Computational and Applied Mathematics (2019) 38:168
https://doi.org/10.1007/s40314-019-0935-0

Adjoint computational methods for 2D inverse design of linear transport equations on unstructured grids

M. Morales-Hernández[1,2] · E. Zuazua[3,4,5]

Received: 26 March 2018 / Revised: 7 June 2019 / Accepted: 24 September 2019

Abstract
We address the problem of inverse design of linear hyperbolic transport equations in 2D heterogeneous media. We develop numerical algorithms based on gradient-adjoint methodologies on unstructured grids. While the flow equation is compulsorily solved by means of a second order upwind scheme so to guarantee sufficient accuracy, the necessity of using the same order of approximation when solving the sensitivity or adjoint equation is examined. Two test cases, including Doswell frontogenesis, are analysed. We show the convenience of using a low order method for the adjoint resolution, both in terms of accuracy and efficiency. An analytical explanation for this fact is also provided in the sense that, when employing higher order schemes for the adjoint problem, spurious high frequency numerical components slow down the convergence process.

Keywords Linear transport · Inverse design · Sensitivity · First and second order schemes · Gradient descent method · Adjoint

Mathematics Subject Classification 35L04 · 49M04 · 93B00

1 Introduction

Adjoint methods have been systematically associated to the optimal control design (Herty et al. 2015) and their applications to aerodynamics (Carpentieri et al. 2007; Castro et al.

Communicated by Cristina Turner.

✉ M. Morales-Hernández
mmorales@unizar.es

1 Fluid Mechanics, LIFTEC-EINA,CSIC-Universidad Zaragoza, Maria de Luna 3, 50018 Zaragoza, Spain

2 Department of Soil and Water, EEAD-CSIC, Avda. Montañana 1005, 50059 Zaragoza, Spain

3 DeustoTech, University of Deusto, 48007 Bilbao, Basque Country, Spain

4 Departamento de Matemáticas, Universidad Autónoma de Madrid, Cantoblanco, Madrid 28049, Spain

5 Facultad Ingeniería, Universidad de Deusto, Avda Universidades 24, 48007 Bilbao, Basque Country, Spain

2007; Giles and Pierce 2000; Ulbrich 2001). During the last decades, several works were oriented to develop a robust control theory based on the concepts of observability, optimality and controllability for linear and non-linear equations and systems of equations (Castro et al. 2008; Zuazua 2002, 2005, 2007). At the same time, and focusing on applications related to computational fluid dynamics, many contributions can be found in the literature concerning inverse design, parameter identification or optimization in general for engineering problems (Power et al. 2006) and in aeronautical applications in particular (Castro et al. 2007; Jameson 1988). The use of adjoint equations and gradient methods for this purposes is widely justified via the minimization of a functional or cost function (Huang and Ascher 2014; Nocedal and Wright 1999), resulting a suitable way of analysing the sensitivity of a complex dynamical system (Li and Petzold 2004).

Optimization methods and, in particular, the best representative gradient descent method, need some information about the gradient of the function to be minimized, which will be closest related to the resolution of the adjoint variables. Two trends or approaches have divided the research community when trying to solve backwards in time the adjoint equation or system of equations: the continuous and the discrete version. While in the continuous approach the set of adjoint equations is derived analytically and then discretized and solved by means of a certain numerical method, the discrete approach consist of a straightforward algebraic manipulation to achieve the discretized version of the adjoint equation from the original numerical methods used for the primitive flow equations. Although a large number of works can be found comparing both strategies Nadarajah and Jameson (2000); Peter and Dwight (2010), Giles and Pierce (2000) clarify the advantages and disadvantages of using one or another approach.

Since the discrete approach forces to use the discrete adjoint problem of the flow solver to numerically solve the adjoint equation, the continuous approach is adopted due to its valuable flexibility of using different solvers for the flow and adjoint equations. It is well known that problems where high frequencies play an important role such as the control of wave equation (Zuazua 2005; Ervedoza and Zuazua 2013; Glowinski 1992) or the shape design optimization (Nochetto et al. 1996) require a filtering or smoothing operator to eliminate the undesirable oscillations that appear when applying the discrete approach for the optimization method. In contrast, as subsequently detailed, the use of such operator can be avoided by adopting a low order numerical scheme for the adjoint equation in unsteady flows.

In this paper, the focus is put on the 2D linear scalar transport equation, that can be expressed in a conservative form as follows:

$$\frac{\partial u(x,y,t)}{\partial t} + \nabla \cdot (\mathbf{v}u) = 0, \quad u(x,y,0) = u_0, \tag{1}$$

where $\mathbf{v} = v(x,y)$ is a time-independent velocity field of propagation.

Given a target function $u^* = u^*(x,y)$, the aim in the 2D inverse design problem consists in finding u_0 such that $u(T) \sim u^*$ via the minimization of a functional J:

$$J(u_0) = \frac{1}{2}|u(T) - u^*|^2. \tag{2}$$

This problem can be easily addressed by simply solving the transport equation backwards in time because of the time reversibility of the model. But such a simple approach fails as soon as the model involves nonlinearities (leading to shock discontinuities) or diffusive terms, making the system time-irreversible.

$$\begin{tikzcd} u^k(0) \arrow[r, "\text{forward}"] & u^k(T) \arrow[d, dashed, "\sigma^k(T) = u^k(T) - u^*"] \\ \sigma^k(0) \arrow[dr, bend right] & \sigma^k(T) \arrow[l, "\text{backward}"'] \\ & \boxed{u^{k+1}(0) = u^k(0) - \varepsilon\sigma^k(0)} \end{tikzcd}$$

Fig. 1 Iteration of the 2D inverse design

We are thus interested in the development of gradient descent methods with the aid of the adjoint equation (solved backwards in time) that can be easily deduced

$$-\frac{\partial\sigma(x,y,t)}{\partial t} - \mathbf{v}\cdot\nabla\sigma = 0, \quad \sigma(T) = u(T) - u^*, \tag{3}$$

where $\sigma = \sigma(x, y, t)$ is the adjoint variable.

To achieve a good match with the continuous solutions, high order numerical schemes need to be used for the forward state Eq. (1). Here, we shall use a second order scheme. Our main objective is to test the convenience of using the same order of accuracy when solving the adjoint equation or, by the contrary, to employ a low order one. When implementing the gradient descent iterations, the numerical scheme employed for solving the adjoint equation determines the direction of descent. Hence, different solvers for the adjoint system provide different results that can be compared in terms of accuracy and efficiency.

As we shall see, and this will be illustrated in two examples, with a full geometric meaning, the use of a first order scheme is not only sufficient but, in fact, provides better results in terms of computational complexity (efficiency) and accuracy. Although this might seem surprising at first, it is a consequence of the fact that when employing a second order scheme for the adjoint problem, spurious high frequency numerical components slow down the convergence process.

Even in this linear purely hyperbolic model, in which nonlinear and diffusive effects have been ignored, when dealing with intricate geometric configurations either for the velocity field or for the shape of the target, building accurate and efficient numerical schemes is not an easy task. The main advantage of the gradient-adjoint methods developed here resides in its potential extrapolation to real problems with complicated non-linear fluxes, source and diffusive terms that cannot be addressed by simple backward resolution.

A gradient-adjoint iterative method is based on iterating a loop where the equation of state (flow equation) is solved in a forward sense while the adjoint equation, which is of hyperbolic nature as well, is solved backwards in time (see Fig. 1).

The adequate resolution of this loop is the main question addressed in this paper, paying attention not only to accuracy but also to reducing its computational complexity. One could expect that the choice of high order numerical methods to solve both the equation of state and the adjoint one should provide the best results in terms of accuracy, at the prize of a high computational cost. But this is not always true and it is possible to relax the necessity of using the same order of accuracy for the resolution of the adjoint equation, not only reducing considerably the computational time needed by the complete loop but also achieving the same order of accuracy on the approximation of the inverse design, which is our ultimate goal.

The outline of this paper is as follows: after this introduction, the 2D inverse design problem is analysed from a continuous point view, where the derivation of the adjoint equation and the gradient descent method are developed. The discrete version of the problem is then analysed. In particular, two schemes are presented, which will be the basis of the numerical experiments: a First Order Upwind (FOU) scheme and a Second Order Upwind (SOU) scheme, both implemented on unstructured grids. Afterwards, two test cases with analytical solution are used to evaluate the convenience of using one or another solver for the resolution of the adjoint equation, based on the accuracy and the computational time. The FOU scheme provides better results and an analytical explanation for this fact is also enclosed. This fact is illustrated and explained by further analytical and computational considerations. We close the article by pointing towards some possible extensions and directions for future research.

2 2D inverse design

2.1 Governing equations and derivation of the continuous adjoint equation

As stated in the introduction, we focus on the following conservative linear transport scalar equation with an heterogeneous time-independent vector field $\mathbf{v} = \mathbf{v}(x, y)$:

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v}u) = 0. \tag{4}$$

Here, and in the sequel $\cdot$ denotes the scalar product in $\mathbb{R}^2$.

Obviously, the solution $u = u(x, y, t)$ exists and it is unique and can be determined by means of the method of characteristics provided $\mathbf{v}$ is smooth enough (say, C^1). Thus, for all initial data $u_0 \in L^2(\mathbb{R}^2)$ there exists an unique solution in the class $C([0, T]; L^2(\mathbb{R}^2))$. Let us consider the inverse design problem: given $\Omega \subset \mathbb{R}^2$ and a target function $u^* = u^*(x, y)$ at $t = T$, to determine the initial condition u_0 such that $u(x, y, T) \equiv u^*(x, y)$ for all $(x, y) \in \Omega$.

From now on, the (x,y) dependence of the variables will be often omitted in the notation for the sake of clarity.

As mentioned above, the problem could be easily addressed solving the Eq. (4) backwards in time from the final datum u^* at $t = T$, to determine $u_0 = u(x, y, 0)$ exactly. But we are interested in addressing it from the point of view of optimal control. Let us, therefore, define the following functional, as a classical measure of the quadratic error with respect to the target function u^*:

$$J(u_0) = \frac{1}{2}\int_\Omega \left(u(T) - u^*\right)^2 \mathrm{d}S. \tag{5}$$

The derivation of the adjoint equation can be achieved by simply multiplying by $\sigma = \sigma(x, y, t)$ and integrating over $\Omega \times [0, T]$

$$\int_0^T \int_\Omega \sigma \left(\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v}u)\right) \mathrm{d}S\mathrm{d}t = 0. \tag{6}$$

Integrating by parts

$$\int_\Omega \sigma u|_0^T \mathrm{d}S - \int_0^T \int_\Omega u \frac{\partial \sigma}{\partial t} \mathrm{d}S\mathrm{d}t + \int_0^T \oint_{\partial\Omega} u\sigma \mathbf{v} \cdot \mathbf{n} \mathrm{d}l\mathrm{d}t - \int_0^T \int_\Omega u\mathbf{v} \cdot \nabla\sigma \mathrm{d}S\mathrm{d}t = 0, \tag{7}$$

where $\partial\Omega$ is the boundary and $\mathbf{n}$ is the outward normal direction to the domain Ω, respectively.

It is feasible to take Gateaux derivatives with respect to the variable u in this identity getting:

$$\int_{\Omega} \sigma \delta u|_0^T \mathrm{d}S - \int_0^T \int_{\Omega} \delta u \frac{\partial \sigma}{\partial t} \mathrm{d}S\mathrm{d}t + \int_0^T \oint_{\partial\Omega} \delta u \sigma \mathbf{v} \cdot \mathbf{n} \mathrm{d}l\mathrm{d}t - \int_0^T \int_{\Omega} \delta u \mathbf{v} \cdot \nabla\sigma \mathrm{d}S\mathrm{d}t = 0. \tag{8}$$

Gathering appropriately the terms:

$$\int_0^T \int_{\Omega} \delta u \overbrace{\left(-\frac{\partial \sigma}{\partial t} - \mathbf{v} \cdot \nabla\sigma\right)}^{(\sharp)} \mathrm{d}S\mathrm{d}t + \int_{\Omega} \sigma \delta u|_0^T \mathrm{d}S + \overbrace{\int_0^T \oint_{\partial\Omega} \delta u \sigma \mathbf{v} \cdot \mathbf{n} \mathrm{d}l\mathrm{d}t}^{(\sharp\sharp)} = 0. \tag{9}$$

Let us select $(\sharp) = 0$

$$-\frac{\partial \sigma}{\partial t} - \mathbf{v} \cdot \nabla\sigma = 0 \tag{10}$$

hence (9) becomes

$$\int_{\Omega} [\sigma(T)\delta u(T) - \sigma(0)\delta u(0)]\, \mathrm{d}S + (\sharp\sharp) = 0. \tag{11}$$

Assuming $\sigma(T) = u(T) - u^*$

$$\int_{\Omega} (u(T) - u^*)\delta u(T)\mathrm{d}S + (\sharp\sharp) = \int_{\Omega} \sigma(0)\delta u(0)\mathrm{d}S. \tag{12}$$

This derivation led to the adjoint equation $(\sharp)$:

$$-\frac{\partial \sigma}{\partial t} - \mathbf{v} \cdot \nabla\sigma = 0 \quad \sigma(T) = u(T) - u^*, \tag{13}$$

$\sigma = \sigma(x, y, t)$ being the adjoint variable. This equation will measure the sensitivity of the solution to changes in the initial condition. It is worth mentioning that the adjoint equation is solved backwards in time (from $t = T$ to $t = 0$). In fact, system (13) is well-posed if and only if the original system is well posed in the forward sense.

The term $(\sharp\sharp)$ includes the boundary conditions. The computation of the sensitivity of the functional and the adjoint system when the PDE model is considered in the whole space (in the absence of boundaries) is straightforward. However, when the model and the cost functional is considered in a bounded domain Ω, suitable boundary conditions need to be imposed for both the state and the adjoint system. In that case Dirichlet null boundary conditions are imposed in complementary subsets of the boundary depending on whether the characteristics are incoming or outgoing. Note that the characteristics are the same for both models, but that the sense of time is reversed from one to the other, which explains the necessity of shaping the role of the boundary subsets to impose the boundary conditions. Therefore, let us split the set $\partial\Omega = \partial\Omega^+ \cup \partial\Omega^-$ where

$$\partial\Omega^+ = \{(x, y) \in \partial\Omega \mid \mathbf{v} \cdot \mathbf{n} \geq 0\}, \quad \partial\Omega^- = \{(x, y) \in \partial\Omega \mid \mathbf{v} \cdot \mathbf{n} < 0\}. \tag{14}$$

Taking the first variation of the boundary condition for the state equation (4) gives:

$$\delta u(x, y, t) = 0 \quad \forall (x, y) \in \partial\Omega^-. \tag{15}$$

Additionally, adjoint equation (13) requires imposing adequate boundary conditions to avoid the problem to be overdetermined when solving it backwards in time:

$$\sigma(x, y, t) = 0 \quad \forall (x, y) \in \partial\Omega^+. \tag{16}$$

Therefore, the term ($\sharp\sharp$) vanishes. On the other hand, it is also feasible to take the first variation of J with respect to u_0 in (5)

$$\delta J(u_0) = \int_\Omega \delta u(T) \left(u(T) - u^*\right) \mathrm{d}S. \tag{17}$$

Finally,

$$\delta J(u_0) = \int_\Omega (u(T) - u^*)\delta u(T)\mathrm{d}S = \int_\Omega \sigma(0)\delta u(0)\mathrm{d}S. \tag{18}$$

2.2 Gradient descent method and classical result of convergence

We apply a classical steepest descent algorithm to minimize the cost functional J. Given any initilization value for u_0, say $u_0^0 \equiv 0$, for instance, the iteration can be written as follows:

$$u_0^{k+1} = u_0^k - \varepsilon\, \nabla J^k, \tag{19}$$

for $k \geq 0$, where

$$\nabla J^k = \frac{\delta J(u_0^k)}{\delta u_0}.$$

From (18):

$$\nabla J^k = \frac{\delta J(u_0^k)}{\delta u_0} = \sigma^k(0). \tag{20}$$

In practice, the numerical scheme used for the resolution of the adjoint equation, determines the descent direction $\sigma^k(0)$ in (19) that governs the next initial condition u_0^{k+1} for the forward state equation.

Here, we take the step ε to be constant, independent of k. Other, more sophisticated and optimal choices are also possible. But the computational results we achieved are essentially the same. For the sake of simplicity we thus take ε independent of k. As it is well known (Ciarlet 1982), assuming that

$$|\nabla J(u_1) - \nabla J(u_2)| \leq M|u_1 - u_2| \quad \langle \nabla J(u_1) - \nabla J(u_2), u_1 - u_2\rangle \geq m|u_1 - u_2|^2 \tag{21}$$

one has the following estimate on the convergence of the gradient iteration

$$|u_0^{k+1} - u_0^*|^2 \leq (1 - 2\varepsilon m + \varepsilon^2 M^2)|u^k - u_0^*|^2, \tag{22}$$

where u_0^* is the exact initial datum driving the solution of the state equation to the target u^*. Therefore, the convergence is guaranteed if and only if

$$1 - 2\varepsilon m + \varepsilon^2 M^2 < 1 \tag{23}$$

or equivalently

$$\varepsilon < \frac{2m}{M^2}. \tag{24}$$

The velocity of convergence is then:

$$|u_0^k - u_0^*| \leq (1 - 2\varepsilon m + \varepsilon^2 M^2)^{k/2} |u_0 - u_0^*| \tag{25}$$

The problem under consideration enters in the class of quadratic minimization problems $Ax = b$, A being symmetric and positive definite that can be solved via minimisation of the functional

$$J(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle. \tag{26}$$

If A is ill-conditioned, $m << M$, and ε should be very small to ensure the convergence.

3 Numerical approach to the continuous 2D inverse design problem

3.1 Numerical schemes for the flow equation

To obtain a numerical solution based on a finite volume approach, (4) is integrated in $\Omega_i \times [t^n, t^{n+1}]$ where Ω_i represents each computational cell of the domain and $t^{n+1} = t^n + \Delta t$

$$\int_{t^n}^{t^{n+1}} \int_{\Omega_i} \left(\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v} u) \right) \mathrm{d}S \mathrm{d}t = 0. \tag{27}$$

Using the Gauss divergence theorem

$$\int_{\Omega_i} u(x, y, t^{n+1}) \mathrm{d}S - \int_{\Omega_i} u(x, y, t^n) \mathrm{d}S + \int_{t^n}^{t^{n+1}} \oint_{\partial \Omega_i} \mathbf{f} \cdot \mathbf{n} \, \mathrm{d}m \, \mathrm{d}t = 0, \tag{28}$$

where $\mathbf{f} = \mathbf{v}u$ is the flux, $\partial \Omega_i$ denotes the boundary of Ω_i and $\mathbf{n} = (n_x, n_y)$ its unit normal vector. The last contour integral in (28) can be replaced by the sum of the fluxes defined at each edge k of length l_k:

$$\int_{\Omega_i} u(x, y, t^{n+1}) \mathrm{d}S - \int_{\Omega_i} u(x, y, t^n) \mathrm{d}S + \int_{t^n}^{t^{n+1}} \sum_{k=1}^{N_E} \mathbf{f}_k \cdot \mathbf{n}_k l_k \mathrm{d}t = 0, \tag{29}$$

where N_E is the number of edges of each cell i ($N_E = 3$ is for triangular grids) with neighbouring cells j_k. Figure 2 clarifies the meaning of each variable.

In this work two numerical schemes are proposed: a first order upwind scheme and a second order upwind scheme based on MUSCL-Hancock approach, which are described afterwards.

3.1.1 First Order Scheme (FOU)

The FOU scheme can be formulated by considering the spatially-averaged value of the variable $u(x, y, t)$ at each cell i with area A_i

$$u_i^n = \frac{1}{A_i} \int_{\Omega_i} u(x, y, t^n) \, \mathrm{d}\Omega \tag{30}$$

and the time-averaged value at each interface k

$$\mathbf{f}_k^{n,*} \mathbf{n}_k = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} (\mathbf{f}_k \cdot \mathbf{n}_k) \, \mathrm{d}t. \tag{31}$$

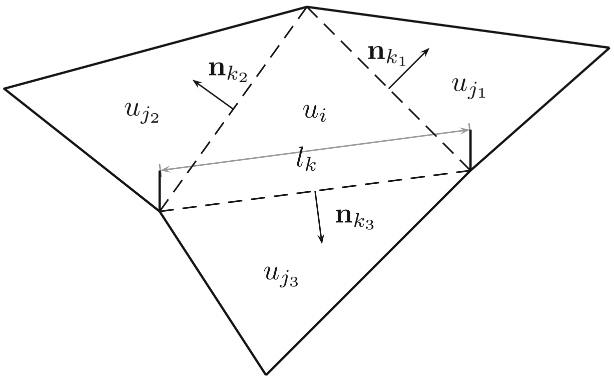

Fig. 2 Sketch of the 2D finite volume approach

Therefore, from (28), the numerical scheme can be written in terms of the numerical fluxes $\mathbf{f}_k^{n,*}$:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{A_i}\sum_{k=1}^{N_E} \mathbf{f}_k^{n,*}\mathbf{n}_k l_k. \tag{32}$$

The choice of the numerical fluxes in (32) will determine the diverse numerical schemes described in the literature (Toro 2009; Godlewski and Raviart 1996).

There exists an equivalent way of formulating a conservative scheme using the flux difference splitting procedure. The numerical fluxes $\mathbf{f}_k^*$ in (32) can be expressed as follows in terms of an upwind difference:

$$\mathbf{f}_k^*\mathbf{n}_k = (\mathbf{f}_k \cdot \mathbf{n}_k) + (\delta\mathbf{fn})_k^-, \tag{33}$$

where

$$(\delta\mathbf{fn})_k^{\pm} = \frac{(\mathbf{f}\cdot\mathbf{n})_j \pm (\mathbf{f}\cdot\mathbf{n})_i}{2}. \tag{34}$$

According to (Godlewski and Raviart 1996), the following property is satisfied:

$$\sum_{k=1}^{N_E} \mathbf{n}_k l_k = 0 \tag{35}$$

hence (32) is rewritten using (35) to achieve the flux difference formulation

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{A_i}\sum_{k=1}^{N_E} (\delta\mathbf{fn})_k^{n,-} l_k \tag{36}$$

whose meaning is simple: cell i is updated from time t^n to time t^{n+1} according to the in-going contributions arriving from the neighbouring interfaces. The time step size is restricted by the CFL condition (Courant et al. 1928). In the case of triangular grids, this expression is (Morales-Hernández et al. 2013):

$$\Delta t = \text{CFL}\ \min_k \left\{ \frac{\min(\chi_i, \chi_j)}{|\mathbf{v}\cdot\mathbf{n}|_k^n} \right\} \quad \text{CFL} \le 0.5, \tag{37}$$

where

$$\chi_i = \frac{A_i}{\max\limits_{k=1,N_E} l_k}. \tag{38}$$

3.1.2 Second Order Scheme (SOU)

We shall also use a SOU scheme, based on the MUSCL-Hancock approach (van Leer 1979). Starting from the formulation of the FOU scheme, a two-step algorithm is adopted to get a better accuracy. First of all, the solution at each cell i is linearly reconstructed using the gradient vectors $\mathbf{L}_i$ and the information provided by the three neighbouring cells j_1, j_2 and j_3 (see Fig. 2)

$$\mathbf{L}_i = \mathbf{J}\begin{pmatrix} u_{j_2} - u_{j_1} \\ u_{j_3} - u_{j_1} \end{pmatrix}, \tag{39}$$

where

$$\mathbf{J} = \frac{1}{(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)}\begin{pmatrix} y_3-y_1 & -y_2+y_1 \\ -x_3+x_1 & x_2-x_1 \end{pmatrix} \tag{40}$$

and (x_1, y_1), (x_2, y_2) and (x_3, y_3) are the centroid coordinates of cells j_1, j_2 and j_3, respectively. The limited slopes $\overline{\mathbf{L}}_i$ are obtained by means of the so-called flux limiter Φ

$$\overline{\mathbf{L}}_i = \min_{k=1,2,3} \Phi(\alpha_k)\mathbf{L}_i, \tag{41}$$

where the values α_k are computed as in Hubbard (1999). In this work, the limiter proposed by Sweby (1984) is used:

$$\Phi(r) = \max[0, \min(\beta r, 1), \min(r, \beta)] \quad 1 \le \beta \le 2. \tag{42}$$

Therefore, it is possible to obtain the values at the edges (capital subscript) according to the limited slopes and the position vectors $\mathbf{r}$ from the middle point M of the cell edge p to the centroid of the cell i as follows:

$$u_{Ip} = u_i + r_{iM}\overline{\mathbf{L}}_i. \tag{43}$$

Figure 3 shows a sketch of the variables and the linear reconstruction for the MUSCL-Hancock approach.
Therefore, for each interface p sharing cells i and j, the quantities u_{Ip} and u_{Jp} can be defined representing a suitable interpolation to the edge p from cells i and j, respectively. Once the values are reconstructed at each interface, ensuring that

$$\min(u_i, u_j) \le u_{Ip}, u_{Jp} \le \max(u_i, u_j) \tag{44}$$

an intermediate step between time t^n and t^{n+1} is required to achieve a second order accurate scheme in space and time according to the wall-cell contributions:

$$u_{Ip}^{n+1/2} = u_{Ip}^n - \frac{\Delta t}{2A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{Ik,i}^n l_k. \tag{45}$$

A second step is then computed, regarding not only for the wall-cell contributions but also for the wall-wall contributions at both sides of each wall

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{J,I}^{n+1/2,-} l_k - \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{Ik,i}^{n+1/2} l_k, \tag{46}$$

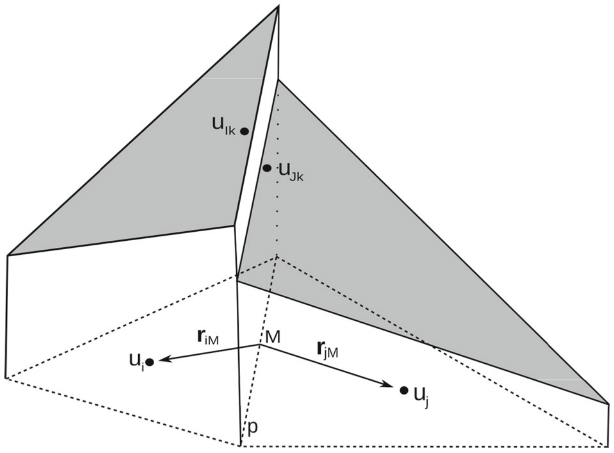

Fig. 3 MUSCL-Hancock reconstruction

where $\delta \mathbf{f}_{J,I}^{n+1/2} = f_J^{n+1/2} - f_I^{n+1/2}$ and $\delta \mathbf{f}_{Ik,i}^{n+1/2} = \mathbf{f}_{Ik}^{n+1/2} - \mathbf{f}_i^n$. It is worth noting that the wall-wall contributions are formulated using an upwind philosophy while the wall-cell contributions are written as a central term.

3.2 Adjoint numerical schemes

To apply the same numerical techniques described above for the flow equation, the adjoint equation:

$$-\frac{\partial \sigma}{\partial t} - \mathbf{v} \cdot \nabla \sigma = 0 \quad \sigma(T) = u(T) - u^*, \tag{47}$$

has to be expressed in a conservative divergence form as follows:

$$-\frac{\partial \sigma}{\partial t} - \nabla \cdot (\mathbf{v}\sigma) + \sigma \nabla \cdot \mathbf{v} = 0 \quad \sigma(T) = u(T) - u^*, \tag{48}$$

where an extra term related to the divergence of the velocity field appears. To solve numerically this equation, we shall consider the term $\sigma \nabla \cdot \mathbf{v}$ as a source term and the same procedure as in (28) can be applied. Assuming the backward resolution of this adjoint equation, the following linearization in time is considered:

$$\int_{t^n}^{t^{n+1}} \int_{\Omega} \sigma \nabla \cdot \mathbf{v} \, \mathrm{d}S \mathrm{d}t \simeq \int_{\Omega} (\sigma \nabla \cdot \mathbf{v})^{n+1} \, \mathrm{d}S. \tag{49}$$

This assumption allow us to formulate the FOU and SOU schemes for the adjoint equation in an analogous way to those proposed for the flow equation. However, it is necessary to take care about the signs in the upwind discretization in view of the fact that the sense of time is to be reversed. At the end, the FOU scheme for the adjoint equation is given by:

$$\begin{aligned}\sigma_i^n &= \sigma_i^{n+1} + \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_k^{n+1,+} l_k - \sigma_i^{n+1}\frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{vn})_k^{n+1,+} l_k \\ &= \sigma_i^{n+1}\left(1 - \underbrace{\frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{vn})_k^{n+1,+} l_k}_{\text{DV}}\right) + \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_k^{n+1,+} l_k, \end{aligned} \tag{50}$$

while the SOU scheme is formulated in two steps as follows:

$$\sigma_{Ip}^{n+1/2} = \sigma_{Ip}^{n+1}\left(1 - \underbrace{\frac{\Delta t}{2A_i}\sum_{k=1}^{N_E}(\delta \mathbf{vn})_{Ik,i}^{n+1} l_k}_{\text{DV}}\right) + \frac{\Delta t}{2A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{Ik,i}^{n+1} l_k \tag{51}$$

$$\begin{aligned}\sigma_i^n = \sigma_i^{n+1}&\left(1 - \underbrace{\frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{vn})_{J,I}^{n+1,+} l_k - \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{vn})_{Ik,i}^{n+1} l_k}_{\text{DV}}\right) \\ &+\frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{J,I}^{n+1/2,+} l_k + \frac{\Delta t}{A_i}\sum_{k=1}^{N_E}(\delta \mathbf{fn})_{Ik,i}^{n+1/2} l_k, \end{aligned} \tag{52}$$

where $\delta \mathbf{f}_{Ik,i}^{n+1/2} = \mathbf{f}_{Ik}^{n+1/2} - \mathbf{f}_i^{n+1}$. The terms underbraced with DV are related to the divergence of the velocity field and they vanish in the following test cases.

4 Test cases

The use of a SOU scheme for the flow equation (forward) is mandatory to achieve the best accuracy since the FOU scheme is unable to capture the detail for the equation of state when trying to reach the target function. A includes the comparison between the FOU and SOU schemes for the forward simulation associated to the test cases presented in this section. The issue of using one or another approach for solving the adjoint equation and, consequently, the gradient direction approach, is worth analysing.

In both of them the vector field generating the dynamics is rather heterogeneous and produces a significant transformation of the initial profiles under the evolution. Furthermore, the final targets correspond to continuous evolutions starting up from an initial datum with sharp edges. Thus, even if the problems under consideration are linear, it is hard to achieve accurate results. To check the efficiency and accuracy of the main two numerical schemes under consideration we take advantage of having an analytical expression of the exact solution.
Furthermore, it is important to mention that the velocity fields in both test cases are divergence-free. The state equation can the be written similarly in divergence or non-divergence form:

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v}u) = \frac{\partial u}{\partial t} + (\nabla \cdot \mathbf{v})u = 0, \tag{53}$$

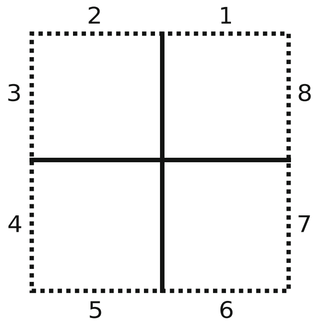

Fig. 4 Boundary regions for both test cases

because

$$\nabla \cdot \mathbf{v} \equiv 0,$$

in the two examples under consideration. Since both test cases are considered over a square domain, eight different boundaries can be distinguished, referenced in Fig. 4.
From now on, by the exact solution we understand the projection or restriction of the analytical solution on the computational mesh.

Additionally, as mentioned in Sect. 2.2, the step size ε is set as a constant at the beginning of each iteration. However, if the convergence is not guaranteed, i.e., $J^{k+1} >= J^k$, the step size is halved until $J^{k+1} < J^k$. Therefore, the stopping criteria for the iterative algorithm are:

1. To reach the maximum number of iterations
2. To achieve the maximum allowable error of the functional, in this case, 10^{-9}
3. To descend below $\varepsilon_{\min} = 10^{-6}$ when halving the step size ε trying to achieve convergence.

4.1 Test case 1: Doswell frontogenesis

The first 2D test case was first proposed (in a forward sense) in Doswell (1984). It symbolizes the presence of horizontal temperature gradients and fronts in the context of meteorological dynamics. The original problem consists of a computational domain $\Omega = [-5, 5]^2$ in which a front defined by the following initial condition:

$$u(x, y, 0) = \tanh\left(\frac{y}{\delta}\right) \tag{54}$$

is advected under the velocity field given by

$$\mathbf{v} = (-y\ g(r), x\ g(r)) \quad g(r) = \frac{1}{r}\overline{v}\ \mathrm{sech}^2(r)\tanh(r) \quad \overline{v} = 2.59807, \tag{55}$$

where $r = \sqrt{x^2 + y^2}$ and δ is a constant representing the sharpness of the front for the initial condition (54). The exact solution on the whole space $\mathbb{R}^2$ has the following explicit form:

$$u(x, y, t) = \tanh\left(\frac{y\cos(gt) - x\ \sin(gt)}{\delta}\right). \tag{56}$$

This allows us to formulate the problem of inverse design. Given the exact value of the solution at time $t = T$, i.e. giving (56) as a target function at $t = T$, we aim to recover by a

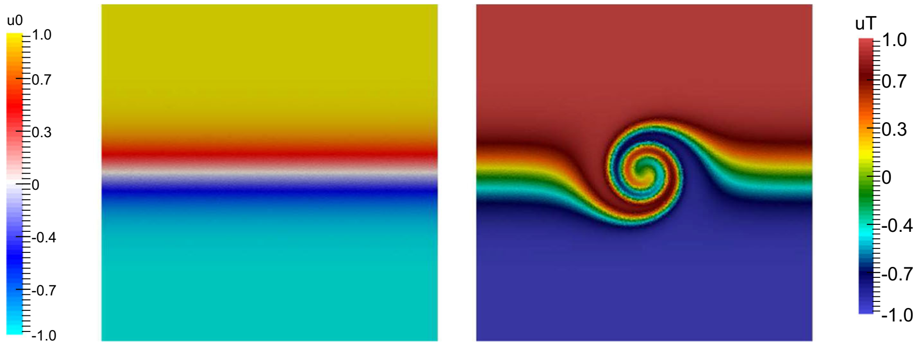

Fig. 5 Doswell frontogenesis: exact initial condition (left) and target function (right) for the inverse problem

computational version of the gradient-adjoint methodology described above, an accurate and efficient approximation of the initial datum (54). According to the velocity field, boundary conditions are imposed on regions 2, 4, 6 and 8 (see Fig. 4) for the state equation. Therefore, the boundary condition (16) for the adjoint equation will be imposed on regions 1, 3, 5 and 7.

In our numerical experiments we take the values $\delta = 1$ and $T = 4s$. The exact initial condition and the exact target are shown in Fig. 5 (left and right, respectively). As can be seen, the target presents a vortex-type profile, due to the heterogeneous geometry of the velocity field.

Our goal is to build a numerical method capable of predicting accurately and efficiently an approximation of (54) out of the target (56) by means of a numerical version of the optimisation algorithm above.

The domain is discretized in 39998 unstructured triangles using *Triangle* (Shewchuk 2002) in a uniform Delaunay mesh. The initial condition for the first iteration in (19) is $u^0(x, y, 0) = 0$. Both the forward and backward resolution are parallelized with OPENMP in 4 cores. The maximum number of iterations for the gradient method is set to 75, a constant step $\varepsilon = 0.5$ is selected and the Courant number is set to $CFL = 0.5$.

The numerical results are shown in Figs. 6, 7 and 8 for the target $u(x, y, T)$, initial condition $u(x, y, 0)$ and adjoint $\sigma(x, y, 0)$, respectively, at the last iteration of the gradient method. On the left side the ∇J^{FOU} approach is displayed while the ∇J^{SOU} resolution is illustrated on the right side.
To complete the qualitative results, the error computed as the difference between the exact and the numerical approximation is displayed in Figs. 9 and 10 for the target $u(x, y, T) - u^*(x, y)$ and the initial datum $u(x, y, 0) - u^e(x, y)$.

Quantitative results can be obtained by means of the computation of L_2 and L_∞ norms, not only with respect to the target function but also to the known exact initial condition u^e in order to evaluate the results achieved by ∇J^{FOU} and ∇J^{SOU}:

$$L_2(u^T) = \sqrt{\sum_i^N (u(x_i, T) - u^*(x_i))^2} \quad L_\infty(u^T) = \max_i |u(x_i, T) - u^*(x_i)|$$

$$L_2(u^0) = \sqrt{\sum_i^N (u(x_i, 0) - u^e(x_i))^2} \quad L_\infty(u^0) = \max_i |u(x_i, 0) - u^e(x_i)|. \tag{57}$$

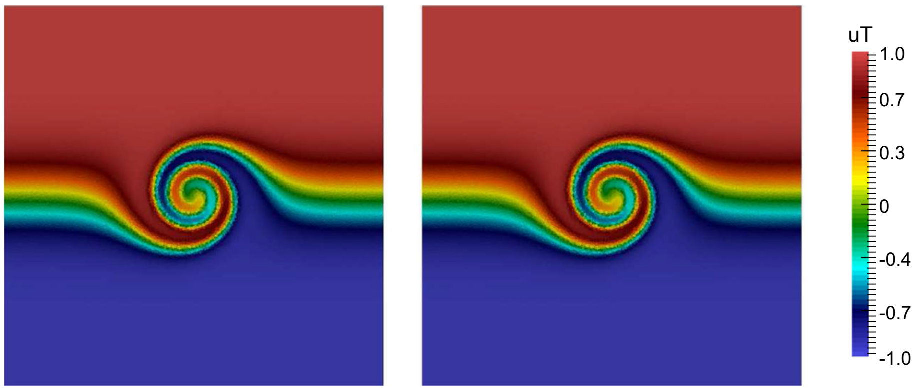

Fig. 6 Doswell frontogenesis: numerical target $u(x, y, T)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

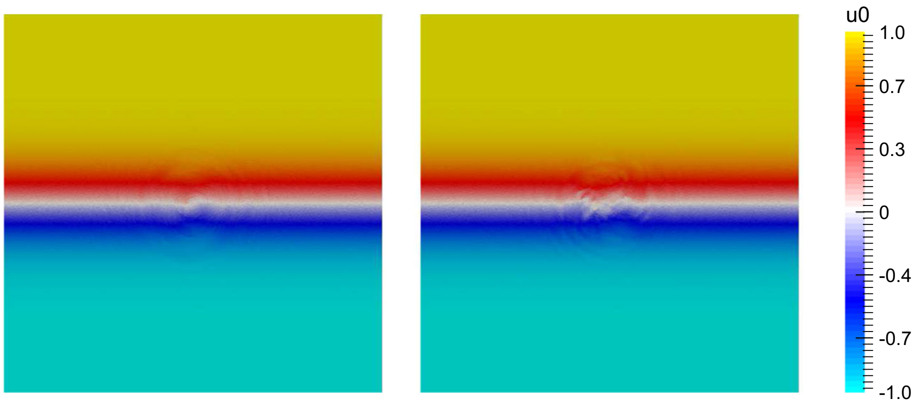

Fig. 7 Doswell frontogenesis: numerical initial condition $u(x, y, 0)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

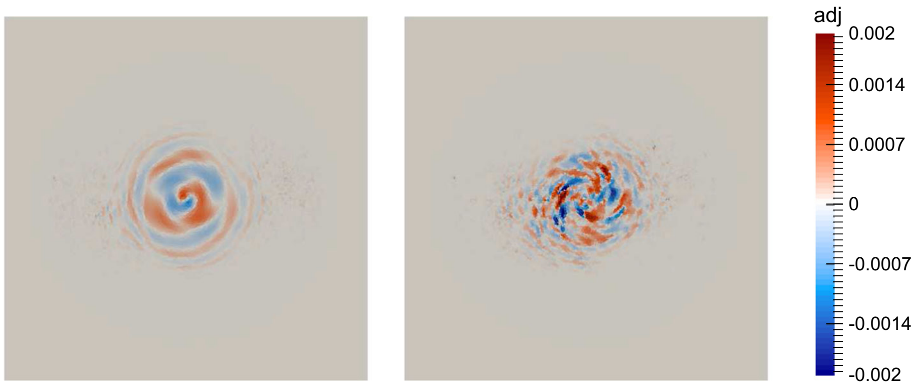

Fig. 8 Doswell frontogenesis: numerical adjoint variable $\sigma(x, y, 0)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

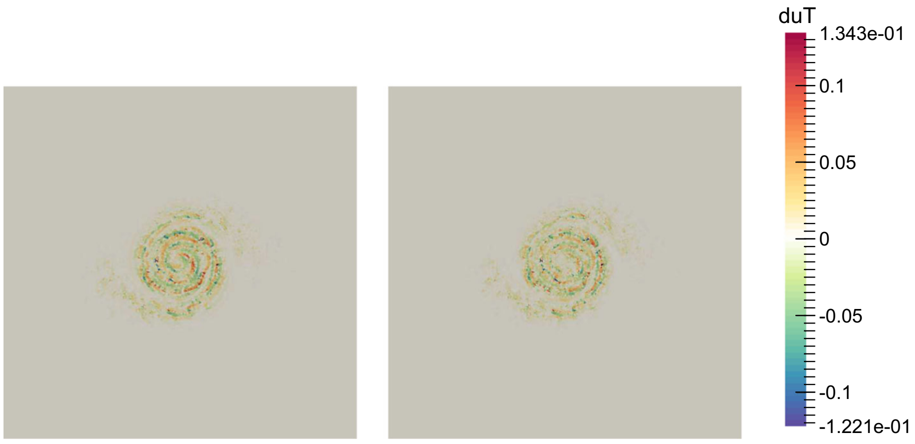

Fig. 9 Doswell frontogenesis: target error $u(x, y, T) - u^*(x, y)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

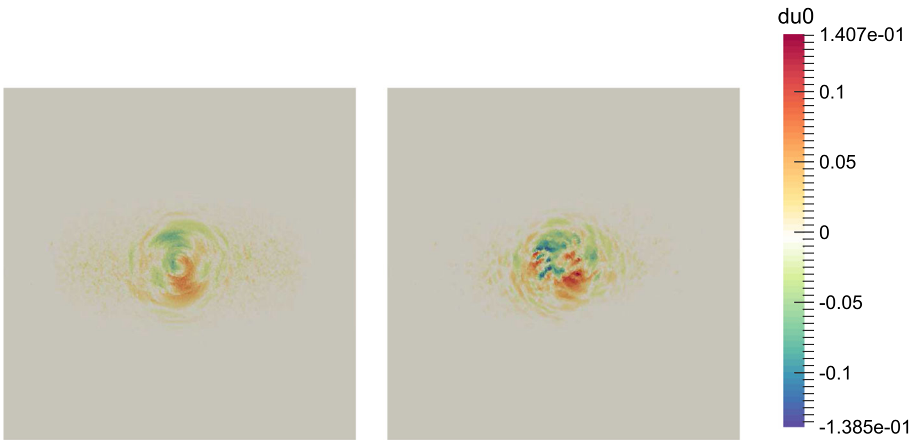

Fig. 10 Doswell frontogenesis: initial datum error $u(x, y, 0) - u^e(x, y)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

Table 1 Doswell Frontogenesis: L_2 and L_∞ norms, number of iterations and CPU time for the ∇J^{FOU} and ∇J^{SOU}

Scheme	L_2 (u^T)	L_∞ (u^T)	L_2 (u^0)	L_∞ (u^0)	nIters	CPU Time (s)
FOU	1.37127e+00	1.37078e−01	**1.36043e+00**	**8.64870e−02**	75	366.26
SOU	**1.23889e+00**	**1.34271e−01**	1.88749e+00	1.40718e−01	45	488.61

Table 1 condenses this information as well as the number of iterations and the CPU time for each proposed scheme. For the sake of clarity and for each comparison, the best results are highlighted in bold font.

Note that the number of iterations for the ∇J^{SOU} approach does not reach the maximum number of iterations set at the beginning of the inverse design process because at iteration

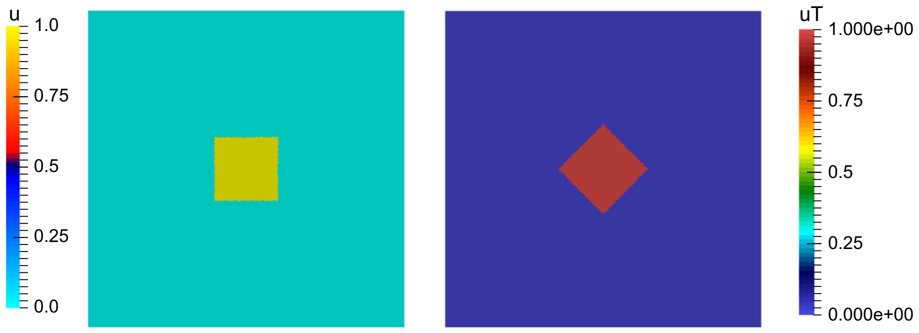

Fig. 11 Square rotation: exact initial condition (left) and target function (right) for the inverse problem

45, the functional is not able to diminish its value to the next iteration due to the deterioration of the initial condition by the high frequencies.

4.2 Test case 2: rotation of a square shape

The second test case is again a forward flow problem with analytical solution representing the advection of a square interface centered at (0.5,0.5)

$$u(x, y, 0) = \chi_{[0.4,0.6]\times[0.4,0.6]}(x, y) \tag{58}$$

in a computational domain $\Omega = [0, 1] \times [0, 1]$ under the influence of a constant vorticity velocity field given by:

$$\mathbf{v} = (y - \frac{1}{2}, \frac{1}{2} - x). \tag{59}$$

Therefore, after $T = \dfrac{3\pi}{4}$ the square shape rotates $\theta = 135^0$ and the analytical solution consists just of a rotation of the initial interface. In this case, boundary conditions are imposed on regions 1, 3, 5 and 7 (see Fig. 4) for the state equation and on regions 2, 4, 6 and 8 for the adjoint equation, respectively.

Thus, we formulate the inverse problem, taking as target the analytical solution at $t = \dfrac{3\pi}{4}$. The aim is then to recover the initial condition given by (58). Figure 11 shows the exact initial condition (left) and the target function (right) for the inverse problem.

The domain is discretized in 77398 unstructured triangles and the code is paralellised in 4 cores using OPENMP as in the previous test case. Both flow and adjoint equations use a CFL value of 0.5, which ensures their stability. For the gradient method (19), the first iteration is set $u^0(x, y, 0) = 0$, the maximum number of iterations is 75 and a constant gradient step $\varepsilon = 0.5$ is chosen.

Figures 12 and 13 show the numerical results achieved by the ∇J^{FOU} (left) and ∇J^{SOU} (right) for the target $u(x, y, T)$ and adjoint $\sigma(x, y, 0)$, respectively. The initial condition $u(x, y, 0)$ obtained by each approach is illustrated in Fig. 14 in plant view and in a 3D view for the ∇J^{FOU} (upper) and ∇J^{SOU} (lower).

The computation of norms L_2 and L_∞ as in (57) is straightforward for this test case. Results are displayed in Table 2, including the information about the number of iterations and the CPU time consumed by each approach.

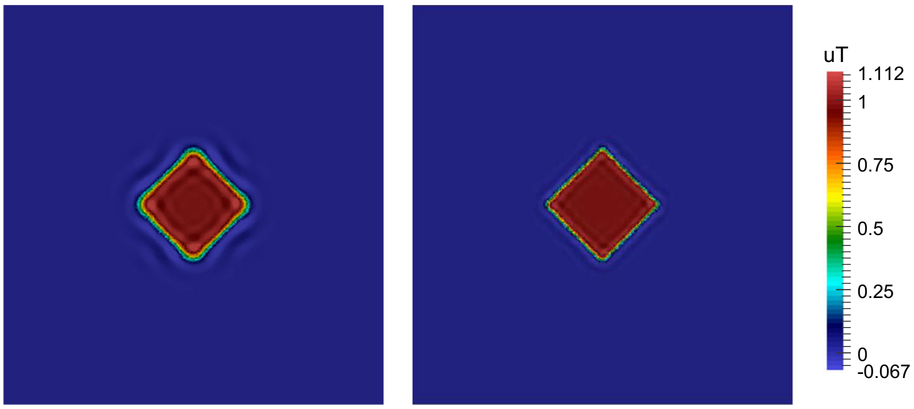

Fig. 12 Square rotation: numerical target $u(x, y, T)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

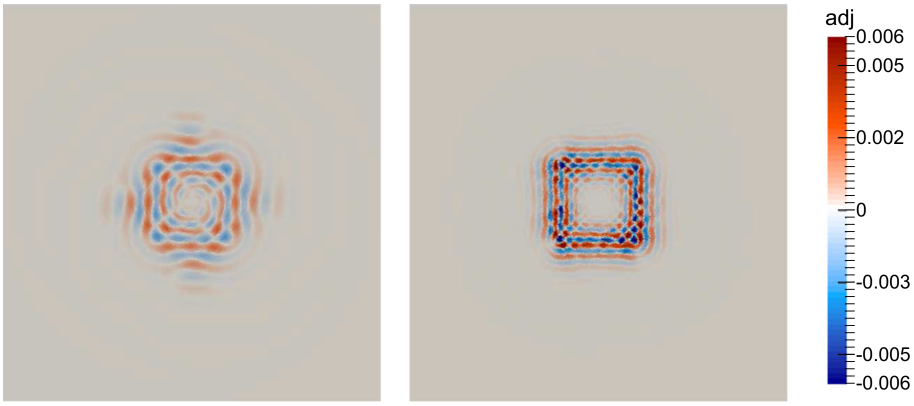

Fig. 13 Square rotation: numerical adjoint variable $\sigma(x, y, 0)$ achieved by ∇J^{FOU} (left) and ∇J^{SOU} (right)

Table 2 Square rotation: L_2 and L_∞ norms, number of iterations and CPU time for the ∇J^{FOU} and ∇J^{SOU}

Scheme	L_2 (u^T)	L_∞ (u^T)	L_2 (u^0)	L_∞ (u^0)	nIters	CPU time (s)
FOU	1.10586e+01	7.66632e−01	**1.15172e+01**	**8.24541e−01**	75	6169.88
SOU	**8.07958e+00**	**6.59056e−01**	1.23071e+01	8.66051e−01	75	11188.89

5 Discussion of the numerical results

First of all, it is important to highlight that both qualitative and quantitative results show the same behaviour: when regarding the error made with respect to the target function, the results obtained by the ∇J^{SOU} approach are more accurate than those achieved by the ∇J^{FOU} one. However, when computing the error with respect to the exact initial condition, the ∇J^{FOU} estimation is more convenient since the L_2 and L_∞ norms are reduced.

As expected, the ∇J^{FOU} recovers an initial condition with a smooth shape, which is mainly due to the numerical diffusion associated to the resolution of the adjoint equation with less order of accuracy. Therefore, in those cases whose exact initial condition is a

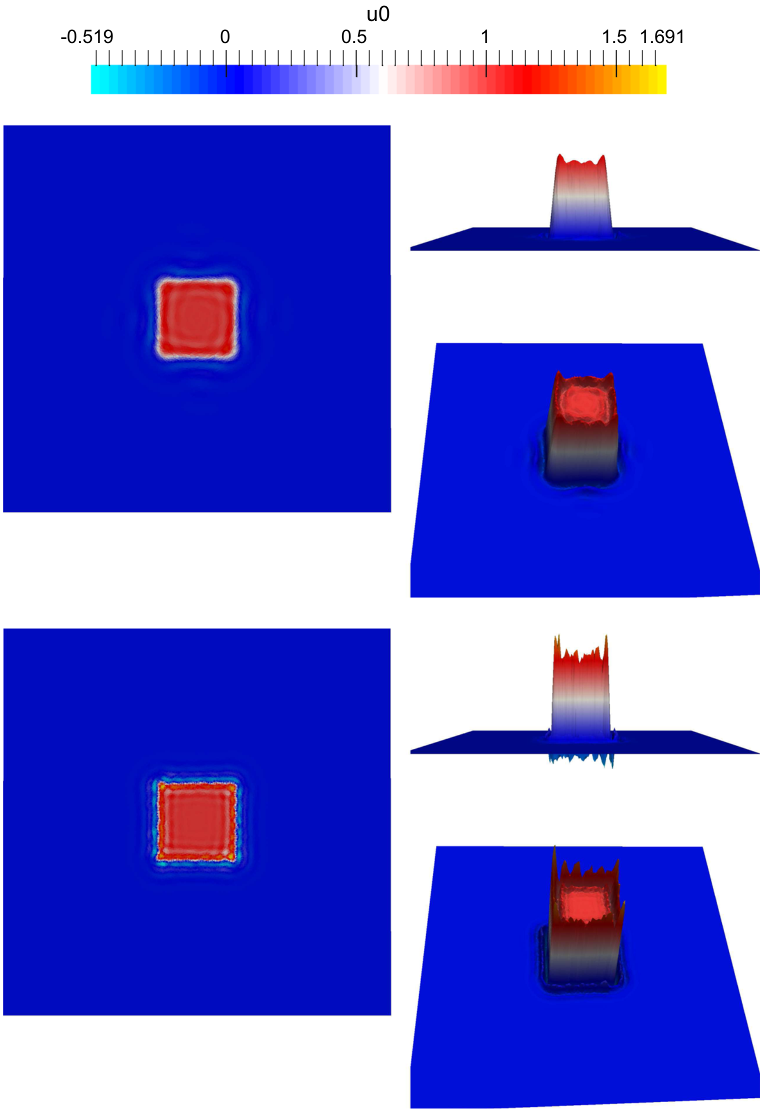

Fig. 14 Square rotation: numerical initial condition $u(x, y, 0)$ achieved by ∇J^{FOU} (upper) and ∇J^{SOU} (lower)

smooth function, this approach is highly recommended in contrast to the ∇J^{SOU} estimation. The difference between both approaches is not noticeable when dealing with discontinuous initial conditions, where both estimations provide an error in the same order of magnitude.

On the other hand, it is worth highlighting that the ∇J^{SOU} approach introduces high frequencies and spurious oscillations in the inverse design process. It is well documented that similar issues arise in other contexts, such as the boundary observation and control of wave-like equations (Zuazua 2005; Ervedoza and Zuazua 2013) or shape design problems for elliptic equations (Nochetto et al. 1996; Doğan et al. 2007) so that, algorithms, to converge, need to implement added smoothing or filtering techniques, to avoid high frequency spurious oscillations. However, the use of ∇J^{FOU} reduces considerably the number of discontinuities and oscillations, acting itself as a smoothing effect over the numerical approximation.

The CPU time, which constitutes a known bottleneck, is also examined. In gradient/adjoint methods the equations have to be solved forward and backward iteratively and it is important to employ light and quick forward and backward solvers providing a correct estimation of the solution. In this work, two solvers are examined: a First Order Upwind (FOU) scheme and a Second Order Upwind (SOU) one. First, as the best accuracy is required for the forward computations, the SOU scheme is adopted for the flow equations. However, both schemes provide similar results in terms of accuracy when dealing with the resolution of the adjoint equation. Therefore, the CPU time consumed by each model can be a decisive factor to choose the adequate solver. It is well known that the first order scheme requires less computations to achieve the numerical approximation of the solution. Hence the computational time is reduced when opting for this FOU rather than SOU not only with the same number of iterations (as in the square rotation) test case but also with a higher number of iterations (as in the frontogenesis test).

6 Analytical elucidation

In this section, the numerical results presented in the previous section are explained from an analytical point of view. To simplify the notation the linear scalar equation under consideration can be rewritten as follows:

$$u_t + Lu = 0 \quad u(0) = u_0 \qquad L = \nabla \cdot (\mathbf{v}u), \tag{60}$$

where $\mathbf{v}$ is a time-independent velocity field.

Given a target $u_d \in H$, H being a Hilbert space, we are interested in achieving $u_0 \in H$ such that the solution of the former evolution equation satisfies $u(T) = u_d$. As stated in the introduction, the focus is put on solving this problem via the minimization of the functional

$$J(u_0) = \frac{1}{2}|u(T) - u_d|^2 \tag{61}$$

by means of a gradient descent method where $\nabla J = \sigma(0)$ and σ is the adjoint variable:

$$u_0^{k+1} = u_0^k - \varepsilon\sigma^k(0). \tag{62}$$

The gradient $\sigma^k(0)$ is achieved by solving the adjoint equation:

$$-\sigma_t + L^*\sigma = 0 \quad \sigma(T) = u^k(T) - u_d, \tag{63}$$

where L^* is the adjoint operator.

Therefore, the minimization of (61) consists in a loop in which each iteration is described by means of the following flowchart:

$$u(0) \xrightarrow[\text{forward}]{(60)} u(T) - u_d \xrightarrow[\text{backward}]{(63)} \sigma(0). \tag{64}$$

When approximating both forward and adjoint solvers by the SOU we are simply implementing the same gradient descent strategy for the SOU approximation of the optimisation problem.

If, at the level of adjoint resolution, we replace the SOU by the FOU one, we rather implement a loop of the form

$$u_t + Lu = 0 \quad u(0) = u_0, \tag{65}$$
$$-\tilde{\sigma}_t + \tilde{L}^*\tilde{\sigma} = 0 \quad \tilde{\sigma}(T) = u(T) - u_d, \tag{66}$$

where $\tilde{L}^*$ is a perturbation of the original adjoint L^* obtained when replacing SOU by FOU.

This loop, a priori, does not correspond to the implementation of a gradient descent algorithm for the minimisation of a cost functional. The corresponding algorithm reads

$$u^{k+1} = u^k - \varepsilon\tilde{\sigma}^k(0) \tag{67}$$

and the modified loop becomes:

$$u(0) \xrightarrow[\text{forward}]{(65)} u(T) - u_d \xrightarrow[\text{backward}]{(66)} \tilde{\sigma}(0). \tag{68}$$

To better compare the two approaches, we do it in this abstract context in which notation is simpler, and the overall effect of modifying the solver for the adjoint equation can be easily identified.

Let us first consider the exact solution of (63)

$$\sigma(0) = e^{-TL_*}(u(T) - u_d) \tag{69}$$

and the exact solution of (66)

$$\tilde{\sigma}(0) = e^{-T\tilde{L}_*}(u(T) - u_d). \tag{70}$$

Thus, (70) can be written in terms of (69)

$$\tilde{\sigma}(0) = e^{-T\tilde{L}_*}(u(T) - u_d) = e^{-TL_*}B\ (u(T) - u_d), \tag{71}$$

where

$$B = e^{TL_*}e^{-T\tilde{L}_*}. \tag{72}$$

This indicates that, when solving the adjoint equation with the FOU scheme instead of with the SOU scheme, we are actually applying the gradient descent method to a modified cost functional of the form

$$J_B(u) = \frac{1}{2}\langle B(u(T) - u_d), u(T) - u_d\rangle \tag{73}$$

whose gradient is given by

$$\langle \nabla J_B(u), z\rangle \ = \ \langle B(u(T) - u_d), z(T)\rangle. \tag{74}$$

The change in the solver for the adjoint equation corresponds to filtering (through the operator B) the functional, weighting in a different manner the requirement $y(T) \sim y_d$. When passing from SOU to FOU, the high frequencies numerical components are penalised, while maintaining, essentially, the low ones. Therefore, the mixed algorithm combining the SOU for the forward resolution and the FOU for the adjoint one we employ in this paper corresponds to a projected adjoint-gradient method proposed as those employed in the numerical control of waves (noise reduction), Zuazua (2005); Ervedoza and Zuazua (2013), or in shape design optimization, Doğan et al. (2007).

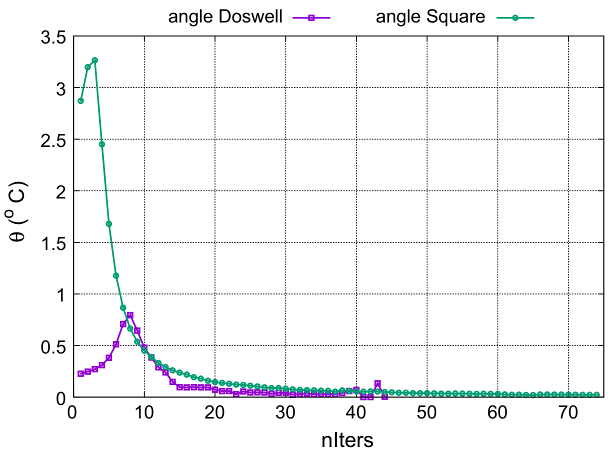

Fig. 15 Angle between the original and the modified adjoint

To compare both gradient descent directions, the following numerical experiment is considered: for each iteration k, let always consider the initial data u_0^k as the data provided by the SOU gradient computation:

$$u_0^k = u_0^{k-1} - \varepsilon \nabla^{k,\mathrm{SOU}}, \tag{75}$$

and let compute both the $\nabla^{k,\mathrm{SOU}}$ and $\nabla^{k,\mathrm{FOU}}$ approaches. In fact it makes no sense to compare the gradient directions in the iterates of the two methods since the adjoint resolution would introduce a modification that will be reflected in the next initial condition. Additionally, when moving to the next iterations, neither the forward resolution nor the backward one would start from the same initial datum, which would produce significant disparity in the gradient directions. Therefore, the following vectors can be defined:

$$\boldsymbol{\omega}^{k,\mathrm{FOU}} = \overrightarrow{(J^{k,\mathrm{SOU}}, J^{k+1,\mathrm{FOU}})} \quad \boldsymbol{\omega}^{k,\mathrm{SOU}} = \overrightarrow{(J^{k,\mathrm{SOU}}, J^{k+1,\mathrm{SOU}})} \tag{76}$$

that accounts for the variation in the functional (of each approach) in each iteration given the same initial datum. In particular, it is feasible to compute the angle θ between the $\boldsymbol{\omega}^{k,\mathrm{FOU}}$ and $\boldsymbol{\omega}^{k,\mathrm{SOU}}$ as follows:

$$\theta_k = \arccos \frac{\boldsymbol{\omega}^{k,\mathrm{FOU}} \cdot \boldsymbol{\omega}^{k,\mathrm{SOU}}}{|\boldsymbol{\omega}^{k,\mathrm{FOU}}|\,|\boldsymbol{\omega}^{k,\mathrm{FOU}}|}. \tag{77}$$

The results are plotted in Fig. 15 for both test cases. Note that only the first 45 iterations are compared for the test case 1.

As can be observed, the angles between both approaches are very close meaning that the use of ∇^{FOU} as represents a slight modification of the original ∇^{SOU} gradient estimation. A second conclusion should be mentioned: the angle decreases as the number of iterations grows. The inverse behaviour during the first seven iterations is due to the fact that the initial condition $u^0 = 0$ is very far from the exact solution hence both approaches make a relevant modification during these iterations, decreasing the functional in the same order of magnitude.

7 Remark: diffusive models

In this paper, we have analysed the purely hyperbolic model. Similar issues arise for convection-diffusion models, in the presence of an added diffusive term. Several comments are in order:

1. When adding the diffusion term, the model becomes parabolic and the boundary condition (the Dirichlet one in analogy with those considered in this article in the hyperbolic model) need to be imposed all along the boundary both for the state and the adjoint equation.
2. The gradient methodology described in this paper can be easily adapted to this case and, although the parabolic model becomes time-irreversible, the state equation is well-posed in the forward sense of time, while the adjoint equation is well-posed in the backward sense, thus making the whole methodology consistent.
3. When adding viscosity terms and thus making the numerical schemes more complex, it becomes even of greater utility to employ the continuous methodology when deriving the adjoint numerical scheme. And the discussion about “SOU versus FOU” to numerically approximate the adjoint system within the gradient iteration remains relevant.
4. The same conclusions of the hyperbolic tests apply on the parabolic context too.

8 Conclusions and perspectives

The convenience of using the same order of accuracy for the adjoint resolution for inverse problems is analysed in this work. In order to eliminate the uncertainties related to non-linear problems, the linear scalar equation in the 2D framework is selected. First, we opt for using the continuous approach for the inverse design problem since it provides a relevant versatility when using different numerical schemes in contrast to the discrete methodology.

The flow equation is solved by means of a second order accurate method. A gradient descent method is chosen to minimize the distance to a target function and the estimation of the gradient in this iterative method, which for the linear case coincides with the adjoint variable at t=0, is examined. Therefore, the numerical resolution of the adjoint equation is analysed by means of two numerical methods: a First Order Upwind (FOU) scheme and a Second Order Upwind (SOU) scheme. Some test cases with exact solution are used to determine the adequate solver and the error in terms of L_2 and L_∞ norms not only with respect to the target function, but also with respect to the exact solution are computed. The CPU time consumed by the iterative algorithm by the FOU and the SOU approaches is also displayed.

Although the ∇^{SOU} estimation is able to achieve less error with respect to the target function, the ∇^{FOU} approach provides a better (or at least the same) error when regarding the norms of the exact solution. In particular, it can be concluded that the first order order estimation is the best choice when dealing with smooth functions since the second order approach introduces some high frequencies and undesirable oscillations. On the other hand, if considering sharp interfaces or discontinuities not only in the target function but also in the exact solution, the differences between both approaches are reduced and the error is in the same order of magnitude. Since the computational time required by the first order estimation is reduced in comparison to the second order approach, ∇^{FOU} is preferred, at least for these kind of unsteady problems. A deep study would be necessary for non-linear problems.

Finally, an analytical explanation for this fact is also provided, where it can be concluded that a modified gradient-adjoint method is indeed used when solving with the ∇^{FOU} approach, that allows to filter naturally the high frequencies and to modify the descent direction in the gradient method to achieve the same or better results in terms of accuracy in comparison with the ∇^{SOU} estimation. A remark regarding diffusive problems has been also included.

To conclude, the main perspectives are related to the extrapolation of these ideas to more complex non-linear fluxes where shock waves are present. Furthermore, the behaviour of the first order adjoint resolution for higher order solver for the flow equation can be also considered as a future work.

Acknowledgements This work was done while the first author was a postdoctoral fellow of the team of the Advanced Grant NUMERIWAVES/FP7-246775 of the European Research Council. The second author was partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement NO: 694126-DyCon), the Grant MTM2017-92996-C2-1-R COSNET of MINECO (Spain), the ELKARTEK project KK-2018/00083 ROAD2DC of the Basque Government, the Air Force Office of Scientific Research under Award NO: FA9550-18-1-0242, the Marie Curie Training Action "ConFlex" SEP-210412102 and the ICON project of the French ANR.

Appendix A: Forward simulation with SOU and FOU schemes

The aim of this Appendix is to numerically demonstrate that the use of the SOU scheme rather than a FOU scheme is essential for the forward simulation (flow equation). Therefore both FOU and SOU schemes are compared for test cases 1 and 2. Figures 16 and 17 shows the numerical results for the frontogenesis test case at t = 4s and for the rotating square test case at $t = \frac{3\pi}{4}$s, respectively. On the left side the FOU scheme is used while on the right side the SOU scheme. As can be observed, the numerical diffusion associated to the FOU scheme is clearly highlighted in comparison with the SOU scheme hence the forward simulation must be solved with a high order scheme, in this case the SOU scheme.

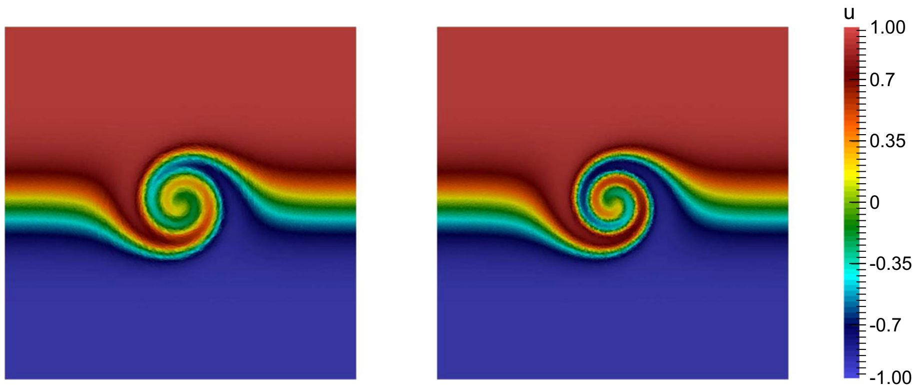

Fig. 16 Doswell frontogenesis: numerical approximation achieved by the FOU (left) and SOU (right) schemes

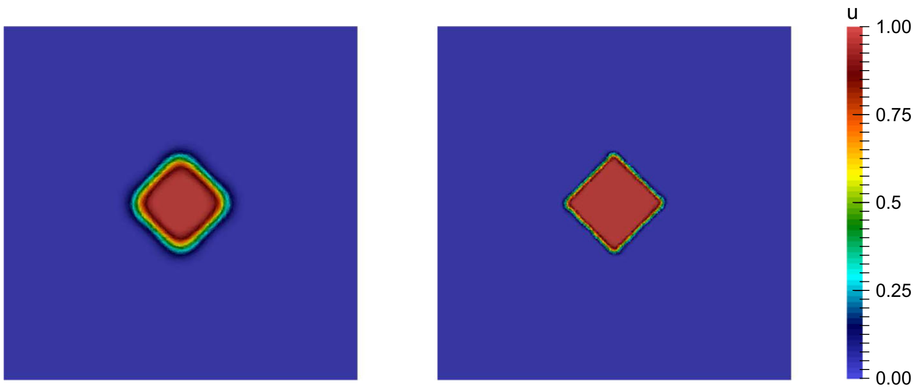

Fig. 17 Square rotation: numerical approximation achieved by the FOU (left) and SOU (right) schemes

References

Carpentieri G, Koren B, van Tooren MJL (2007) Adjoint-based aerodynamic shape optimization on unstructured meshes. J Comput Phys 224:267–287

Castro C, Lozano C, Palacios F, Zuazua E (2007) Systematic continuous adjoint approach to viscous aerodynamic design on unstructured grids. AIAA J 45–9:2125–2139

Castro C, Palacios F, Zuazua E (2008) An alternating descent method for the optimal control of the inviscid Burgers equation in the presence of shocks. Math Models Methods Appl Sci 18:369–416

Ciarlet PG (1982) Introduction à l'analyse numérique matricielle et à l'optimisation, Collection Mathématiques appliquées pour la maîtrise

Courant R, Friedrichs K, Lewy H (1928) Über die partiellen Differenzengleichungen der mathematischen Physik. Math Ann (in German) 100(1):32–74

Doğan G, Morin P, Nochetto RH, Verani M (2007) Discrete gradient flows for shape optimization and applications. Comput Methods Appl Mech Eng 196:3898–3914

Doswell CA (1984) A kinematic analysis of frontogenesis associated with a nondivergent vortex. J Atmos Sci 41:1242–1248

Ervedoza S, Zuazua E (2013) On the numerical approximation of exact controls for waves. Springer Briefs in Mathematics, XVII, ISBN 978-1-4614-5808-1

Giles MB, Pierce NA (2000) An introduction to the adjoint approach to design. Flow Turbul Combus 65:393–415

Glowinski R (1992) Ensuring well-posedness by analogy; stokes problem and boundary control for the wave equation. J Comput Phys 103–2:189–221

Godlewski E, Raviart PA (1996) Numerical approximation of hyperbolic systems of conservation laws. Applied Mathematical Sciences. Springer, Berlin

Herty M, Kurganov A, Kurochkin D (2015) Numerical method for optimal control problems governed by nonlinear hyperbolic systems of PDE's. Commun Math Sci 13(1):15–48

Huang H, Ascher U (2014) Faster gradient descent and the efficient recovery of images. Vietnam J Math 42:115–131

Hubbard ME (1999) Multidimensional slope limiters for MUSCL-type finite volume schemes on unstructured grids. J Comput Phys 155:54–74

Jameson A (1988) Aerodynamic design via control theory. J Sci Comput 3–3:233–260

Li S, Petzold L (2004) Adjoint sensitivity analysis for time-dependent partial differential equations with adaptative mesh refinement. J Comput Phys 198:310–325

Morales-Hernández M, García-Navarro P, Burguete J, Brufau P (2013) A conservative strategy to couple 1D and 2D models for shallow water flow simulation. Comput Fluids 81:26–44

Nadarajah SK, Jameson A (2000) A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. In: Proceedings of the 38th AIAA aerospace sciences meeting and exhibit, AIAA 2000-0667

Nocedal J, Wright SJ (1999) Numerical optimization. Springer series in operations research, Springer, Berlin

Nochetto RH, Paolini M, Verdi C (1996) A dynamic mesh algorithm for curvature dependent evolving interfaces. J Comput Phys 123:296–310
Peter JEV, Dwight RP (2010) Numerical sensitivity analysis for aerodynamic optimization: a survey of approaches. Comput Fluids 39:373–391
Power PW, Piggott MD, Fang F, Gorman GJ, Pain CC, Marshall DP, Goddard AJH, Navon IM (2006) Adjoint goal-based error norms for adaptive mesh ocean modelling. Ocean Model 15(1–2):3–38. https://doi.org/10.1016/j.ocemod.2006.05.001
Shewchuk JR (2002) Delaunay refinement algorithms for triangular mesh generation. Comput Geom Theory Appl 22:21–74
Sweby PK (1984) High resolution schemes using flux-limiters for hyperbolic conservation laws. SIAM J Numer Anal 21(5):995–1011
Toro EF (2009) Riemann solvers and numerical methods for fluid dynamics: a practical introduction. Springer, Berlin
Ulbrich S (2001) Optimal control of nonlinear hyperbolic conservation laws with source terms, Habilitation Thesis, Zentrum Mathematik, Technische UniversitYat Munchen, Germany
van Leer B (1979) Towards the ultimate conservative difference scheme, V. A second order sequel to Godunov's method. J Comput Phys 32:101–136
Zuazua E (2002) Controllability of partial differential equations and its semi-discrete approximations. Discrete Contin Dyn Syst 8(2):469–513
Zuazua E (2005) Propagation, observation, and control of waves approximated by finite difference methods. SIAM Rev 47(2):197–243
Zuazua E (2007) Controllability and observability of partial differential equations: some results and open problems. In: Handbook of differential equations: evolutionary equations, vol 3, chap 7. Elsevier, pp 527–621