obtain optimal solutions to PDE-constrained problems. These algorithms solve the same set of equations, namely direct (forward) and adjoint equations, to determine the sensitivity of a given cost function to the design parameters. The difference is, however, in the manner by which the optimal solutions are obtained by each algorithm. The direct-adjoint-looping (DAL) method [17] involves repeated 'full solves' of the direct PDE and the adjoint equations to convergence, up to a desired numerical residual. The design parameters are updated via gradient descent at the end of each looping step. The equation involving the parameter update is termed as the design or sensitivity

*Corresponding author
Email addresses: nabi@merl.com (S. Nabi), piyush.grover@unl.edu (P. Grover), cpc12@cam.ac.uk (C.P. Caulfield)

equation. The DAL approach is also known as the reduced space approach [18], and the Nested Analysis and Design (NAND) [19] method.
The DAL method does not require factorization of the inverse of the Jacobian during the evaluation of Newton steps for direct or adjoint variables. Naturally, each full solve involves several iterations of the associated linear systems. Overall, the cost of solving the adjoint equations is similar to that of the direct equations.
It has been argued that it is computationally sub-optimal to solve the forward-adjoint system fully in order to extract a good descent direction for the design parameters. This has motivated the development of the one-shot algorithm [20] that solves the coupled system of direct, adjoint and design equations only once. The one-shot method has computational cost comparable to that of solving only the direct equations [20, 21]. One-shot methods are also referred to as full-space approach, all-at-once, simultaneous Analysis and Design (SAND), and piggy-backing methods. The linear systems that arise in most popular one-shot methods are either solved via Jacobi or Seidel iterations. The former involves simultaneous update of all variables during each iteration. On the other hand, each Seidel iteration involves a sequential update of the direct, adjoint and design parameters. Several variants of one-shot method [22, 23] for unsteady PDEs [18], in the presence of additional constraints [24, 25], and chaotic dynamical systems [26] have been studied recently.
The DAL method has been studied extensively for fluid dynamics applications. However, less attention has been given to the one-shot method and its variants. Moreover, the comparison of the DAL method and one-shot methods on turbulent flows governed by turbulent Navier-Stokes equations is lacking. In this paper, we compare the performance of DAL method employed in [9, 8] with two variants of the one-shot method. The two variants are obtained by preconditioning the one-shot system in two different ways. Both these variants involve Siedel iterations. The preconditioners we employ in this paper are closely related to previously reported preconditioners [22, 23]. We perform extensive numerical studies to quantify the effect of the choice of preconditioners and other parameters on the convergence of the optimizer. Finally, we compare the one-shot method to the DAL method in terms of computational cost.
The algorithms are tested on two test problems. The first is an inverse-design problem whose globally optimal solution is known analytically. The second problem is a challenging inverse-design problem involving indoor airflow where a cool zone is being maintained within a larger open space by an appropriate choice of inlet velocities.
The rest of the paper is organized as follows. In Section 2, we describe the class of optimization problems considered in this work, and the direct, adjoint and design equations that make up the system of equations to be solved. In section 3, we describe the DAL method and two algorithms for implementing the one-shot method. We provide a mathematical rationale for choosing the form of preconditioning in the two one-shot algorithms. In Section 4, we provide two case studies and compare the performance of the various algorithms. In Section 5, we provide conclusions and sketch out directions for future research.

2. Problem Formulation

2.1. Abstract Formulation of Optimization Problem

We first introduce the optimization problem using an abstract notation. We consider the optimization problem

$$\begin{aligned} \min_{w,u} \quad & \mathcal{J}(w,u), \\ \text{s.t.} \quad & \mathcal{R}(w,u) = 0, \end{aligned} \tag{1}$$

where $w \in \mathcal{W}$ is the state vector, $u \in \mathcal{U}$ is the design vector and $\mathcal{W}, \mathcal{U}$ are appropriate Hilbert spaces. Here $\mathcal{R} : \mathcal{W} \times \mathcal{U} \to \mathcal{W}$ represents the governing ('forward' or 'direct') equations such as the turbulent Navier-Stokes or Boussinesq equations, and $\mathcal{J} : \mathcal{W} \times \mathcal{U} \to \mathbb{R}$ is the objective function. To avoid notational overload, we use the same notation for infinite dimensional quantities and their finite discretizations.
To solve Eq. 1, we construct a functional $\mathcal{L}(w,u) = \mathcal{J}(w,u) - \lambda^\top \mathcal{R}(w,u)$, where λ is the vector of Lagrange multipliers or the adjoint variables [15, 27, 17]. Hence, the necessary optimality conditions are

$$\mathcal{R}(w,u) = 0 \text{ (direct equations)}, \tag{2}$$
$$\nabla_w \mathcal{L}(w,u,\lambda) = 0 \text{ (adjoint equations)}, \tag{3}$$
$$\nabla_u \mathcal{L}(w,u,\lambda) = 0 \text{ (design equations)}. \tag{4}$$

For a differential operator $\mathcal{A}$, an exact solution $\mathcal{F}$ is such that $\mathcal{A}\mathcal{F} = 0$. An approximate solution obtained from numerical methods is denoted by F. We define the residual as either $\|\mathcal{F} - F\|$ or $\|\mathcal{A}F\|$ where $\|\|$ is an appropriate norm. We denote residuals for direct equations and adjoint equations by Res_w and Res_λ, respectively. The residual for the algebraic design equations is denoted by Res_u. Finally, as $\mathcal{R}$ is zero at feasible points by construction, we may choose λ freely, yielding $\mathcal{L}(w,u) = \mathcal{J}(w,u)$ and, hence, $\nabla\mathcal{L} = \nabla\mathcal{J}$. Throughout the paper, we may choose either notation.

2.2. Governing Equations

We consider flows on a two-dimensional domain $\mathbb{D}$, modeled by the steady Reynolds-averaged Navier-Stokes (RANS) equations,

$$\begin{aligned} &\frac{\partial v_j}{\partial x_j} = 0, \\ &\frac{\partial v_i v_j}{\partial x_j} + \frac{\partial p_i}{\partial x_i} - \frac{\partial}{\partial x_j}(\frac{1}{Re}\frac{\partial v_i}{\partial x_j}) = 0, \end{aligned} \tag{5}$$

where $\mathbf{v}$ and p are non-dimensionalized ensemble-averaged velocity and pressure, respectively. Using the Boussinesq hypothesis for turbulence, the effective viscosity ν_{eff} is the sum of molecular viscosity ν and eddy viscosity ν_t. The latter is computed using a two equation turbulence model, e.g. the standard $k-\epsilon$ closure model [28]. Reynolds number in Eq. 5 is defined as $Re = \frac{V_{ref}L_{ref}}{\nu_{eff}}$,where V_{ref} and L_{ref} are the reference velocity and length of the problem.
We prescribe the following boundary conditions.

$$\begin{aligned} &\text{inlet} : \mathbf{v} = \mathbf{v}_{in}, (n_i\partial/\partial x_i)p = 0, \\ &\text{outlet} : (n_i\partial/\partial x_i)v_i = 0, p = 0, \\ &\text{wall} : \mathbf{v} = 0, (n_i\partial/\partial x_i)p = 0, \end{aligned} \tag{6}$$

where $\mathbf{n}$ is the unit vector normal to the surface, and the subscript *'in'* denotes the values at the inlet.

2.3. Cost Function and Adjoint Equations

We consider the class of optimization problems where the objective is to maintain a desired velocity field $\mathbf{v}_d$ in a region of interest Ω, by choosing optimal inlet velocities $\mathbf{v}_{in}$ as the design parameter. We define the cost function as

$$\mathcal{J} = \int_\Omega (\mathbf{v} - \mathbf{v}_d)^2 dV, \tag{7}$$

The adjoint equations Eq. 3 are explicitly given by

$$\begin{aligned} &\frac{\partial v_{a,j}}{\partial x_j} = 0, \\ &v_{a,j}\frac{\partial v_j}{\partial x_i} - v_j\frac{\partial v_{a,i}}{\partial x_j} - \frac{\partial}{\partial x_j}(\frac{1}{Re}\frac{\partial v_{a,i}}{\partial x_j}) + \frac{\partial p_a}{\partial x_i} + \beta(v_i - v_{d,i}) = 0, \end{aligned} \tag{8}$$

The corresponding adjoint boundary conditions are

$$\begin{aligned}
&\text{inlet}: \mathbf{v}_a = 0, (n_i\partial/\partial x_i)p_a = 0,\\
&\text{outlet}: v^n v_a^t + \frac{1}{Re}(n_i\partial/\partial x_i)v_a^t = 0,\\
&p_a = v^n v_a^n + \frac{1}{Re}(n_i\partial/\partial x_i)v_a^n,\\
&\text{wall}: \mathbf{v}_a = 0, (n_i\partial/\partial x_i)p_a = 0,
\end{aligned} \tag{9}$$

where superscripts n and t denote the normal and tangential components, respectively.
We use the 'frozen turbulence' hypothesis [29] in deriving Eq. 8. As a result of this assumption, the effective viscosity used in Eq. 8 is obtained by solving the $k-\epsilon$ equations along with the direct system of equations Eq. 5. An assessment of the validity of this assumption for problems similar to those studied in the current work has been carried out in [9].
The design equation Eqs. 4 are explicitly given by

$$\nabla_u \mathcal{L} = p_a \mathbf{n} - \frac{1}{Re}\mathbf{n}.\nabla\mathbf{v}, \tag{10}$$

where all values are evaluated at the inlet. For the specific RANS-constrained optimization problem, $w = (\mathbf{v}, p)$ is the vector of state variables, and u is the vector of design variables i.e. $u = (\mathbf{v}_{in})$. The vector of adjoint variables is $\lambda = (\mathbf{v}_a, p_a)$. In that context, Eqs. 5, 6 correspond to Eq. 2, Eqs. 8, 9 correspond to Eq. 3, and Eq. 10 corresponds to Eq. 4, respectively. The Eqs. 8-10 are derived in [8].
The gradient descent based update of the design parameters, given by Eq. 10, requires selection of a step size. Properly chosen step sizes using line-search [30], or adaptation [31], may increase the rate of convergence. Other strategies, e.g. relaxation approach or conjugate gradient, can also be adopted; however, it is unclear which, if any, is superior [27]. In this paper, we do not seek optimal step sizes (although this is an important direction for future studies) and instead our focus is on i) the role of preconditioner on the convergence and robustness, and ii) comparison of the DAL method with variants of the one-shot method. For each algorithm, we picked the step size based on numerical experiments.

2.4. Details of Numerical Solver

We use the finite-volume [32] solver OpenFOAM [33]. This solver uses a collocated grid arrangement and offers object-oriented implementations that suit the employed continuous adjoint formulation (the solver is based on icoFoam). Pressure and velocity are decoupled using the SIMPLE algorithm [34] technique in the direct/adjoint equations. For the convection terms, second order Gaussian integration is used with the Sweby limiter [35] for numerical stability. For diffusion, Gaussian integration with central-differencing-interpolation is used. The discretized algebraic equations are solved using the Preconditioned BiConjugate Gradient (PBiCG) method [36]. The adjoint equations are also solved using the numerical method described for solving the forward or direct equations. We found that using an upwind and first order method for solving the adjoint equations Eqs. 8 resulted in inaccurate gradients, and hence those methods were not adopted.

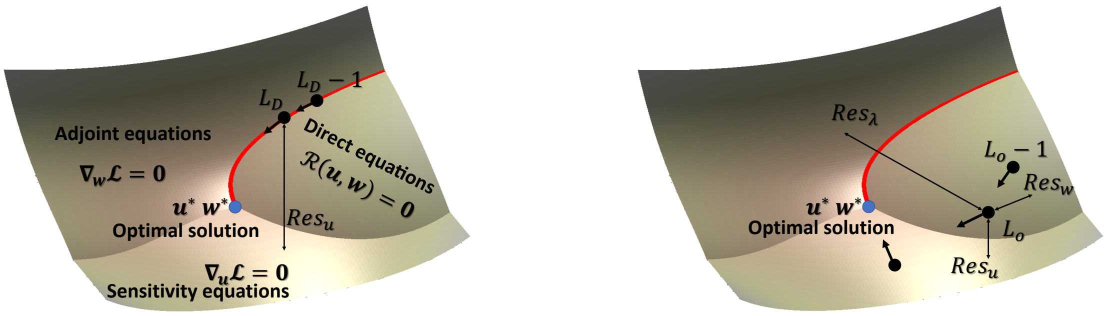
3. Algorithms for solving optimality equations

3.1. Direct-Adjoint-Looping Method

Each 'loop' of the DAL method 1, indexed by L_D, involves a 'full solve' (up to a chosen residual) of the direct Eq. 2, followed by the solution of the adjoint Eq. 3. In general, each such 'full solve' requires multiple individual iterations of the linear systems corresponding to direct and adjoint equations, which are indexed by k_D and k_a, respectively. This is followed by a gradient descent step of the algebraic design Eqs. 4. The trajectory $(w^{L_D}, u^{L_D}, \lambda^{L_D})$ approaches the optimal solution (w^*, u^*, λ^*) along the intersection of the solution hypersurfaces of the direct and adjoint equations, as shown in Fig. 1(a). Since our aim is to compare the DAL method with preconditioned one-shot methods, we also use a BFGS type preconditioner (B^D) in our DAL method implementation. This preconditioner is similar to the one discussed in Section 3.2.1.

Algorithm 1 DAL Algorithm (D)

```
1. Put $L_D = 0$. Obtain an admissible guess $u^{L_D}$ for the vector of design parameters.
2. Solve the direct Eqs. $\mathcal{R}(w, u^{L_D}) = 0$, up to a residual $Res_w < \epsilon_w$, to obtain $w^{L_D}$.
3. Solve the adjoint Eqs. $\nabla_w \mathcal{L}(w^{L_D}, u^{L_D}, \lambda) = 0$, up to a residual $Res_\lambda < \epsilon_\lambda$, to obtain $\lambda^{L_D}$.
4. IF $Res_u > \epsilon_u$
     update $u^{L_D+1} = u^{L_D} - \alpha(B^D)^{-1}\nabla_u \mathcal{L}(w^{L_D}, u^{L_D}, \lambda^{L_D})$,
     $L_D \to L_D + 1$,
     Go to Step 2.
   ELSE
     Terminate.
```



(a) DAL method (b) One-shot method

Figure 1: Hypersurfaces of direct (or forward), adjoint and sensitivity equations. The residuals for the design variable, Re_u, direct variables, Res_w, and adjoint variables Res_λ, defined in the text, are shown via a thin double-headed arrow. The red curve is the intersection of forward and adjoint hypersurfaces. Panel (a) is a schematic of the DAL method and panel (b) is a schematic of the one-shot method. For the DAL method, the residuals for direct and adjoint equations are assumed to be negligible. The black circles represent iterations as defined in the text, the black arrows indicate a hypothetical direction of the (scaled) gradient. L_D and L_o are the number of iteration loops for the DAL method and one-shot methods, respectively. The optimal solution is denoted by the blue circle at the intersection of the three hypersurfaces and corresponds to u^* and w^*.

3.2. One-shot Methods

The one-shot method involves solving the combined optimality system of Eqs. (2,3,4) only once. It was first proposed by Hazra and Schulz [37], inspired by an earlier research by Ta'saan [38]. It has been observed numerically that one-shot methods can outperform DAL methods by about an order of magnitude in terms of iteration counts and runtime [19].

Several variants were proposed by Griewank et al. [21] and others (e.g., [39, 40]). As mentioned in the introduction, one classification of these variants can be done according to the type of linear systems iterations involved in the process [22]. Jacobi iteration has been employed by [21, 39, 40, 41, 42], while [20, 22, 23, 37] have employed Siedel iteration variants. It has been reported that the computational cost of the Siedel method could be lower than that of the Jacobi variant [22, 43].

For each one-shot variant, several choices for the preconditioner B have been proposed to ensure convergence of the coupled iteration. For the Jacobi variants, a modified exact projected Hessian was introduced in [21] to ensure that real eigenvalues of the coupled iteration are within the unit circle. However, as shown in [44], complex eigenvalues with modulus greater than 1 may still occur in that case. Hence, for the Jacobi variant, most efforts are aimed at constructing the preconditioner using an approximation of the partial Hessian of the augmented Lagrangian [41], i.e.

$$\mathcal{L}^a(w, u) = N(w, u, \lambda) - \lambda^\top w + \frac{\alpha'}{2}\|G(w, u) - w\| + \frac{\beta'}{2}\|N_w(w, \lambda, u) - \lambda\|.$$

Here, G is a contractive operator that arises in fixed point iteration of the forward equations, i.e., Eq. 2 is replaced by $w = G(w, u)$. Finally, $N(w, \lambda, u) \coloneqq \mathcal{J} + G(w, u)\lambda$ is the shifted Lagrangian. Convergence can

be enforced by suitable choice of α' and β', and an approximate $B^{L_o} \approx \nabla^2\mathcal{L}^a$ can be efficiently computed by low-rank secant updating, e.g. BFGS type updating. However, theoretical bounds may result in conservative design of the parameters and, thereby, lead to slow convergence. In practice moderate estimates of α' and β' may lead to good convergence behavior [23].
On the other hand, preconditioners for the Siedel one-shot variant with continuous adjoint formulation have been less explored. In [23], the authors proposed to control the complex eigenvalues without making design updates excessively conservative and they demonstrated local convergence of the coupled iteration under moderate assumptions in the neighborhood of the fixed point of the KKT system. They used a BFGS update on the Hessian of Lagrangian, i.e., $\nabla^2\mathcal{L}$ (instead of augmented Lagrangian $\nabla^2\mathcal{L}^a$). The test cases were restricted to systems with linear Laplace-like operators. Subsequently, [22] performed a comparison of Jacobi and Siedel variants for a laminar incompressible Boussinesq problem with boundary control, and used AD to calculate the adjoint variables.
The one-shot methods we study in this paper are modifications of the Siedel type variants discussed in [20, 37]. First, we show that under some assumptions, these variants result in the same linear systems as the Jacobi variants introduced in [21, 41]. As we shall see, the difference between these methods boils down to the choice of the preconditioner.
The necessary conditions for optimality Eqs. (2,3,4) can be written in a single system as

$$F(w,\lambda,u) = -\begin{pmatrix} \nabla_w \mathcal{L}(w,\lambda,u) \\ \nabla_u \mathcal{L}(w,\lambda,u) \\ \mathcal{R}(w,\lambda,u) \end{pmatrix} = 0. \tag{11}$$

Consider a linearization of Eq. 1 by a Taylor expansion to construct a Reduced Sequential Quadratic Programming (RSQP) formulation of the problem:

$$\begin{aligned} \min_{w,u} \quad & \mathcal{J}(w,u) \\ \text{s.t.} \quad & \mathcal{R}(w,u) + J(w,u)\Delta w + \mathcal{R}_u(w,u)\Delta u = 0, \end{aligned} \tag{12}$$

with the exact Jacobian $J = \frac{\partial \mathcal{R}}{\partial w}$ assumed to be invertible in this study.
It can be shown that the solution of Eq. 12 can be written as a coupled linear system of the form

$$\begin{pmatrix} J^\top & 0 & 0 \\ \frac{\partial \mathcal{R}}{\partial u}^\top & H & 0 \\ 0 & \frac{\partial \mathcal{R}}{\partial \mathcal{U}} & J \end{pmatrix} \begin{pmatrix} \Delta\lambda \\ \Delta u \\ \Delta w \end{pmatrix} = -\begin{pmatrix} \nabla_w \mathcal{L} \\ \nabla_u \mathcal{L} \\ \mathcal{R} \end{pmatrix}, \tag{13}$$

where H is the exact Hessian.
Eq. 13 can be considered as an approximate Newton step for the optimality system of Eq. 11. Inspired by the research presented in [45, 46], a preconditioner K with block structure similar to Eq. 13 was proposed in [37]. Let A, B denote the approximate Jacobian and Hessian, respectively. We split the state and design variable vectors into two parts $w = (w_\mathcal{I}, w_b)$ and $u = (u_\mathcal{I}, u_b)$, where the subscript b refers to the part of domain where the design vector explicitly appears in Eq. 11. The subscript $\mathcal{I}$ corresponds to the rest of the domain. For the class of problems we consider in this work, design parameters only appear in the inlet boundary conditions. Hence, $w_\mathcal{I} = (\mathbf{v}, p)$, $w_b = (\mathbf{v}_{in})$, and $u_\mathcal{I} = 0$. Similarly, $R_b = w_b - u_b = 0$ is the inlet boundary condition, and $R_\mathcal{I} = 0$ denotes the direct equation along with its remaining boundary conditions. This yields

$$\frac{\partial \mathcal{R}_b}{\partial u_b} = -I,\ \frac{\partial \mathcal{R}_\mathcal{I}}{\partial u_\mathcal{I}} = 0.$$

Furthermore, we can write the approximate Jacobian as

$$A = \begin{pmatrix} A_{\mathcal{I}\mathcal{I}} & A_{\mathcal{I}b} \\ A_{b\mathcal{I}} & A_{bb} \end{pmatrix},$$

where

$$A_{\mathcal{I}b} = \frac{\partial \mathcal{R}_{\mathcal{I}}}{\partial w_b} = 0, \; A_{b\mathcal{I}} = \frac{\partial \mathcal{R}_b}{\partial w_{\mathcal{I}}} = 0.$$

Then Eq. 11 with a preconditioner K is

$$-K \begin{pmatrix} \nabla_{w_{\mathcal{I}}} \mathcal{L}(w, \lambda, u) \\ \nabla_{w_b} \mathcal{L}(w, \lambda, u) \\ \nabla_u \mathcal{L}(w, \lambda, u) \\ \mathcal{R}_b(w, \lambda, u) \\ \mathcal{R}_{\mathcal{I}}(w, \lambda, u) \end{pmatrix} = 0, \tag{14}$$

where

$$K = \begin{pmatrix} A_{\mathcal{I}\mathcal{I}}^{\top} & 0 & 0 & 0 & 0 \\ 0 & A_{bb}^{\top} & 0 & 0 & 0 \\ 0 & \frac{\partial \mathcal{R}_b}{\partial u_b}^{\top} & B & 0 & 0 \\ 0 & 0 & \frac{\partial \mathcal{R}_b}{\partial u_b} & A_{bb} & 0 \\ 0 & 0 & 0 & 0 & A_{\mathcal{I}\mathcal{I}} \end{pmatrix}^{-1}. \tag{15}$$

$$= \begin{pmatrix} A_{\mathcal{I}\mathcal{I}}^{-\top} & 0 & 0 & 0 & 0 \\ 0 & A_{bb}^{-\top} & 0 & 0 & 0 \\ 0 & -B^{-1}A_{bb}^{-\top} & B^{-1} & 0 & 0 \\ 0 & A_{bb}^{-1}B^{-1}A_{bb}^{-\top} & -A_{bb}^{-1}B^{-1} & A_{bb}^{-1} & 0 \\ 0 & 0 & 0 & 0 & A_{\mathcal{I}\mathcal{I}}^{-\top} \end{pmatrix}. \tag{16}$$

Hence, Eq. 14 reduces to

$$\begin{aligned} A_{\mathcal{I}\mathcal{I}}^{-\top} \nabla_{w_{\mathcal{I}}} \mathcal{L} &= 0, \\ A_{bb}^{-\top} \nabla_{w_b} \mathcal{L} &= 0, \\ -B^{-1} A_{bb}^{-T} \nabla_{w_b} \mathcal{L} + B^{-1} \nabla_u \mathcal{L} &= 0, \\ A_{bb}^{-1} B^{-1} A_{bb}^{-\top} \nabla_{w_b} \mathcal{L} + A_{bb}^{-1} B^{-1} \nabla_u \mathcal{L} + A_{bb}^{-1} \mathcal{R}_b &= 0, \\ A_{\mathcal{I}\mathcal{I}}^{-\top} \mathcal{R}_{\mathcal{I}} &= 0, \end{aligned} \tag{17}$$

By definition again, $A_{bb} = \frac{\partial \mathcal{R}_b}{\partial w_b} = I$. The term $A_{\mathcal{I}\mathcal{I}} = \frac{\partial \mathcal{R}_{\mathcal{I}}}{\partial w_{\mathcal{I}}}$ is the (approximate) Jacobian, and we set $A_{\mathcal{I}\mathcal{I}} = I$ following [47]. As indicated in Eq. 9, Dirichlet boundary conditions for the direct equations result in homogeneous (zero) boundary conditions for the adjoint equations, i.e., since $\delta w_b = 0$, the optimality conditions are satisfied for any value of $\nabla_{w_b} \mathcal{L}$. Without loss of generality for inlet boundaries, we set $\nabla_{w_b} \mathcal{L} = 0$. As a result, and by uniting b and $\mathcal{I}$ domains, the preconditioned Eqs. 14 are reduced to

$$\begin{aligned} \mathcal{R} &= 0, \\ \nabla_w \mathcal{L} &= 0, \\ B^{-1} \nabla_u \mathcal{L} &= 0. \end{aligned} \tag{18}$$

Hence, we have shown the equivalence of Eq. 14 and 18 under assumptions for the choice of quantities in Eq. 15.
Next, we discuss different preconditioning choices for Siedel type one shot methods.

3.2.1. Preconditioned One-Shot methods with Siedel Iterations

In [47], the authors introduced pseudo-time stepping as a means of under-relaxing Eq. 18 as they iterate towards the steady-state feasibility and optimality. In this work, instead of adopting the notion of pseudo-time stepping, we use iterations with explicit under-relaxation. In the following, L_o is the iteration index for the one-shot method, and B^{L_o} is the approximate Hessian of the Lagrangian. Each iteration of the one-shot method for Eq. 18 involves one iteration of the linear systems of the direct and adjoint fixed point equations, followed by a gradient descent step of the preconditioned design equation. Therefore, a typical trajectory $(w^{L_o}, u^{L_o}, \lambda^{L_o})$ approaches the optimal solution (w^*, u^*, λ^*) along a path that need not lie on any solution hypersurface, as shown schematically in Fig. 1(b).

Hazra and Schulz [37, 20, 47] proposed the one-shot method 2, with the following approximation for the Hessian

$$B^{L_o} = \begin{cases} \bar{\beta}\frac{(z^{L_o})^\top s^{L_o}}{(z^{L_o})^\top z^{L_o}} I & (z^{L_o})^\top s^{L_o} > 0, \\ \hat{\beta} I & \text{otherwise}, \end{cases} \tag{19}$$

where $z^{L_o} = \nabla_u \mathcal{L}^{L_o+1} - \nabla_u \mathcal{L}^{L_o}$ and $s^{L_o} = u^{L_o+1} - u^{L_o}$. The following upper and lower bounds on $\bar{\beta}$ are imposed:

$$\beta_{min} \leq \bar{\beta}\frac{(z^{L_o})^\top s_{L_o}}{z_{L_o}^\top z_{L_o}} \leq \beta_{max},$$

where $\bar{\beta}, \hat{\beta}, \beta_{min}$ and β_{max} are constants to be chosen based on the problem. There is no systematic way to pick these constants. Furthermore, the approximate Hessian is re-evaluated at each iteration with no explicit memory of the previous iterations. To remedy this, we modify the approximate Hessian as follows:

$$B^{L_o} = \begin{cases} \bar{\beta}\frac{(z^{L_o})^\top s^{L_o}}{(z^{L_o})^\top z^{L_o}} I & (z^{L_o})^\top s^{L_o} > 0, \\ B^{L_o-1} & \text{otherwise}. \end{cases} \tag{20}$$

This update eliminates one of the constants $\hat{\beta}$, and uses the previous iteration for the approximate Hessian if positive definiteness of B^{L_o} is not achieved. As we will see in the next section, this modification leads to better convergence in practice. Moreover, in practice we set $\alpha = 1$ for this algorithm so that we only optimize $\bar{\beta}$. The one-shot method with the above preconditioner is referred to as the 'curvature-based one-shot algorithm' (OC).

Algorithm 2 Curvature-based one-shot algorithm (OC)

```
1. Set $L_o = 0$, $\lambda^{L_o} = 0$ and and obtain an admissible initial guess for design parameters $u$. Also need an initial
   guess for $w_0$ and approximate Hessian $B^{L_o}$.
2. Compute $\lambda^{L_o+1} = N(w^{L_o}, \lambda^{L_o}, u^{L_o})$, i.e., obtain the next iterate of the adjoint equation.
3. Compute $B^{L_o}$ from either Eq. 19 or 20.
4. Update the design parameter $u^{L_o+1} = u^{L_o} - \alpha(B^{L_o})^{-1}\nabla_u \mathcal{L}(w^{L_o}, \lambda^{L_o+1}, u^{L_o})$.
5. Compute $w^{L_o+1} = G(w^{L_o}, u^{L_o+1})$, i.e., obtain the next iterate of the direct equation.
6. IF $Res_w > \epsilon_w$ OR $Res_\lambda > \epsilon_\lambda$ OR $Res_u > \epsilon_u$
     $L_o \to L_o + 1$.
     Go to step 2.
   ELSE
     Terminate.
```

Algorithm 2 still requires judicious selection of several constants. The diagonal structure of the approximate Hessian is also excessively restrictive. To remedy these shortcomings, we use the popular BFGS [30] method for the update, which has also been used by other researchers [22, 39]. This preconditioner is applied to $\nabla^2 \mathcal{L}$ instead of $\nabla^2 \mathcal{L}^a$, as is common for Siedel iterations. This method proposes the following choice for the inverse of the approximate Hessian, $\hat{H} = B^{-1}$.

$$\hat{H}^{L_o+1} = \begin{cases} \left(I - \rho^{L_o} s^{L_o}(z^{L_o})^\top\right)\hat{H}^{L_o}\left(I - \rho^{L_o} z^{L_o}(s^{L_o})^\top\right) + \rho^{L_o} s^{L_o}(s^{L_o})^\top & \rho^{L_o} > 0, \\ \hat{H}^{L_o} & \text{otherwise}, \end{cases} \tag{21}$$

with

$$\rho^{L_o} = \frac{1}{(z^{L_o})^\top s^{L_o}}.$$

In addition to superior convergence properties of the BFGS update, the use of the inverse of the approximate Hessian also allows the search direction to be calculated by means of a simple matrix-vector multiplication. Also, following some previous results [48], and for the reasons to be clarified in Section 4, we normalize the gradient in the search direction.
Finally, in algorithm 3, we use the following update for design variables:

$$u^{L_o+1} = u^{L_o} - \alpha \hat{H}^{L_o} \frac{\nabla_u \mathcal{L}^{L_o}}{\|\nabla_u \mathcal{L}^{L_o}\|}. \tag{22}$$

Algorithm 3 BFGS-based one-shot algorithm with normalized gradient (OB)

```
1. Set $L_o = 0$, $\lambda^{L_o} = 0$ and and obtain an admissible initial guess for design parameter $u$. Also need an initial guess for $w_0$ and approximate Hessian $B^{L_o}$.
2. Compute $\lambda^{L_o+1} = N(w^{L_o}, \lambda^{L_o}, u^{L_o})$, i.e., obtain the next iterate of the adjoint equation.
3. Compute $\hat{H}^{L_o}$ from Eq. 21.
4. Compute the updated design parameter, $u^{L_o+1}$, via Eq. 22.
5. Compute $w^{L_o+1} = G(w^{L_o}, u^{L_o+1})$, i.e., obtain the next iterate of the direct equation.
6. IF $Res_w > \epsilon_w$ OR $Res_\lambda > \epsilon_\lambda$ OR $Res_u > \epsilon_u$
     $L_o \to L_o + 1$.
     Go to step 2.
   ELSE
     Terminate.
```

In the next section, we compare the results of Algorithms 1 (D), 2 (OC), and 3 (OB). We assess the accuracy and computational cost of each algorithm. We pick the step size α to be a reasonable value across all methods. Ideally an optimal step can be found using line search methods and strong Wolfe conditions [30]. However, for CFD applications, such a method is not numerically tractable, as it requires expensive function evaluations. We quantify the computational cost by the operation counts n_D and n_o for the DAL method and the particular one-shot method, respectively. Using the notation of Fig. 1, we have $n_D = (k_D + k_a)L_D$ and $n_o = 2L_o$.

4. Results and Discussion

To validate our numerical framework, we first consider the analytically solvable case of a laminar parallel flow between two plates. Next, we consider the case of a turbulent mixing flow in an enclosure with 8 design variables. This problem is solved using the two variants of one-shot method introduced in the previous sections, as well as the preconditioned DAL method.

4.1. Analytical case study: Plane Poiseuille Flow

In this case study, we consider the two-dimensional, laminar, incompressible Poiseuille flow between two horizontal parallel plates of length L_{ref}, separated by a vertical distance h. We denote the inlet velocity of this pressure driven flow by V_{in}. The flow variables (velocity and pressure) are governed by the two-dimensional Navier–Stokes equations 5. We choose a uniform (in the wall-normal direction) inlet condition of $V_{in}^o(y) = 1m/s$, and denote the corresponding steady state by $\mathbf{v}_d$, as the desired velocity in the region of interest.
The aim of the inverse problem is to recover the correct inlet speed, assuming that the inlet flow is uniform. We employ both the DAL method and one-shot methods to optimize the cost function given by Eq. 7.

Since the 'optimal' solution is known (V_{in}^o), this procedure serves as a method to validate the numerical optimization framework. For this test case we have:

$$\begin{aligned}
\mathbf{v}_d &= \frac{6}{h^2} V_{in}(yh - y^2)\hat{\imath}, \\
\mathcal{J} &= \big(\frac{6}{h^2}\big)^2 (V_{in} - V_{in}^o)^2 \frac{\ell h^5}{30}, \\
\nabla_{V_{in}} \mathcal{J} &= 2\big(\frac{6}{h^2}\big)^2 (V_{in} - V_{in}^o) \frac{\ell h^5}{30},
\end{aligned} \tag{23}$$

where ℓ is the length of the rectangular region of interest. This region is taken to be in the fully developed region for which $\dfrac{\partial \mathbf{v}}{\partial x} = 0$. Here $\hat{\imath}$ is the unit vector in the horizontal direction. x and y are horizontal and wall-normal (vertical) coordinates, respectively. We non-dimensionalize using $L_{ref} = 2m$ and $V_{ref} = 1m/s$ as reference scales. The region of interest is taken to be $[0.5, 0.75] \times [0, h]$. We start the optimization procedure for all methods from a guess of $V_{in} = 0.5$.

To verify that one-shot Algs. 2 and 3 are computing the correct sensitivities, we first use $\alpha = 0$ to obtain the gradient of the cost function at the initial guess. We set the residuals as $\epsilon_w = \epsilon_\lambda = 10^{-6}$ in this case. Fig. 2 shows the evolution of $\nabla_{V_{in}}\mathcal{J}$ along with Res_w and Res_λ for $V_{in} = 0.5$. After initial oscillations, the gradient converges to $\nabla_{V_{in}}\mathcal{J} = -5.797 \times 10^{-4}$. The analytical value from Eq. 23 is -6×10^{-4}. Hence, the relative error compared to the analytical value is less than 0.3 percent for a mesh of the size 400 elements (20×20). An alternative calculation using the finite difference:

$$\nabla_{V_{in}}\mathcal{J} = \frac{\mathcal{J}(V_{in} + \delta V_{in}) - \mathcal{J}(V_{in} - \delta V_{in})}{2\delta V_{in}}, \tag{24}$$

yields $\nabla_{V_{in}}\mathcal{J} = -5.98 \times 10{-}4$ with δV_{in} as the perturbation of the inlet velocity. Thus, the relative error between the gradient calculated by the adjoint method and that of the finite difference is only 0.05 percent. By refining the mesh adjacent to the inlet, one may obtain a more accurate estimate of $\nabla_{V_{in}}\mathcal{J}$ [9].

After evaluating the accuracy of gradients, we now consider the inverse problem using both the one-shot (OC variant) method and the DAL method. As shown in Fig. 3, both one-shot and DAL converge to the optimal solution, i.e. $V_{in}^o = 1$, with the desired accuracy for $\nabla_{V_{in}}\mathcal{J}$, which we set to be 10^{-6}. In all figures the norm of the gradients are plotted on the log-scale. The one-shot method reaches the residual threshold of 1e-6 in $L_o = 628$ iterations, implying $n_o = 1256$ operation counts for the direct and adjoint equations. Each DAL method 'loop' comprises a direct (or forward) solution and an adjoint solution, each of which require approximately $k_D = 200$ and $k_a = 150$ iterations respectively on average to satisfy the same convergence criteria. Overall, $L_D = 53$ loop iterations are required for the gradient to reach $Res_u = 10^{-6}$. Hence the total operation count is $n_D = 18550$, which is ≈ 15 times that of the one-shot method. We note that for the current case study, we are using a fixed step with $B_k = I$.

4.2. Numerical case study: Zone cooling

This case study involves buoyancy-driven airflow in a 2D room. A schematic for the problem domain $\mathbb{D}$ is shown in Fig. 4. The height and the length of the room are denoted by H and L, respectively. We divide the room into two hypothetical regions: an 'occupied zone' on the left with length ℓ_1; and an 'unoccupied zone' on the right of the room. Each zone has two inlets and one outlet, which are indexed as shown in Fig. 4. A prescribed inflow is provided, for example, by an air conditioning unit or a mechanical fan. The inlet velocity is $\mathbf{V}_{in,i}$ with $i \in 1, 2, 3, 4$ indicating the index of the inlet. For instance, $\mathbf{V}_{in,1}$ is the velocity boundary condition of inlet 1 with horizontal and vertical components of $V_{in,1}^x$ and $V_{in,1}^y$. The design variables are taken to be the magnitude and direction of the inlet velocity.

For mesh sensitivity analysis, we carried out two separate tests for the direct and adjoint equations. For the solutions of the direct equations, we used the cost function as the monitoring parameter to check the mesh sensitivity between three mesh sizes of 28,400, 133,920 and 525,120 elements. For the steady state solutions,

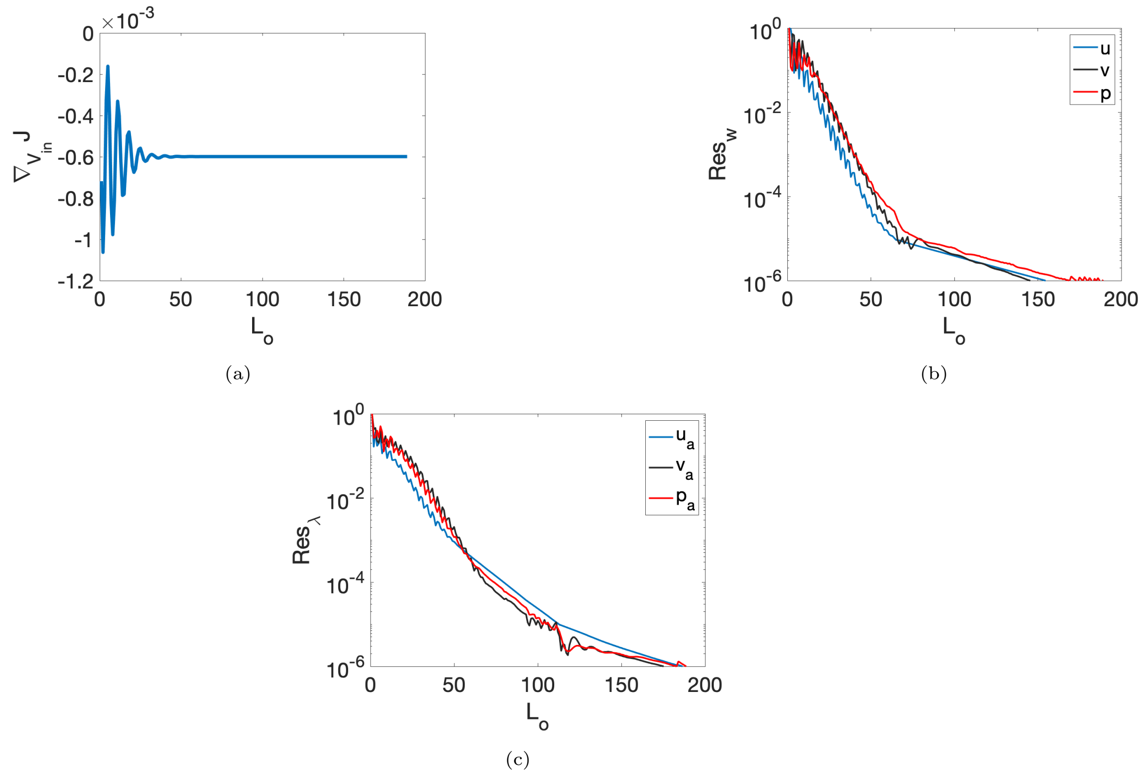


Figure 2: Evolution of a) the gradient of the cost function $\nabla_{V_{in}}\mathcal{J}$, b) residuals for the direct equations Res_w, i.e. two components of velocity and pressure, and c) residuals for adjoint equations Res_λ, i.e. two components of adjoint velocity and pressure, for $V_{in} = 0.5$.

we noticed that the cost function differs by less than 8% relative error between the solutions calculated using the 28,400 elements and 525,120 elements meshes. For the adjoint equations, on the other hand, sensitivity to the mesh size is much more pronounced, particularly near the inlet. This can be seen from Eq. 10, where the gradient of adjoint velocity at the inlet boundary is employed to calculate the gradient of the cost function. Hence, we refined the mesh in particular around the inlet. We found significant discrepancy between the gradient value obtained using Eq. 10 for a 28,400 elements mesh, and the gradient computed using a finite-difference calculation. Therefore, a refined mesh with 133,920 elements is used to solve the adjoint equation, even though significantly fewer mesh elements are required if only direct simulations were to be considered. Such computationally demanding behavior of the continuous adjoint method is also observed in other studies (e.g. see [49]).

The test case is motivated by application to 'targeted' cooling of the occupied zones in large open areas within a built environment. Such a zone cooling approach potentially leads to a decrease in the energy consumption while maintaining thermal comfort for the occupants. The region of interest, denoted by Ω, is a rectangular region that extends from $x = 3m$ to $x = 5m$ in the horizontal direction and from the floor to a height of $y = 1.5m$, to mimic a typical occupancy height in the occupied zone. For such architectural fluid mechanics problems, the airflow patterns depend on the relative location of the heat source and inlets [50, 51]. Hence, we consider two regions of interest, Ω_I and Ω_{II}.

We pick the reference inlet velocities at all inlets to be pointing downward. The velocity magnitude is chosen such that the Reynolds number is $Re = 4000$, which is typical for this class of problems within 'architectural fluid mechanics' [50]. The steady state $\mathbf{v}_d$ obtained using these input velocities is shown in Fig. 5a. As shown, by carefully designing the velocity at inlets, two large source-sink pairs from each inlet to the nearby outlet are formed in each zone, separated by two horizontally-narrow vortices which extend vertically over the entire cavity. These two narrow vortices minimize the exchange flow between the two zones and form a

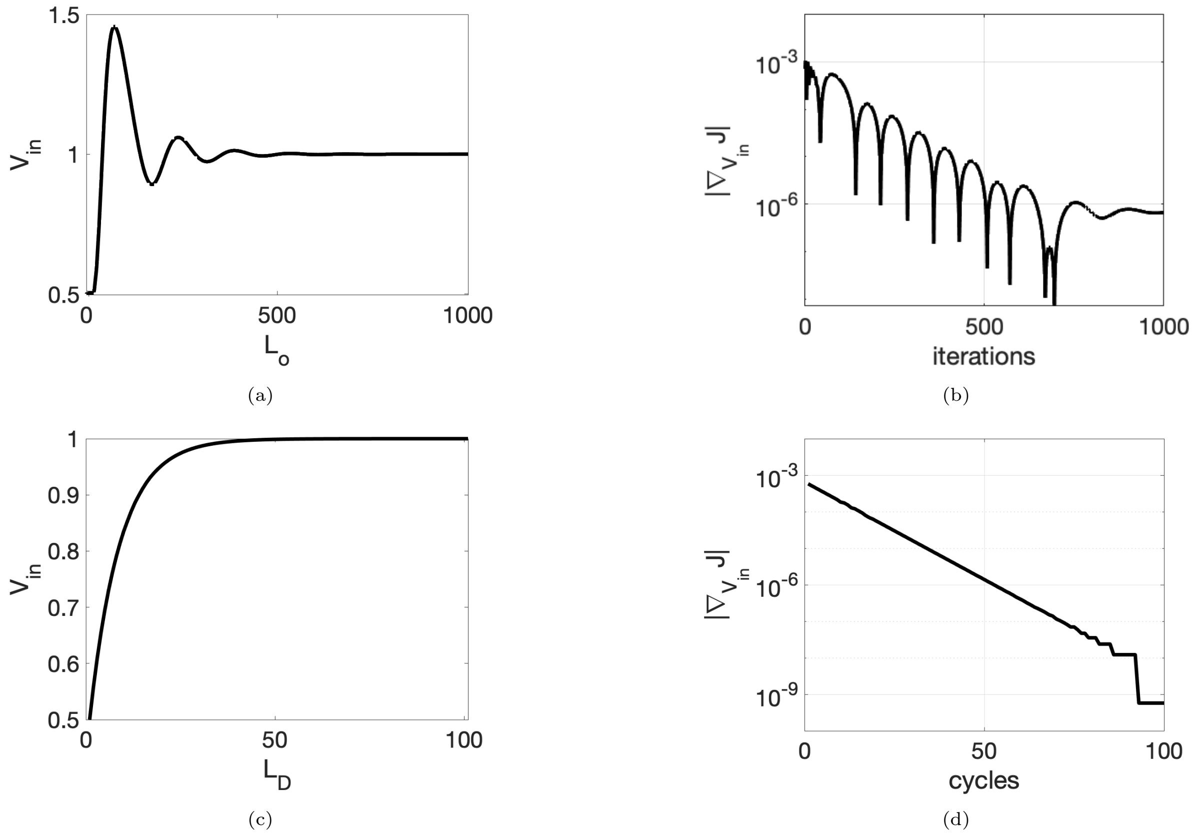


Figure 3: Optimization results for the case study involving laminar plane Poiseuille flow. (a,b) Design variable, and $\|\nabla_{V_{in}}\mathcal{J}\|$ based on the (OC) one-shot method as a function of L_o. (c,d) Design variable, and $\|\nabla_{V_{in}}\mathcal{J}\|$ based on the DAL method as a function of L_D.

virtual air curtain.
The aim of the inverse problem is to identify the correct inlet velocities, and the cost function is defined in Eq. 7. We consider two regions of interest, denoted by Ω_I and Ω_{II} (as shown in Fig. 4). Region of interest I focuses on local mixing in the occupied zone, while region of interest II extends across both the occupied and unoccupied zones. To examine the robustness of the methods, we also consider two different initial guesses. The initial guesses are listed in Table 1.
Next, we discuss the results of the test problem 1, i.e. the case with region of interest I (see Fig. 4) and initial guess I. When the direct (forward) problem is solved with the provided initial guess, we obtain $\mathcal{J}^0 = 8.49 \times 10^{-5}$, the maximum (among all inlets i) of $\nabla_{V^y_{in,i}}\mathcal{J}^0 = -6.00 \times 10^{-5}$, and the maximum (among all inlets i) of $\nabla_{V^x_{in,i}}\mathcal{J}^0 = 4.36 \times 10^{-6}$. This steady-state flow, corresponding to the initial guess I in Ω_I, is shown in Fig. 5-b. We define three quantities R_J, R_x and R_y to quantify the performance of each

Table 1: Various tests for zone cooling case study. Region of interests I and II are denoted by Ω_I and Ω_{II}, respectively.

Test 1	Initial guess I	$V^y_{in,i}/V^x_{in,i} = -0.7914, 0.7914, -0.7914, 0.7914$	Ω_I
Test 2	Initial guess II	$V^y_{in,i}/V^x_{in,i} = 1.8610, 0.8992, 0.2567, -1.7011$	Ω_I
Test 3	Initial guess I	$V^y_{in,i}/V^x_{in,i} = -0.7914, 0.7914, -0.7914, 0.7914$	Ω_{II}
Test 4	Initial guess II	$V^y_{in,i}/V^x_{in,i} = 1.8610, 0.8992, 0.2567, -1.7011$	Ω_{II}

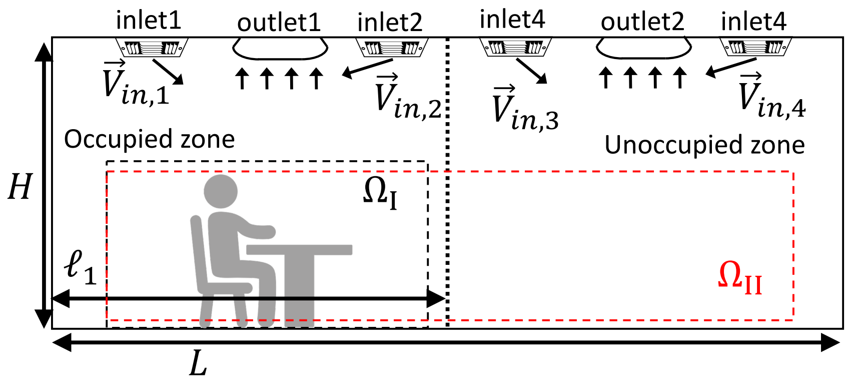


Figure 4: Problem domain $\mathbb{D}$ for buoyancy-driven flow with occupied and unoccupied regions marked. The design variables are the inlet velocities $\mathbf{V}_{in,1}\dots\mathbf{V}_{in,4}$ with subscripts $1\dots4$ indicating the inlet index. Each region has one outlet at the top. The regions of interest Ω_I and Ω_{II} are highlighted with black and red dashed boxes respectively within $\mathbb{D}$.

optimization algorithm:

$$R_J = \frac{J^*}{J^0},\ R_x = \max_i \frac{\nabla_{V^x_{in,i}}\mathcal{J}^*}{\nabla_{V^x_{in,i}}\mathcal{J}^0},\ R_y = \max_i \frac{\nabla_{V^y_{in,i}}\mathcal{J}^*}{\nabla_{V^y_{in,i}}\mathcal{J}^0},$$

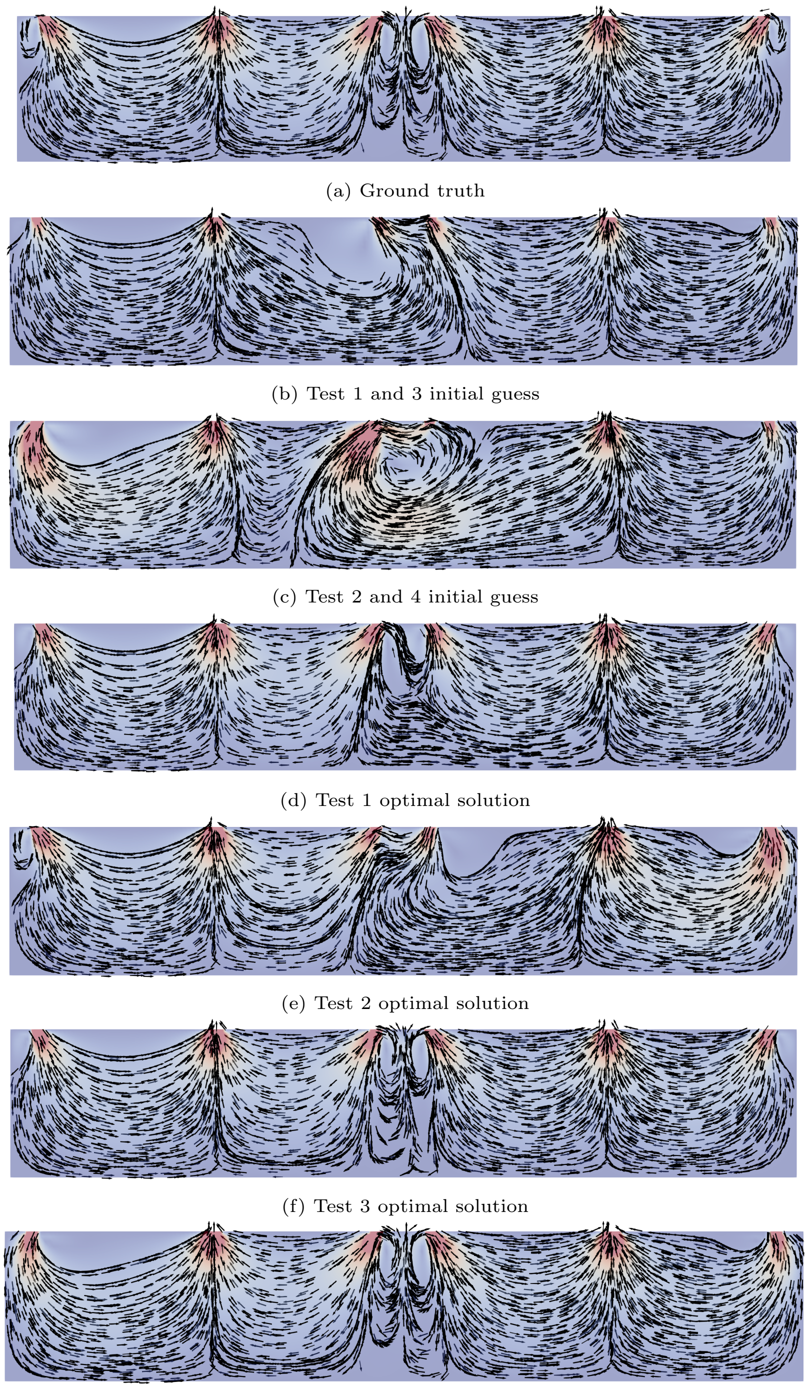
where the superscript 0 and * denote, respectively, the values related to the initial guess and the (near-) optimal solution obtained from the optimization.

We solve this inverse problem using the two variants of one-shot methods for one-shot method, i.e. Alg. 2 (OC) with $\bar{\beta} = 1000, \beta_{min} = 10$ and $\beta_{max} = 10000$, and Alg. 3 (OB) with $\alpha = 0.001$. The results are shown in Fig. 6. All the vertical velocities reach an approximate steady state after $L_o \approx 10000$, and this value is different from the initial guess. For this problem, the x-velocity of input 4 has almost no impact on the cost function, as is reflected in Fig. 6-d; the values of $\nabla_{V^y_{in}}$ are in general much larger than $\nabla_{V^x_{in}}$. Specifically, the value for $\nabla_{V^x_{in,4}}$ is negligible. This implies that the flow in region of interest I is mostly affected by incoming flow from inlets 1, 2, and 3.

As shown in Fig. 6(c), the BFGS-based Alg. 3 results in a lower cost function with a larger L_o. Since the step size for BFGS with normalized gradient is proportional to the design parameter, a typical value of $\alpha = 0.01 - 0.001$ can be used for the step size. However, for the curvature-based Alg. 2 (OC), the step size (or alternatively $\bar{\beta}$) is less intuitive and should be proportional to the gradient of the cost function, which is not available a priori and can depend on the initial conditions or regions of interest. At convergence for the BFGS cases, we obtain $R_J = 3.99 \times 10^{-5}$, $R_x = 0.009$, and $R_y = 0.0026$, which shows orders of magnitude of reduction of cost function and sensitivity values after optimization.

Next, we study the impact of the initial guess on the optimization results to assess the robustness of one-shot methods. To do this, we choose test 2 of Table 1. The steady-state solution for this case with an initial guess (without optimization) is shown in Fig. 5-c. The velocity field is quite different from the reference solution, and notably does not have vertical air curtain in the middle of the interior space. As a result, extensive mixing between occupied and unoccupied regions is evident. The optimization results are shown in Fig. 7. For the optimal solution using Alg. 3 (OB), we get $R_J = 2.67 \times 10^{-4}$, $R_y = 0.0018$, and $R_x = 8.59 \times 10^{-4}$, which again demonstrates significant progress towards the optimal solution. Similar reduction in the cost function is also observed for optimization initialized with several different initial guesses. Hence, both one-shot methods (OB) and (OC) seem to be robust to variations in the choice of initial guess, at least in a reasonable neighborhood.

The results shown in Figs. 6 and 7 demonstrate the minimal effect of horizontal inflow from inlet 4 on cost function when using region of interest I. Fig. 8 shows the results for test 3 that involves region of interest II extending over both the occupied and unoccupied zones. We note that the impact of inflow from inlets 2 and 3 becomes much more pronounced in this case.



(a) Ground truth

(b) Test 1 and 3 initial guess

(c) Test 2 and 4 initial guess

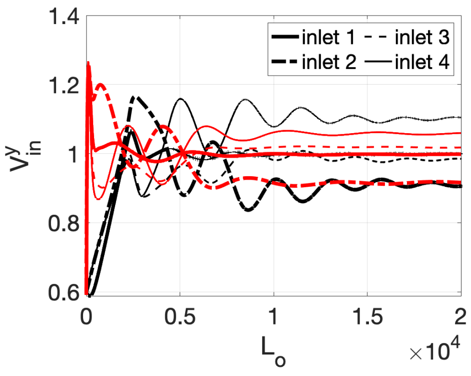
(d) Test 1 optimal solution

(e) Test 2 optimal solution

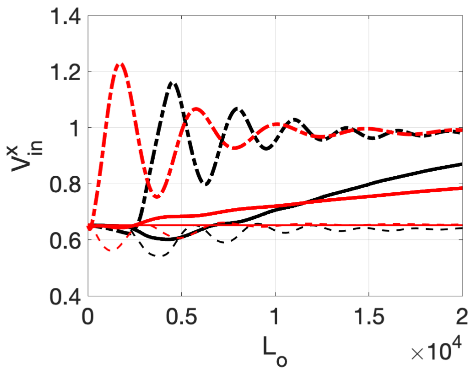
(f) Test 3 optimal solution

(g) Test 4 optimal solution

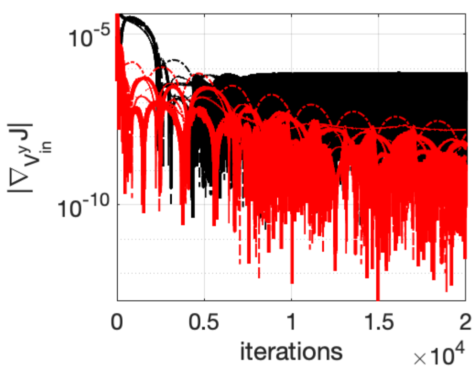
Figure 5: Streamlines (black) superposed on a colormap of the velocity magnitude for various tests, as described in the text. Panel a) shows the reference field for which the inverse problem seeks to find the corresponding inlet velocities. The optimal solutions are based on the one-shot (BFGS variant) method with normalized gradient, i.e. Alg. 3 (OB).



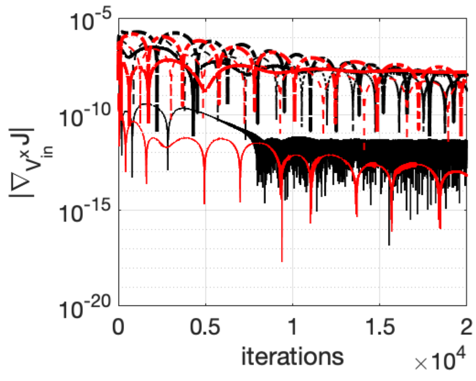
(a)



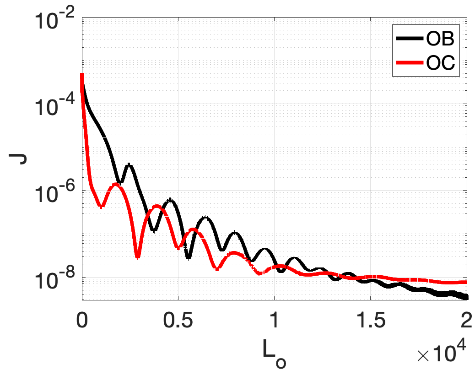
(b)



(c)

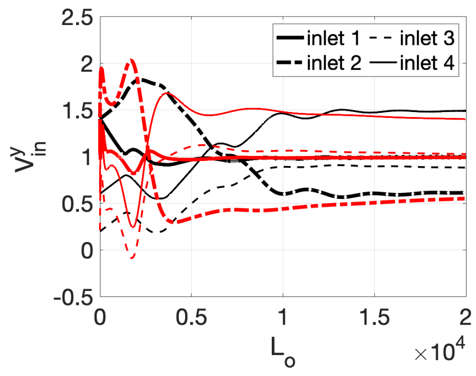


(d)

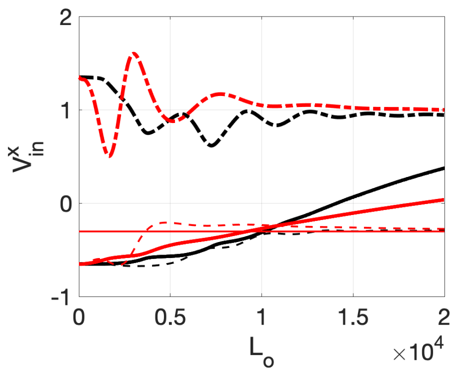


(e)

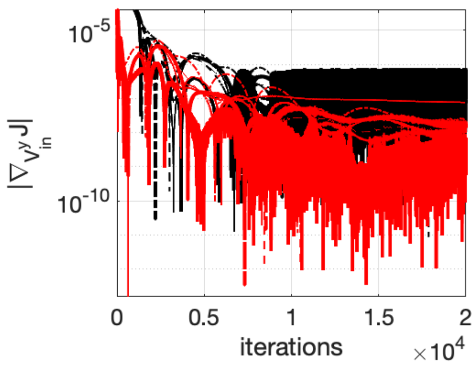
Figure 6: (a,b) Design parameters, (c,d) sensitivities and (e) cost function for test 1. Inlets 1, 2, 3 and 4 are denoted by thick solid, solid-dotted, dotted, and thin solid lines, respectively. The black curves are for the results obtained by Alg. 3 (OB) and the red curves are for the results obtained by Alg. 2 (OC).



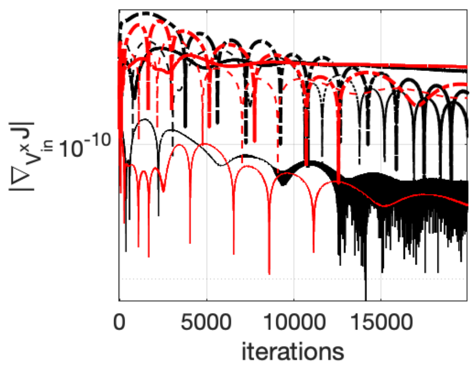
(a)



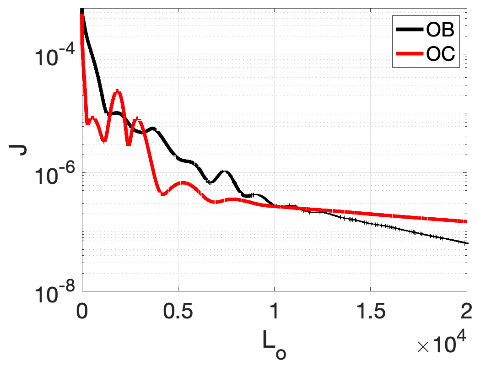
(b)



(c)



(d)



(e)

Figure 7: Same as Fig. 6 but for test 2.

If we use the test 1 step size $\bar{\beta} = 1000$ for test case 3, the OC Alg. 2 diverges. Hence, a new step size is required to make it work. On the other hand, the OB Alg. 3 converges for test case 3 using the same step size α used in test 1. This robustness of the OB Alg. 3 to changes in the parameters of the optimization problem implies that minimal user input is required for it to work across different problems. The optimal solution is shown in Fig. 5-f.

Another comparison between the OC algorithm with a new step size $\bar{\beta}$ where we get convergence, and the OB algorithm, for test case 3 is shown in Fig. 8. Similar results are obtained for test 4, and we omit discussing those results. The optimal solution for test 4 is however shown in Fig. 5-g. In both panels f and g of Fig. 5, the double counter-rotating vortices in the middle of the cavity are apparent. Hence, when the region of interest in the cost function extends across both the occupied zone and the unoccupied zone, this important feature of the flow field can be recovered.

In order to compare the performance of the one-shot method with the DAL method, we also carry out DAL-based optimization for test 1, using Alg. 1 (D). Results are shown in Fig. 9. We used $\alpha = 0.6$ with BFGS-type update for the preconditioner.

Using the one-shot method, the cost function is approximately 10^{-8} after $L_o \approx 14000$ iterations (or $n_o \approx 28000$ total operation counts for the direct and adjoint equations). On the other hand, the cost function computed by the DAL method, even after $L_D = 100$ loops, does not converge to such a low value. We note that that each DAL method 'loop' requires approximately $k_D = 5200$ and $k_a = 4100$ iterations on average to converge. Hence, even after $n_D = 93000$ operation counts for the DAL method, equivalent convergence to that of the one-shot method is not obtained. Finally, we note that the total operation count of one-shot method (i.e. n_o) is less than six times that of a single simulation of the direct equations (i.e. k_D). In terms of the computational cost, a typical direct simulation takes about 110.5s of CPU time. We ran the case in parallel mode on 10 Intel Xeon 2.40GHz CPUs. Using the one-shot method for test 1, the overall optimization took 722.84s, and hence the 'retardation factor', the ratio of the optimization computational cost to the simulation computational cost, is less than 7. However, for the preconditioned DAL method, the overall optimization cost for 100 DAL loops is 12238s.

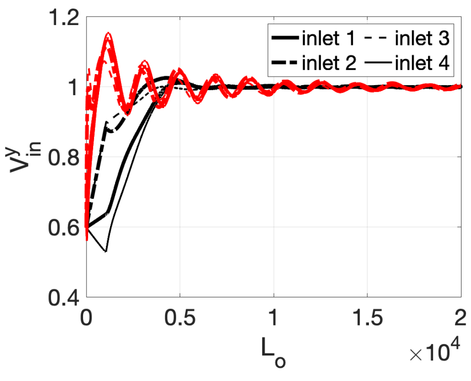
5. Conclusions and future work

We have numerically investigated the performance of variants of one-shot methods with two types of preconditioners for continuous adjoint-based PDE-constrained optimization. Consistent with previous results, we found that the choice of preconditioner has a significant impact on the convergence rate and stability of one-shot methods.

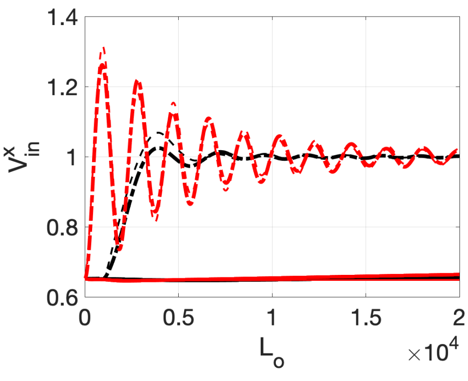
The validation of the algorithms was carried out using an analytically solvable laminar (plane Poiseuille) flow inverse problem. The performance of different one-shot variants was compared by solving a complex turbulent flow inverse problem, motivated by HVAC applications in the built environment. The robustness of the algorithms was verified by using several different initial guesses and problem regions of interest. All algorithms show robustness to the choice of cost function and initial guess.

We found that for various case studies, the BFGS based algorithm (OB) has several advantages. First, the only optimization parameter that needs to be tuned in this case is the step size for the gradient descent step. This can be chosen intuitively, e.g., to be a small fraction of the design parameters. Secondly, the stability of the method has been shown to be superior compared to the curvature-based one-shot algorithm (OC). The steady state optimal solution obtained by the OB algorithm has been found to have a smaller optimal value of cost function than the converged optimal solution of the OC algorithm.

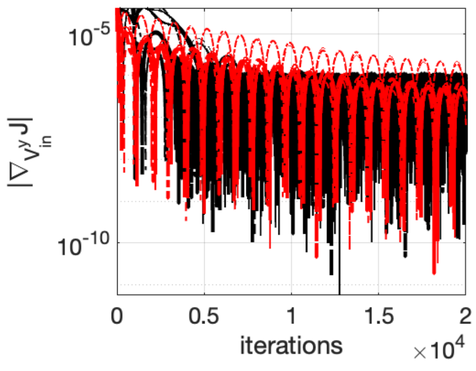
Finally, we have also compared the one-shot variants with the preconditioned DAL method to promote better one-shot variants for practical use in the fluid dynamics community. We found that the one-shot method can converge an order of magnitude faster than the DAL method. In fact, consistent with [20, 21, 40], the total operation count of the one-shot method is a single-digit multiple of the operation count required for a single solution of the direct (or governing) equations. The reduced computational cost associated with the appropriate one-shot method makes this approach feasible for increasingly challenging fluid mechanics applications.



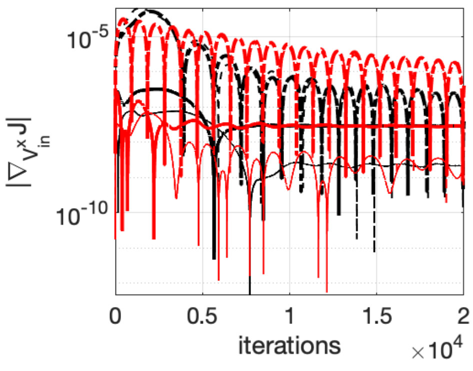
(a)



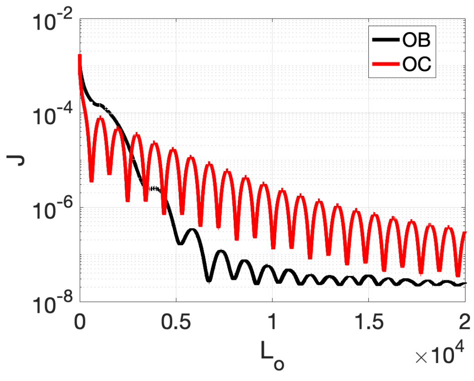
(b)



(c)



(d)



(e)

Figure 8: Same as Fig. 6 but for test 3. With $\bar{\beta} = 1000$ Alg. 2 (OC) diverges. The black curves denote the results for Alg. 3 (OB). The red curves denote the results for Alg. 2 (OC) with $\bar{\beta} = 100$.

An important application of this work is in airflow pattern optimization in the built environment. *Customized* airflow patterns in various regions of the room help avoid a (potentially undesirable) well-mixed environment, and instead allow for a focus on individual thermal comfort demands consistent with potential savings in energy consumption. A natural extension of this work would be to account for the temperature distribution by using Boussinesq equations. Such an extension is straightforward and all algorithms provided in Section 3 can be used for such a purpose.
For small size problems when $dim(u) \leq 50$, we have obtained very robust and efficient solutions with the BFGS-variant algorithm. We suspect that this would remain the case for intermediate problems when

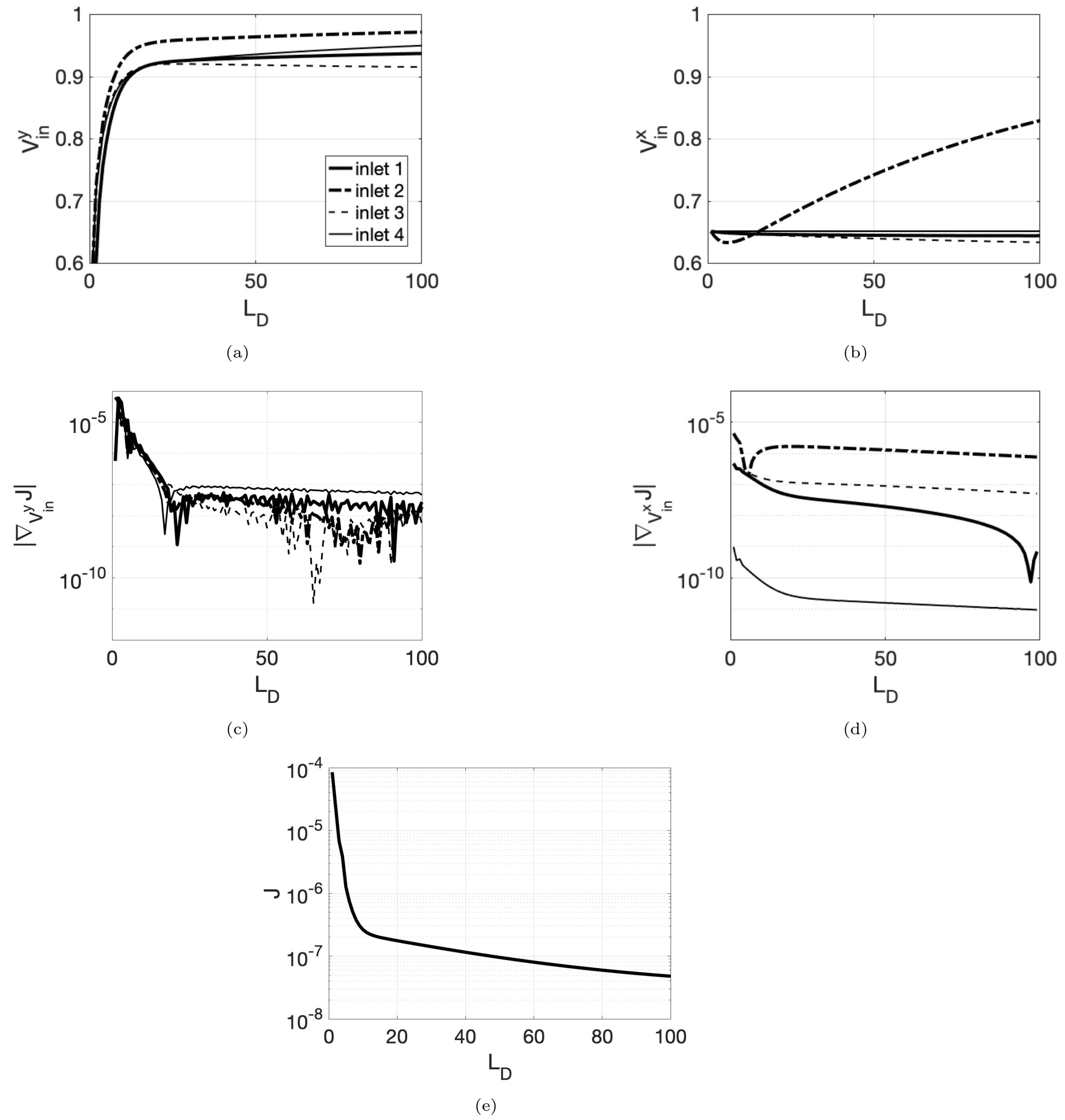


Figure 9: (a,b) Design parameters, (c,d) sensitivities, and (e) cost function for test 1 based on Alg. 1 (D), i.e. DAL method. Inlets 1, 2, 3 and 4 are denoted by thick solid, solid-dotted, dotted, thin solid lines, respectively. The design parameters $V_{in,i}^{y}$ and $V_{in,i}^{x}$ values are shown in panel a and b, respectively. A BFGS-tyoe preconditioner is used for the DAL method.

$dim(u) \leq 1000$ [52]. Most boundary control problems would fall in the above two categories. However, for larger scale problems, e.g. optimal initial perturbation and minimal seed problems [53], when $dim(u) \geq 1000$, limited-memory variants of BFGS, i.e. L-BFGS, should serve as a better choice. For such large scale problems, approximate Hessian matrices cannot be computed within a reasonable computational budget and, furthermore, there is no guarantee they are sparse. L-BFGS algorithms maintain implicit and compact approximation of such Hessian matrices by storing only a few representative vectors, and despite avoiding the construction of the full Hessian matrix, they often offer acceptable accuracy and rate of convergence

[30]. We believe that the algorithms presented in this work can be modified to use the L-BFGS algorithm for large scale problems, and intend to address this topic in the future.

References

References

[1] S. B. Hazra, Large-scale PDE-constrained optimization in applications, Vol. 49, Springer Science & Business Media, 2009.

[2] M. Fisher, J. Nocedal, Y. Trémolet, S. J. Wright, Data assimilation in weather forecasting: a case study in pde-constrained optimization, Optimization and Engineering 10 (3) (2009) 409–426.

[3] S. Nabi, N. Nishio, P. Grover, R. Matai, Y. Kajiyama, N. Kotake, S. Kameyama, W. Yoshiki, M. Iida, Improving lidar performance on complex terrain using cfd-based correction and direct-adjoint-loop optimization, in: Journal of Physics: Conference Series, Vol. 1452, IOP Publishing, 2020, p. 012082.

[4] J. W. Pearson, M. Stoll, Fast iterative solution of reaction-diffusion control problems arising from chemical processes, SIAM Journal on Scientific Computing 35 (5) (2013) B987–B1009.

[5] E. Oktay, H. Akay, O. Merttopcuoglu, Parallelized structural topology optimization and cfd coupling for design of aircraft wing structures, Computers & Fluids 49 (1) (2011) 141–145.

[6] E. F. Campana, D. Peri, Y. Tahara, F. Stern, Shape optimization in ship hydrodynamics using computational fluid dynamics, Computer methods in applied mechanics and engineering 196 (1-3) (2006) 634–651.

[7] T. Borrvall, J. Petersson, Topology optimization of fluids in stokes flow, International journal for numerical methods in fluids 41 (1) (2003) 77–107.

[8] S. Nabi, P. Grover, C. Caulfield, Nonlinear optimal control strategies for buoyancy-driven flows in the built environment, Computers & Fluids 194 (2019) 104313.

[9] S. Nabi, P. Grover, C.-c. P. Caulfield, Adjoint-based optimization of displacement ventilation flow, Building and Environment 124 (2017) 342–356.

[10] W. Liu, Q. Chen, Optimal air distribution design in enclosed spaces using an adjoint method, Inverse Problems in Science and Engineering 23 (5) (2015) 760–779.

[11] L. T. Biegler, O. Ghattas, M. Heinkenschloss, B. van Bloemen Waanders, Large-scale pde-constrained optimization: an introduction, in: Large-Scale PDE-Constrained Optimization, Springer, 2003, pp. 3–13.

[12] J. E. Peter, R. P. Dwight, Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches, Computers & Fluids 39 (3) (2010) 373–391.

[13] W. K. Anderson, V. Venkatakrishnan, Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation, Computers & Fluids 28 (4) (1999) 443–480.

[14] R. Lanzafame, M. Messina, Fluid dynamics wind turbine design: Critical analysis, optimization and application of bem theory, Renewable energy 32 (14) (2007) 2291–2305.

[15] 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.

[16] M. D. Gunzburger, Sensitivities, adjoints and flow optimization, International Journal for Numerical Methods in Fluids 31 (1) (1999) 53–78.

[17] D. Foures, C. Caulfield, P. J. Schmid, Optimal mixing in two-dimensional plane Poiseuille flow at finite Péclet number, Journal of Fluid Mechanics 748 (2014) 241–277.

[18] S. Guenther, N. R. Gauger, Q. Wang, Simultaneous single-step one-shot optimization with unsteady PDEs, Journal of Computational and Applied Mathematics 294 (2016) 12–22.

[19] V. Akçelik, G. Biros, O. Ghattas, J. Hill, D. Keyes, B. van Bloemen Waanders, Parallel algorithms for pde-constrained optimization, in: Parallel processing for scientific computing, SIAM, 2006, pp. 291–322.

[20] S. Hazra, V. Schulz, J. Brezillon, N. Gauger, Aerodynamic shape optimization using simultaneous pseudo-timestepping, Journal of Computational Physics 204 (1) (2005) 46–64.

[21] A. Griewank, Projected hessians for preconditioning in one-step one-shot design optimization, in: Large-Scale Nonlinear Optimization, Springer, 2006, pp. 151–171.

[22] T. Bosse, N. R. Gauger, A. Griewank, S. Günther, V. Schulz, One-shot approaches to design optimization, in: Trends in PDE Constrained Optimization, Springer, 2014, pp. 43–66.

[23] T. Bosse, L. Lehmann, A. Griewank, Adaptive sequencing of primal, dual, and design steps in simulation based optimization, Computational Optimization and Applications 57 (3) (2014) 731–760.

[24] T. Bosse, Augmenting the one-shot framework by additional constraints, Optimization Methods and Software 31 (6) (2016) 1132–1148.

[25] A. Walther, N. R. Gauger, L. Kusch, N. Richert, On an extension of one-shot methods to incorporate additional constraints, Optimization Methods and Software 31 (3) (2016) 494–510.

[26] S. Günther, N. R. Gauger, Q. Wang, A framework for simultaneous aerodynamic design optimization in the presence of chaos, Journal of Computational Physics 328 (2017) 387–398.

[27] R. Kerswell, C. Pringle, A. Willis, An optimization approach for analysing nonlinear stability with transition to turbulence in fluids as an exemplar, Reports on Progress in Physics 77 (8) (2014) 085901.

[28] B. Mohammadi, O. Pironneau, Analysis of the k-epsilon turbulence model (1993).

[29] E. Papoutsis-Kiachagias, K. Giannakoglou, Continuous adjoint methods for turbulent flows, applied to shape and topology optimization: industrial applications, Archives of Computational Methods in Engineering 23 (2) (2016) 255–299.

[30] J. Nocedal, S. Wright, Numerical optimization, Springer Science & Business Media, 2006.
[31] S. Rabin, C. Caulfield, R. Kerswell, Triggering turbulence efficiently in plane Couette flow, Journal of Fluid Mechanics 712 (2012) 244–272.
[32] H. G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Computers in physics 12 (6) (1998) 620–631.
[33] OpenFOAM - the open source computational fluid dynamics (cfd) toolbox, `http://openfoam.com`.
[34] S. V. Patankar, D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International journal of heat and mass transfer 15 (10) (1972) 1787–1806.
[35] P. K. Sweby, High resolution schemes using flux limiters for hyperbolic conservation laws, SIAM journal on numerical analysis 21 (5) (1984) 995–1011.
[36] J. H. Ferziger, M. Peric, A. Leonard, Computational methods for fluid dynamics (1997).
[37] S. B. Hazra, V. Schulz, Simultaneous pseudo-timestepping for pde-model based optimization problems, Bit Numerical Mathematics 44 (3) (2004) 457–472.
[38] S. Ta'asan, Pseudo-time methods for constrained optimization problems governed by PDE., Tech. rep., INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING HAMPTON VA (1995).
[39] E. Özkaya, N. R. Gauger, Automatic transition from simulation to one-shot shape optimization with Navier-Stokes equations, GAMM-Mitteilungen 33 (2) (2010) 133–147.
[40] A. Borzì, V. Schulz, Computational optimization of systems governed by partial differential equations, SIAM, 2011.
[41] A. Hamdi, A. Griewank, Properties of an augmented Lagrangian for design optimization, Optimization Methods & Software 25 (4) (2010) 645–664.
[42] E. Özkaya, N. R. Gauger, An efficient one-shot algorithm for aerodynamic shape design, in: New Results in Numerical and Experimental Fluid Mechanics VII, Springer, 2010, pp. 35–42.
[43] A. Griewank, A. Hamdi, E. Özkaya, Quantifying retardation in simulation based optimization, in: Optimization, simulation, and control, Springer, 2013, pp. 79–96.
[44] H. G. Bock, A. Potschka, S. Sager, J. P. Schlöder, On the connection between forward and optimization problem in one-shot one-step methods, in: Constrained Optimization and Optimal Control for Partial Differential Equations, Springer, 2012, pp. 37–49.
[45] A. Battermann, E. W. Sachs, Block preconditioners for KKT systems in PDE—governed optimal control problems, in: Fast solution of discretized optimization problems, Springer, 2001, pp. 1–18.
[46] A. Battermann, M. Heinkenschloss, Preconditioners for Karush-Kuhn-Tucker matrices arising in the optimal control of distributed systems, in: Control and Estimation of Distributed Parameter Systems, Springer, 1998, pp. 15–32.
[47] S. B. Hazra, V. Schulz, Simultaneous pseudo-timestepping for aerodynamic shape optimization problems with state constraints, SIAM Journal on Scientific Computing 28 (3) (2006) 1078–1099.
[48] R. Murray, B. Swenson, S. Kar, Revisiting normalized gradient descent: Fast evasion of saddle points, IEEE Transactions on Automatic Control 64 (11) (2019) 4818–4824.
[49] A. Önder, J. Meyers, Optimal control of a transitional jet using a continuous adjoint method, Computers & Fluids 126 (2016) 12–24.
[50] S. Nabi, M. Flynn, The hydraulics of exchange flow between adjacent confined building zones, Building and Environment 59 (2013) 76–90.
[51] S. Nabi, M. Flynn, Influence of geometric parameters on the eventual buoyancy stratification that develops due to architectural exchange flow, Building and Environment 71 (2014) 33–46.
[52] A. Skajaa, Limited memory BFGS for nonsmooth optimization, Master's thesis (2010).
[53] R. Kerswell, Nonlinear nonmodal stability theory, Annual Review of Fluid Mechanics 50 (2018) 319–345.