Available online at www.eccomasproceedia.org
Eccomas Proceedia EUROGEN (2023) 223-238

ECCOMAS
Proceedia

EUROGEN 2023
15th ECCOMAS Thematic Conference on
Evolutionary and Deterministic Methods for Design, Optimization and Control
N. Gauger, K. Giannakoglou, M. Papadrakakis, J. Periaux (eds.)
Chania, Crete Greece, 1–3 June 2023

THE *THINK DISCRETE–DO CONTINUOUS* ADJOINT IN AERODYNAMIC SHAPE OPTIMIZATION

Kyriakos Giannakoglou, Varvara Asouti, Evangelos Papoutsis–Kiachagias, Nikolaos Galanos, Marina Kontou and Xenofon Trompoukis

Parallel CFD & Optimization Unit (PCOpt), School of Mechanical Engineering,
National Technical University of Athens (NTUA), Athens, Greece
e-mails: {kgianna,vasouti,vpapout,ngalanos,mkontou}@ntua.mail.gr, xeftro@gmail.com

Abstract. *This paper presents activities carried out in the Parallel CFD & Optimization Unit of NTUA, regarding the formulation, programming and assessment of an adjoint method that combines the advantages of continuous and discrete adjoint, for use in gradient-based optimization, in problems governed by PDEs. The paper presents, for the first time in the literature, the concept of the Think-Discrete-Do-Continuous (TDDC) adjoint. The idea is as simple as that: the hand-differentiated discrete adjoint is used to guide the development of consistent discretization schemes for the continuous adjoint PDEs. By doing so, the new continuous adjoint (TDDC adjoint) computes sensitivity derivatives with the same accuracy as discrete adjoint, without though an excessive memory footprint; in contrast, the TDDC adjoint retains the useful insight into the adjoint equations, the adjoint boundary conditions and the expression of sensitivity derivatives. The development is made for two widely used classes of CFD codes: (a) pressure-based solvers for incompressible fluid flows, such as the open-source OpenFOAM© code and (b) hyperbolic solvers for compressible fluid flows, such as the GPU-enabled flow solver PUMA developed by the group of authors. Cases related to duct flows or flows around airfoils and (transonic) wings are used to demonstrate the accuracy with which the TDDC adjoint computes the gradient; the implementation of the TDDC adjoint in shape optimization in the same cases is shown too.*

Keywords: Continuous Adjoint, Discretization Schemes, Aerodynamic Shape Optimisation

doi: 10.7712/140123.10201.18965

1 INTRODUCTION

The CFD-based optimization has clear cost benefits compared to the extensive use of experiments, and has already been extended to multi-disciplinary problems in which one of the disciplines is fluid mechanics. The present work is exclusively related to gradient-based methods (GBMs) that start from a given set of design variables (i.e. a given design) and improve it step-by-step by computing and using the gradient (sensitivity derivatives, SDs) of the objective function J with respect to (w.r.t.) the design variables $b_n, n = 1, ..., N$. Supported by an efficient method to compute gradients, a GBM becomes fast, though occasionally trapped into local optima. The cost of GBMs is determined by the cost of computing the gradient. The adjoint method [1, 2] is the only one with a cost that is independent of the number of design variables, and is thus the only method that can handle problems with many design variables.

To set-up the adjoint method, J is augmented by the sum of the residuals of the flow (primal) equations multiplied by the adjoint variables. In continuous adjoint [2, 3], J is augmented using the flow equations in the form of PDEs, and the resulting adjoint equations are PDEs to be discretized and numerically solved. Finding appropriate discretization schemes for the adjoint PDEs is a challenge. On the other hand, in discrete adjoint [4, 5, 6], the discrete expression of J is augmented by the discretized residuals of the primal equations; its differentiation directly leads to the adjoint equations in discrete form.

Developers of discrete adjoint are proud of computing gradients that are fully consistent with the primal (CFD) solver and of making the adjoint solver inherit its convergence characteristics from the primal one. Strong arguments of those developing continuous adjoint methods are the ease of implementation, the physical insight into the adjoint terms/equations, the lower computational cost, and, the low memory footprint of the adjoint code. Research performed in the last years by the group of authors, under the code name "*Think Discrete – Do Continuous*" (TDDC) adjoint, ended up with a continuous adjoint method supported by discretization schemes which are inspired by discrete adjoint. Practically, the new TDDC adjoint bridges the gap between the two adjoint approaches, by combining the best of both worlds. This is achieved by developing discretization schemes for the differential operators (convection, diffusion, gradients, etc) of the continuous adjoint PDEs, their boundary conditions and the SDs expressions, to replicate what discrete adjoint does, without its weaknesses though. The TDDC adjoint allows the understanding of the discretization of the adjoint PDEs, which is not the case in discrete adjoint. It can easily be programmed by re-using a large part of the flow solver, such as that dealing with higher-order terms or the communication of fluxes between adjacent subdomains processed by different processors. Furthermore, the TDDC adjoint will have the memory footprint of continuous adjoint, which is much lower than that of discrete adjoint, [7, 8]. Regarding accuracy and consistency with the primal solver, an agreement of an adequate number of the first significant digits between TDDC adjoint and FDs is expected, as with discrete adjoint.

Consistent discretization schemes for the TDDC adjoint are presented using two different ways of formulating and solving the flow equations: (a) pressure-based methods, as in OpenFOAM (the continuous adjoint solver of which has been made publicly available by the group of authors, [9], and this is the first software to be enhanced with the TDDC adjoint, herein) and (b) a time-marching hyperbolic-type solver for compressible fluid flows, by employing the TDDC adjoint to the in–house GPU-accelerated code PUMA developed by the group of authors (the "standard" adjoint of this code can be found in [10]).

A literature survey reveals that works in the special field of our enquiry are seriously restricted. We should though report [11] which presents a mathematically more rigorous approach, by deriving by hand the discrete adjoint fluxes and reverse-engineering a discretization scheme that, when applied to one or more of the continuous adjoint terms, reproduces the former. This work mainly focused on the proper discretization of the so-called adjoint transpose convection term and covers the first-order part of the convection term and the orthogonal part of the diffusion fluxes. In addition, an attempt to build the adjoint to the segregated SIMPLE algorithm was made, without though focusing on some delicate parts of the discretization process, such as the Rhie–Chow interpolation (see below, [12]) or the second-order fluxes of the convection and diffusion terms.

2 TDDC ADJOINT FOR PRESSURE–BASED SOLVERS

2.1 Primal equations & discretization

To showcase the key idea of the TDDC adjoint, applied to pressure–based CFD solvers for incompressible flows, a quasi–1D flow problem can be used. If $S(x)$ is the cross–section distribution along the unit length of the channel, which is controlled by the design variables b_n, the continuity and momentum equations are written in the form

$$R^p = -\frac{d\,(vS)}{dx} = 0 \tag{1}$$

$$R^v = \frac{d\,(vSv)}{dx} - \frac{d}{dx}\left(\nu S\frac{dv}{dx}\right) + S\frac{dp}{dx} + \lambda\sqrt{S}v^2 = 0 \tag{2}$$

where v is the velocity, p the static pressure divided by the fluid's density, and ν is the kinematic viscosity of the fluid. The last term in eq. 2 contributes to the total pressure (p_t) drop inside the duct due to the effect of shear forces; λ is a Darcy coefficient.

Let us assume that eqs. 1, 2 are discretized using a vertex–centered, finite volume scheme, with a collocated arrangement of the flow variables, after having discretized the length of the channel with N_p equidistant nodes, with constant spacing equal to Δx. Even if the ultimate goal of this section is to derive discretization schemes for the adjoint code for multi–dimensional flows as developed in OpenFOAM (which uses cell–centered finite volumes), the present (vertex–centered) development is both useful and easy to understand. Practically, it does not make any difference, since the exact same development can be made on the dual grid.

A second–order upwind discretization scheme, is written (at node i) as

$$R^p_i = -v_{i+\frac{1}{2}}\overline{S}_{i+\frac{1}{2}} + v_{i-\frac{1}{2}}\overline{S}_{i-\frac{1}{2}} = 0 \tag{3}$$

$$\begin{aligned} R^v_i = {} & v_{i+\frac{1}{2}}\overline{S}_{i+\frac{1}{2}}v^{UPW}_{i+\frac{1}{2}} - v_{i-\frac{1}{2}}\overline{S}_{i-\frac{1}{2}}v^{UPW}_{i-\frac{1}{2}} - \nu\overline{S}_{i+\frac{1}{2}}\frac{v_{i+1}-v_i}{\Delta x} + \nu\overline{S}_{i-\frac{1}{2}}\frac{v_i - v_{i-1}}{\Delta x} \\ & + S_i\frac{p_{i+1}-p_{i-1}}{2} + \lambda\sqrt{S_i}v_i^2\Delta x = 0 \end{aligned} \tag{4}$$

At midnodes, $v^{UPW}_{i+\frac{1}{2}} = v_i + \frac{v_{i+1}-v_{i-1}}{4}$, $\overline{\phi}_{i+\frac{1}{2}} = \frac{\phi_i + \phi_{i+1}}{2}$, where ϕ is any flow or geometric quantity, and convecting velocities are computed via the so–called Rhie–Chow interpolation, [12], as follows

$$v_{i+\frac{1}{2}} = \overline{v}_{i+\frac{1}{2}} - \overline{D}_{i+\frac{1}{2}}\overline{S}_{i+\frac{1}{2}}\left[\frac{dp}{dx} - \overline{\frac{dp}{dx}}\right]_{i+\frac{1}{2}} \tag{5}$$

The term into brackets corresponds to the third–derivative of pressure

$$\left[\frac{dp}{dx} - \overline{\frac{dp}{dx}}\right]_{i+\frac{1}{2}} = \frac{-p_{i+2} + 3p_{i+1} - 3p_i + p_{i-1}}{4\Delta x} \cong -\frac{1}{4\Delta x}\frac{d^3p}{dx^3}\Big|_{i+\frac{1}{2}}\Delta x^3 \tag{6}$$

and helps overcoming odd–even decoupling in the pressure field. Hereafter, $\left.\frac{d^3\phi}{dx^3}\right|_{i+\frac{1}{2}}$ stands for the finite difference stencil involving the surrounding nodal values of ϕ, as in eq. 6. Also, $D_i = \Delta x / A_{Pi}$, where A_{Pi} is the coefficient of v_i in the discretized (at node i) eq. 4, [13]. Using eq. 5, eq. 3 takes the form of a Poisson–type pressure equation, the solution of which contributes to a divergence–free velocity field. For the sake of simplicity, during the development of the (discrete and) TDDC adjoint as presented in this section, it is assumed that D_i does not depend on the flow variables; of course, this is not the case of the software (in OpenFOAM; for multi–dimensional flows) used in the Results section. To keep the presentation short, we refrain from presenting the discretization of boundary conditions, and focus on consistent discretization schemes for the continuous adjoint equations, for internal nodes only.

2.2 Continuous and discrete adjoint

The derivation of the continuous adjoint method for an objective function $J = \int j dx$ (written in the form of a field integral along the duct length) starts by the definition of the Lagrangian $J_{aug} = J + \int uR^v dx + \int qR^p dx$, where u is the adjoint velocity and q the adjoint pressure. Using integration by parts, and since $\frac{d}{dx}$ and $\frac{\delta}{\delta b_n}$ permute, the derivatives of J_{aug} w.r.t. b_n take the form

$$\begin{aligned}\frac{\delta J_{aug}}{\delta b_n} = & \int R^u \frac{\delta v}{\delta b_n} dx + \int R^q \frac{\delta p}{\delta b_n} dx + \left[BC^u \frac{\delta v}{\delta b_n} \right]_{x=0}^{x=1} + \left[BC^q \frac{\delta p}{\delta b_n} \right]_{x=0}^{x=1} \\ & + \int \left(v\frac{dq}{dx} - v^2 \frac{du}{dx} + u\frac{dp}{dx} + \nu \frac{dv}{dx}\frac{du}{dx} + \lambda \frac{v^2}{2\sqrt{S}} u + \frac{\partial j}{\partial S} \right) \frac{\delta S}{\delta b_n} dx \end{aligned} \tag{7}$$

To avoid computing $\frac{\delta v}{\delta b_n}$ and $\frac{\delta p}{\delta b_n}$, their field multipliers in eq. 7 are set to zero, giving rise to the field adjoint equations

$$R^q = -\frac{d(uS)}{dx} + \frac{\partial j}{\partial p} = 0 \tag{8}$$

$$R^u = -2vS\frac{du}{dx} + S\frac{dq}{dx} - \frac{d}{dx}\left(\nu S \frac{du}{dx} \right) + 2\lambda\sqrt{S}vu + \frac{\partial j}{\partial v} = 0 \tag{9}$$

Setting the multipliers of $\frac{\delta v}{\delta b_n}$ and $\frac{\delta p}{\delta b_n}$ to zero (if not otherwise eliminated) at the boundary nodes gives rise to the adjoint boundary conditions; this is related with the terms denoted by BC^u and BC^q. Adjoint boundary conditions will be omitted here, as we did for the primal boundary conditions too. After satisfying the field adjoint equations and their boundary conditions, the last integral in eq. 7 stands for the SDs expression.

To derive the discrete adjoint to the same problem, J is now augmented by the sum of the discretized residuals (eqs. 3, 4) multiplied by their adjoint variables, giving rise to the Lagrangian $J_{aug} = J + u_i R_i^v + q_i R_i^p$, where J is in discrete form and repeated indices imply summation over all nodes. Its differentiation w.r.t. b_n yields

$$\frac{\delta J_{aug}}{\delta b_n} = \underbrace{\left(u_j \frac{\partial R_j^v}{\partial v_i} + q_j \frac{\partial R_j^p}{\partial v_i} + \frac{\partial J}{\partial v_i} \right)}_{R_i^u} \frac{\delta v_i}{\delta b_n} + \underbrace{\left(u_j \frac{\partial R_j^v}{\partial p_i} + q_j \frac{\partial R_j^p}{\partial p_i} + \frac{\partial J}{\partial p_i} \right)}_{R_i^q} \frac{\delta p_i}{\delta b_n} + \underbrace{\left(u_j \frac{\partial R_j^v}{\partial S_i} + q_j \frac{\partial R_j^p}{\partial S_i} + \frac{\partial J}{\partial S_i} \right)}_{SD} \frac{\delta S_i}{\delta b_n} \tag{10}$$

where $R_i^u = 0$, $R_i^q = 0$ are the discrete adjoint momentum and continuity equations at node i and the last term on the r.h.s. is the expression for the SDs.

A hand–differentiation of eqs. 3, 4 w.r.t. v_i and p_i leads to the following terms in the discrete adjoint continuity and momentum equations (written for node i)

$$R_i^q = \tfrac{1}{2}\left(u_{i-1}S_{i-1} - u_{i+1}S_{i+1}\right) + \frac{\Delta x^3}{4}\frac{d\Phi^3}{dx^3}\Big|_i + \frac{\partial J}{\partial p_i} = 0 \tag{11}$$

$$\begin{aligned} R_i^u =& \tfrac{1}{8}\overline{S}_{i-\frac{1}{2}}\left(v_{i-2}u_i - v_{i-2}u_{i-1} + 5v_{i-1}u_{i-1} - 5v_{i-1}u_i + 2v_iu_{i-1} - 2v_iu_i\right) \\ &+\tfrac{1}{8}\overline{S}_{i+\frac{1}{2}}\left(v_{i-1}u_{i+1} - v_{i-1}u_i + v_iu_i - v_iu_{i+1} + 5v_{i+1}u_i - 5v_{i+1}u_{i+1}\right) \\ &+\tfrac{1}{8}\overline{S}_{i+\frac{3}{2}}\left(v_{i+1}u_{i+2} - v_{i+1}u_{i+1} + v_{i+2}u_{i+2} - v_{i+2}u_{i+1}\right) \\ &-\xi_{i-\frac{1}{2}} - 4\xi_{i+\frac{1}{2}} + \xi_{i+\frac{3}{2}} - \nu\overline{S}_{i+\frac{1}{2}}\tfrac{1}{\Delta x}\left(u_{i+1}-u_i\right) + \nu\overline{S}_{i-\frac{1}{2}}\tfrac{1}{\Delta x}\left(u_i - u_{i-1}\right) \\ &+\tfrac{1}{2}\overline{S}_{i+\frac{1}{2}}\left(q_{i+1}+q_i\right) - \tfrac{1}{2}\overline{S}_{i-\frac{1}{2}}\left(q_i + q_{i-1}\right) + 2\lambda\sqrt{S_i}v_iu_i\Delta x + \frac{\partial J}{\partial v_i} = 0 \end{aligned} \tag{12}$$

where Φ, ξ are defined only at midnodes as $\Phi_{i+\frac{1}{2}} = \overline{S}_{i+\frac{1}{2}}\overline{D}_{i+\frac{1}{2}}\left(v^{UPW}_{i+\frac{1}{2}}\frac{u_{i+1}-u_i}{\Delta x} - \frac{q_{i+1}-q_i}{\Delta x}\right)$ and $\xi_{i+\frac{1}{2}} = \frac{1}{16}\Delta x^3\overline{S}^2_{i+\frac{1}{2}}\overline{D}_{i+\frac{1}{2}}\frac{d^3p}{dx^3}\Big|_{i+\frac{1}{2}}\frac{u_{i+1}-u_i}{\Delta x}$. The term $\frac{\Delta x^3}{4}\frac{d^3\Phi}{dx^3}\Big|_i$ in eq. 11 is a shifted finite difference stencil which, compared to eq. 6, is now defined at nodes and is computed using the Φ values at the four surrounding midnodes. Eq. 11 gets contributions from both the primal continuity and momentum equations; it is expressed as a difference of the product of adjacent u and S nodal values without involving values at midnodes (compared to eq. 3). Finally, the discrete adjoint SDs are

$$\begin{aligned} \frac{\delta J}{\delta b_n} = & \Bigg(-\tfrac{1}{2}v_{i+\frac{1}{2}}v^{UPW}_{i+\frac{1}{2}}\left(u_{i+1}-u_i\right) - \tfrac{1}{2}v_{i-\frac{1}{2}}v^{UPW}_{i-\frac{1}{2}}\left(u_i - u_{i-1}\right) \\ &+\tfrac{\nu}{2\Delta x}\left[\left(v_{i+1}-v_i\right)\left(u_{i+1}-u_i\right) + \left(v_i - v_{i-1}\right)\left(u_i - u_{i-1}\right)\right] + \tfrac{1}{2}u_i\left(p_{i+1}-p_{i-1}\right) \\ &+\tfrac{1}{2}v_{i+\frac{1}{2}}\left(q_{i+1}-q_i\right) + \tfrac{1}{2}v_{i-\frac{1}{2}}\left(q_i - q_{i-1}\right) + \frac{\lambda}{2\sqrt{S_i}}v_i^2u_i\Delta x + \frac{\partial J}{\partial S_i}\Bigg)\frac{\delta S_i}{\delta b_n} \end{aligned} \tag{13}$$

where midnodal primal velocities $v_{i+\frac{1}{2}}$ are given by eq. 5.

2.3 The TDDC adjoint

Having the expressions for the discrete and continuous adjoint equations available, the TDDC adjoint introduces discretization schemes which, when used to discretize the continuous adjoint equations, lead to the exact same SDs as discrete adjoint. Let us indicatively focus on the convection term $-2vS\frac{du}{dx}\Big|_i$ of eq. 9. Its discretization (integrated over the corresponding finite volume) should be given by the formula

$$\begin{aligned} -2\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} vS\frac{du}{dx}dx = & \tfrac{1}{4}\frac{v_{i+1}+v_{i+2}}{2}\overline{S}_{i+\frac{3}{2}}\left(u_{i+2}-u_{i+1}\right) - \tfrac{6}{4}\frac{v_{i-1}+8v_i+3v_{i+1}}{12}\overline{S}_{i+\frac{1}{2}}\left(u_{i+1}-u_i\right) \\ & -\tfrac{3}{4}\frac{-v_{i-2}+5v_{i-1}+2v_i}{6}\overline{S}_{i-\frac{1}{2}}\left(u_i - u_{i-1}\right) + \tfrac{\Delta x^3}{16}\left(-W_{i-\frac{1}{2}} - 4W_{i+\frac{1}{2}} + W_{i+\frac{3}{2}}\right) \end{aligned} \tag{14}$$

where $W_{i+\frac{1}{2}} = \overline{S}^2_{i+\frac{1}{2}}\overline{D}_{i+\frac{1}{2}}\frac{d^3p}{dx^3}\Big|_{i+\frac{1}{2}}\frac{u_{i+1}-u_i}{\Delta x}$.

One can easily understand the scheme presented in eq. 14. It consists of three contributions coming from three consecutive intervals ($[i-1, i]$, $[i, i+1]$, $[i+1, i+2]$; rather than just

$\left[i-\frac{1}{2}, i+\frac{1}{2}\right]$, as in the primal problem). Within each interval, $\frac{du}{dx}$ is approximated through central differences, and S by averaging nodal values. Thanks to the TDDC adjoint, one may see the way v is defined at these three intervals, see corresponding terms in eq. 14. This reflects the effect of the upwind scheme used in the primal equation. The last term in eq. 14 reflects the way eq. 5 computes midnode velocities in the primal problem.

The idea of the TDDC adjoint covers, also, the discretization of the expression for the SDs derived in continuous adjoint. Guided by eq. 13, the most important terms in the last integral of eq. 7 must be discretized as

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} v\frac{dq}{dx}dx = \tfrac{1}{2}\left(v_{i+\frac{1}{2}}\frac{q_{i+1}-q_i}{\Delta x} + v_{i-\frac{1}{2}}\frac{q_i - q_{i-1}}{\Delta x}\right)\Delta x \tag{15}$$

$$-\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} v^2\frac{du}{dx}dx = -\tfrac{1}{2}\left(v_{i+\frac{1}{2}}v^{UPW}_{i+\frac{1}{2}}\frac{u_{i+1}-u_i}{\Delta x} + v_{i-\frac{1}{2}}v^{UPW}_{i-\frac{1}{2}}\frac{u_i-u_{i-1}}{\Delta x}\right) \tag{16}$$

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u\frac{dp}{dx}dx = u_i\frac{p_{i+1}-p_{i-1}}{2\Delta x}\Delta x \tag{17}$$

and

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \nu\frac{dv}{dx}\frac{du}{dx}dx = \frac{\nu}{2}\left[\frac{v_{i+1}-v_i}{\Delta x}\frac{u_{i+1}-u_i}{\Delta x} + \frac{v_i-v_{i-1}}{\Delta x}\frac{u_i-u_{i-1}}{\Delta x}\right]\Delta x \tag{18}$$

3 TDDC ADJOINT FOR COMPRESSIBLE HYPERBOLIC SOLVERS

3.1 Primal equations & discretization

To present the TDDC adjoint for the compressible fluid model, the multi-dimensional Euler equations, are written as

$$R_n^{MF} = \frac{\partial f_{nk}}{\partial x_k} = 0 \ \ , \ \ n=1,..,4(,5), \ \ k=1,2(,3) \tag{19}$$

and solved by the GPU–accelerated PUMA code of the PCOpt/NTUA.

Here, $f_{nk} = [\rho v_k \ \ \rho v_k v_{n-1} + p\delta_{n-1,k} \ \ \rho v_k h_t]$ are the inviscid fluxes. Eq. 19 is solved for the conservative flow variables $\mathbf{U} = [\rho \ \ \rho v_k \ \ \rho E]^T$, where ρ, v_k, E, h_t and δ_{km} are the fluid's density, the velocity components, the total energy per unit mass, the total enthalpy and the Kronecker symbol, respectively. The primitive flow variables' array is defined as $\mathbf{V} = [\rho \ \ v_k \ \ p]^T$, with p being the static pressure. A vertex-centered finite volume formulation on unstructured grids is used; an integration of eq. 19 over a finite volume Ω_P defined around an internal or boundary node P, by applying the Green–Gauss theorem, results to the balance of numerical fluxes Φ crossing the boundaries of Ω_P, $\sum\limits_Q \Phi_n^{PQ} + \sum\limits_f \Phi_n^f = 0$, where $\sum\limits_Q$ denotes summation over all adjacent nodes Q connected with P by a grid edge, and $\sum\limits_f$ summation over all boundary faces $f \in \Omega_P$, fig. 1. Eq. 19 is numerically integrated in pseudo-time by adding a pseudo-time derivative $\partial U_n/\partial\tau$ to it.

Fluxes Φ_n crossing the interface of Ω_P and Ω_Q are discretized based on the Roe's upwind scheme, [14],

$$\Phi_n^{PQ} = \tfrac{1}{2}\left(A^P_{nmk}U^P_m + A^Q_{nmk}U^Q_m\right)\mathsf{n}_k^{PQ} - \tfrac{1}{2}\left|\tilde{A}^{LR}_{nmk}\mathsf{n}_k^{PQ}\right|\left(U^R_m - U^L_m\right) \tag{20}$$

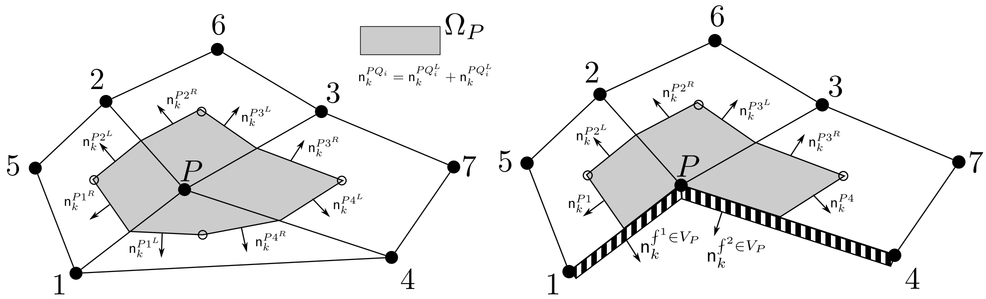


Figure 1: Finite volumes formed around an internal node P (left) and a boundary node P (right). Nodes $(1,2,3,4)$ which are connected by a grid edge with P consist the set of Q neighbours, while nodes $(1,2,3,4,5,6,7)$ which are or are not connected by a grid edge with P consist the set of $\mathcal{N}(P)$ neighbours. The Q nodes form a subset of $\mathcal{N}(P)$. The magnitude of the normal vectors ($\mathbf{n}$) on the finite volume boundaries is equal to the area (length) of this boundary (i.e. it is dimensional normal vector).

where $A_{nmk} = \frac{\partial f_{nk}}{\partial U_m}$ stands for the flux Jacobian and $\tilde{A}_{nmk}$ for the Jacobian computed using Roe-averaged quantities. Superscripts L and R indicate the left (towards P) and right (towards Q) states of the interface and n_k^{PQ} is the dimensional face normal pointing towards Q. U_m^L, U_m^R are computed based on the primitive V_m^L, V_m^R quantities, extrapolated using the spatial gradients computed at P, Q , as follows

$$V_m^L = V_m^P + \frac{1}{2}\mathsf{t}_\ell^{PQ} \left.\frac{\partial V_m}{\partial x_\ell}\right|^P, \quad V_m^R = V_m^Q - \frac{1}{2}\mathsf{t}_\ell^{PQ} \left.\frac{\partial V_m}{\partial x_\ell}\right|^Q, \quad \mathsf{t}^{PQ} = \overrightarrow{PQ} \tag{21}$$

$$\left.\frac{\partial V_m}{\partial x_\ell}\right|^P = D_\ell^P V_m^P + \sum_{\Lambda \in \mathcal{N}(P)} Z_\ell^{P\Lambda} V_m^\Lambda, \quad \left.\frac{\partial V_m}{\partial x_\ell}\right|^Q = D_\ell^Q V_m^Q + \sum_{K \in \mathcal{N}(Q)} Z_\ell^{QK} V_m^K \tag{22}$$

where coefficients D_ℓ and Z_ℓ are based on geometrical data.

3.2 Continuous and discrete adjoint

The development of the continuous adjoint method is herein performed according to the Field Integral (FI) adjoint method (term introduced in [15], there for incompressible flows though). The development makes use of the expression $\frac{\delta}{\delta b_i}\left(\frac{\partial(\cdot)}{\partial x_k}\right) = \frac{\partial}{\partial x_k}\left(\frac{\delta(\cdot)}{\delta b_i}\right) - \frac{\partial(\cdot)}{\partial x_\ell}\frac{\partial}{\partial x_k}\left(\frac{\delta x_\ell}{\delta b_i}\right)$, [15], which allows the derivatives of the Lagrangian J_{aug} to be written as

$$\begin{aligned}\frac{\delta J_{aug}}{\delta b_i} &= \frac{\delta J}{\delta b_i} + \int_\Omega \Psi_n \frac{\delta R_n}{\delta b_i} \mathrm{d}\Omega \\ &= \underbrace{\frac{\delta J}{\delta b_i} + \int_{\partial\Omega} \Psi_n \hat{\mathsf{n}}_k \frac{\delta f_{nk}}{\delta b_i} \mathrm{d}S}_{\rightarrow \text{ABC}} \underbrace{- \int_\Omega A_{nmk} \frac{\partial \Psi_n}{\partial x_k} \frac{\delta U_m}{\delta b_i} \mathrm{d}\Omega}_{\rightarrow \text{FAE}} \underbrace{- \int_\Omega \Psi_n \frac{\partial f_{nk}}{\partial x_\ell} \frac{\partial}{\partial x_k}\left(\frac{\delta x_\ell}{\delta b_i}\right) \mathrm{d}\Omega}_{\rightarrow \text{SDs}}\end{aligned} \tag{23}$$

where Ψ_n are the mean flow adjoint variables. The integral marked as FAE gives rise to the Field Adjoint Equations which read

$$-A_{mnk} \frac{\partial \Psi_m}{\partial x_k} = 0 \tag{24}$$

Integrating eq. 24 over Ω_P, in the standard finite volume notation, the inviscid term can be written as a balance of fluxes Φ

$$-\int_{\Omega_P} A_{mnk}\frac{\partial \Psi_m}{\partial x_k} d\Omega \simeq -\sum_Q \Phi_n^{\text{adj},PQ} - \sum_f \Phi_n^{\text{adj},f} \tag{25}$$

where $\hat{\mathsf{n}}_k$ is the outward unit normal vector. The TDDC adjoint relies on the definition of consistent discretization schemes for the adjoint flux $\Phi_n^{\text{adj},PQ}$, inspired by the discrete adjoint to the same problem. As in the incompressible case before, we will refrain from focusing on the Adjoint Boundary Conditions associated with the surface integrals marked as ABC. So, the next paragraphs are exclusively concerned with the internal nodes.

The development of the discrete adjoint starts by the corresponding J_{aug} that is now based on the discretized primal equations which, if differentiated, becomes:

$$\frac{\delta J_{aug}}{\delta b_i} = \frac{\delta J}{\delta b_i} + \sum_P \Psi_n^P \sum_Q \frac{\delta \Phi_n^{PQ}}{\delta b_i} + \sum_P \Psi_n^P \sum_f \frac{\delta \Phi_n^f}{\delta b_i} \tag{26}$$

Using eq. 20, the second term on the r.h.s. of eq. 26, after swapping nodes P and Q or P and $\Lambda \in \mathcal{N}(P)$ (wherever the contribution to the adjoint flux for node Q or $\Lambda \in \mathcal{N}(P)$ appears), takes the form

$$\begin{aligned}
&\sum_P \Psi_n^P \sum_Q \frac{\delta \Phi_n^{PQ}}{\delta b_i} = \\
&\underbrace{\sum_P\sum_Q \left(\Psi_n^P - \Psi_n^Q\right)\left(\mathcal{T}_{n\lambda}^P + \mathcal{T}_{n\lambda}^{LR}\right)\frac{\delta U_\lambda^P}{\delta b_i} + \sum_P \mathcal{D}_{\ell r}^P D_r^P \left.\frac{\partial V_\ell}{\partial U_\lambda}\right|^P \frac{\delta U_\lambda^P}{\delta b_i} + \sum_P \sum_{\Lambda\in\mathcal{N}(P)} \mathcal{D}_{\ell r}^\Lambda Z_r^{\Lambda P} \left.\frac{\partial V_\ell}{\partial U_\lambda}\right|^P \frac{\delta U_\lambda^P}{\delta b_i}}_{\text{FAE}} \\
&\underbrace{+ \frac{1}{4}\sum_P\sum_Q \left(\Psi_n^P - \Psi_n^Q\right)\left[\left|\tilde{A}_{nm}^{LR}\right| \left.\frac{\partial U_m}{\partial V_\ell}\right|^L - \left(U_m^R - U_m^L\right)\frac{\partial \left|\tilde{A}_{nm}^{LR}\right|}{\partial V_\ell^L}\right]\frac{\partial V_\ell^P}{\partial x_r}\frac{\delta \mathsf{t}_r^{PQ}}{\delta b_i} + \sum_P \mathcal{D}_{\ell r}^P V_\ell^P \frac{\delta D_r^P}{\delta b_i}}_{SDs} \\
&\underbrace{+ \frac{1}{2}\sum_P\sum_Q \left(\Psi_n^P - \Psi_n^Q\right)\left(A_{nmk}^P U_m^P + U_m^L \frac{\partial \left|\tilde{A}_{nm}^{LR}\right|}{\partial \mathsf{n}_k^{PQ}}\right)\frac{\delta \mathsf{n}_k^{PQ}}{\delta b_i} + \sum_P \sum_{\Lambda\in\mathcal{N}(P)} \mathcal{D}_{\ell r}^\Lambda V_\ell^P \frac{\delta Z_r^{\Lambda P}}{\delta b_i}}_{SDs}
\end{aligned} \tag{27}$$

and

$$\mathcal{T}_{n\lambda}^P = \tfrac{1}{2}A_{n\lambda}^P, \;\; A_{n\lambda}^P = A_{n\lambda k}^P \mathsf{n}_k^{PQ}, \;\; \mathcal{T}_{n\lambda}^{LR} = \tfrac{1}{2}\left[\left|\tilde{A}_{nm}^{LR}\right| \left.\frac{\partial U_m}{\partial V_\ell}\right|^L - \left(U_m^R - U_m^L\right)\frac{\partial \left|\tilde{A}_{nm}^{LR}\right|}{\partial V_\ell^L}\right]\left.\frac{\partial V_\ell}{\partial U_\lambda}\right|^P$$

$$\mathcal{D}_{\ell r}^P = \tfrac{1}{4}\sum_Q \left(\Psi_n^P - \Psi_n^Q\right)\left(\left|\tilde{A}_{nm}^{LR}\right| \frac{\partial U_m^L}{\partial V_\ell^L} - \left(U_m^R - U_m^L\right)\frac{\partial \left|\tilde{A}_{nm}^{LR}\right|}{\partial V_\ell^L}\right)\mathsf{t}_r^{PQ}$$

So, the discrete FAE for Ω_P reads

$$\sum_P \sum_Q \left[-\frac{1}{2}\left(\Psi_n^P + \Psi_n^Q\right) A_{n\lambda}^P + \frac{1}{2}\left(\Psi_n^P - \Psi_n^Q\right) \left[\left|\tilde{A}_{nm}^{LR}\right| \frac{\partial U_m}{\partial V_\ell}\bigg|^L - \left(U_m^R - U_m^L\right) \frac{\partial \left|\tilde{A}_{nm}^{LR}\right|}{\partial V_\ell^L} \right] \frac{\partial V_\ell}{\partial U_\lambda}\bigg|^P \right.$$
$$\left. + \mathcal{D}_{\ell r}^P D_r^P \frac{\partial V_\ell}{\partial U_\lambda}\bigg|^P + \mathcal{D}_{\ell r}^{\mathcal{N}(P)} Z_r^{\mathcal{N}(P)P} \frac{\partial V_\ell}{\partial U_\lambda}\bigg|^P \right] = 0 \tag{28}$$

Note that, according to eq. 27, the first term on the l.h.s. of eq. 28 could have been written as $\frac{1}{2}\left(\Psi_n^P - \Psi_n^Q\right) A_{n\lambda}^{PQ}$; however, this appears as $-\frac{1}{2}\left(\Psi_n^P + \Psi_n^Q\right) A_{n\lambda}^{PQ} + \Psi_n^P A_{n\lambda}^P$ and the last term can be neglected as $\sum\limits_P \sum\limits_Q \Psi_n^P A_{n\lambda}^P = \sum\limits_P \Psi_n^P A_{n\lambda k}^P \sum\limits_Q \mathsf{n}_k^{PQ}$ and $\sum\limits_Q \mathsf{n}_k^{PQ} = 0$ for all internal nodes.

3.3 The TDDC adjoint

Based on eq. 28, the adjoint fluxes appearing in the continuous adjoint equations, eqs. 24 and 25 must be discretized as follows:

$$\Phi_n^{\text{adj},PQ} = -\frac{1}{2} A_{mn}^P \left(\Psi_m^P + \Psi_m^Q\right) - \frac{1}{2}\left[\left(\left|\tilde{\mathcal{A}}_{m\ell}^{LR}\right| \Psi_m\right)^{R,\text{adj}} - \left(\left|\tilde{\mathcal{A}}_{m\ell}^{LR}\right| \Psi_m\right)^{L,\text{adj}} \right] \frac{\partial V_\ell}{\partial U_n}\bigg|^P \tag{29}$$

Regarding the dissipation term (let M be any node connected with the nodes of $\mathcal{N}(P)$ by an edge), the adjoint left and right states (denoted as "L,adj" and "R,adj") associated with the edge PQ for a quantity ϕ are derived as

$$\phi^{L,\text{adj}} = \phi^P + \frac{1}{2} \frac{\partial (\mathsf{t}_r \phi)}{\partial x_r}\bigg|^{P,\text{adj}}, \quad \frac{\partial (\mathsf{t}_r \phi)}{\partial x_r}\bigg|^{P,\text{adj}} = \mathsf{t}_r^{PQ} D_r^P \phi^P + \sum_{\Lambda \in \mathcal{N}(P)} Z_r^{\Lambda P} \sum_M \mathsf{t}_r^{\Lambda M} \phi^\Lambda$$
$$\phi^{R,\text{adj}} = \phi^Q + \frac{1}{2} \frac{\partial (\mathsf{t}_r \phi)}{\partial x_r}\bigg|^{Q,\text{adj}}, \quad \frac{\partial (\mathsf{t}_r \phi)}{\partial x_r}\bigg|^{Q,\text{adj}} = \mathsf{t}_r^{PQ} D_r^P \phi^Q + \sum_{\Lambda \in \mathcal{N}(P)} Z_r^{\Lambda P} \sum_M \mathsf{t}_r^{\Lambda M} \phi^M$$

and $\left|\tilde{\mathcal{A}}_{m\ell}^{LR}\right|$ is expressed as $\left|\tilde{\mathcal{A}}_{m\ell}^{LR}\right| = \left|\tilde{A}_{m\lambda}^{LR}\right| \frac{\partial U_\lambda^L}{\partial V_\ell^L} - \left(U_\lambda^R - U_\lambda^L\right) \frac{\partial \left|\tilde{A}_{m\lambda}^{LR}\right|}{\partial V_\ell^L}$. Downgrading eq. 29 to first-order accuracy (now $\left|\tilde{\mathcal{A}}_{m\ell}^{PQ}\right| = \left|\tilde{A}_{m\ell}^{PQ}\right| - \left(U_\lambda^Q - U_\lambda^P\right) \frac{\partial \left|\tilde{A}_{m\lambda}^{PQ}\right|}{\partial U_\ell^P}$) one obtains

$$\Phi_n^{\text{adj},PQ} = -\frac{1}{2} A_{mn}^P \left(\Psi_m^P + \Psi_m^Q\right) - \frac{1}{2}\left|\tilde{\mathcal{A}}_{mn}^{PQ}\right| \left(\Psi_m^Q - \Psi_m^P\right) \tag{30}$$

Some comments on the similarity (also differences) of the expressions for the primal and adjoint fluxes, i.e. eqs. 20 and 28, 30, are due. Eq. 30 presents a typical, downwind, non-conservative scheme, consistent with the upwind Roe's scheme of the primal discretization; the only difference is the use of $\left|\tilde{\mathcal{A}}_{m\ell}^{PQ}\right|$ instead of the standard absolute Jacobian. Eq. 29 presents a non-conservative, consistent to the primal, second-order accurate discretization scheme. One should notice that, now, the L and R states for an adjoint variable are defined differently than for the primal ones and are denoted by "L,adj" and "R,adj". Moreover, the use of $\left|\tilde{\mathcal{A}}_{m\ell}^{LR}\right|$ as the absolute Jacobian, instead of $\left|\tilde{A}_{m\ell}^{LR}\right|$ is introduced; by doing so, information regarding the derivatives of the absolute Jacobian w.r.t. the flow variables is added to the discretization scheme.

4 TEST CASES FOR THE ASSESSMENT OF THE TDDC ADJOINT

The TDDC adjoint is assessed in four problems: *Case I & II* using the pressure-based flow solver OpenFOAM©(programmed following the material presented in section 2.3, after adapting formulas written there to a cell–centered discretization) and *Case III & IV*, using the compressible hyperbolic solver PUMA. The four cases are:

Case I-OpenFOAM©: Verification of the TDDC adjoint on a 2D S-bend duct with a laminar flow at $Re=2533$. The shape of the duct is parameterized by a 7×5 volumetric NUBRS control lattice (fig. 3a). The objective function is the volume–averaged total pressure losses. The effect of grid size to the accuracy of the SDs computed by the TDDC adjoint is investigated on three progressively refined grids (fig. 2a).

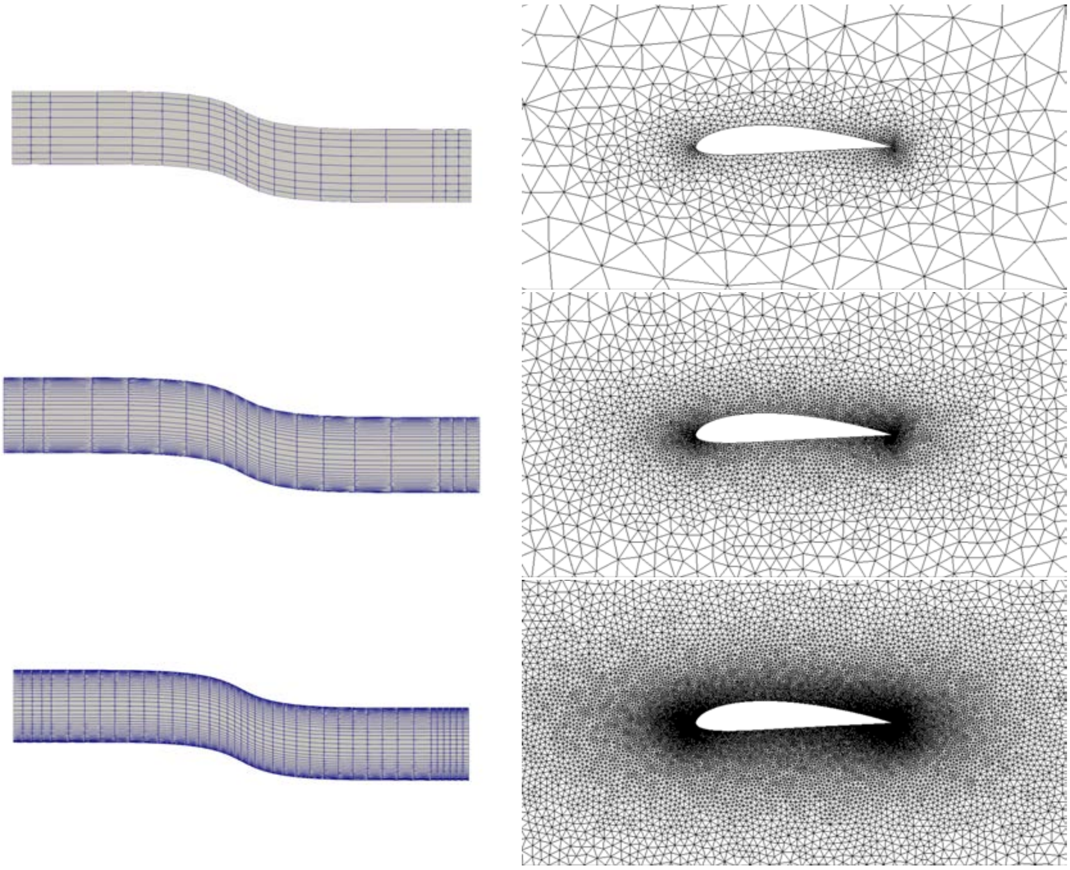
Case II-OpenFOAM©: Shape optimization of an isolated airfoil, operating at an angle of attack equal to $\alpha_\infty=1.5°$ and $Re=33391$ (also, a laminar case). Its shape is parameterized by a 6×4 volumetric NURBS control lattice, fig. 3b. The optimization aims at minimizing the drag coefficient (C_D) while retaining the lift coefficient (C_L) of the starting (reference) airfoil ($\pm1‰$); though this is, in fact, an equality constraint, it is imposed as a double–sided inequality constraint. An additional inequality constraint, requiring that the airfoil area should not drop below 85% of the starting one, is imposed.

Case III-PUMA: Shape optimization of the NACA4415 isolated airfoil for min. C_D with the constraints that C_L remains close to its reference value (within $\pm1‰$) and the airfoil volume does not drop below 85% of the initial one. The flow is inviscid with free-stream Mach number $M_\infty=0.70$ and $\alpha_\infty=2.0°$. Three grids, fig. 2b, are used to investigate the grid density effect on the computed SDs. The airfoil is controlled by a 10×7 volumetric NURBS lattice, fig. 3c; the 16 control points in red can be displaced in the normal-to-the-chord direction.

Case IV-PUMA: Shape optimization of a transonic isolated wing; the geometry of [16] is used as the reference wing. The flow conditions are: $M_\infty=0.8395$, $\alpha_{\infty,pitch}=3.06°$ and $\alpha_{\infty,yaw}=0°$. The wing shape and the grid (~ 73000 nodes) are parameterized using a $8\times7\times5$ volumetric NURBS control grid, fig. 3d; 18 control points are allowed to move in the chordwise and the normal-to-the-planform direction, resulting to 36 design variables, in total. The optimization aims at max. C_L, with the constraint that C_D should not exceed that of the reference wing.

4.1 Assessment of the TDDC adjoint for pressure–based solvers (OpenFOAM)

In *Case I*, SDs of the volume–averaged total pressure drop between the inlet and outlet of the duct are computed based on the new TDDC adjoint method on three progressively finer grids (see fig. 2a), and are compared against Finite–Differences (FDs), which is considered as the method computing reference sensitivities, and those computed using OpenFOAM©with the "standard" discretization schemes for the adjoint equations. The latter, to be refered to as Standard Continuous Adjoint (Standard CA), involves a second-order downwind discretization of the convection terms in the adjoint PDEs, second-order discretization schemes for Laplacian operators using Gauss' theorem and the computation of spatial gradients based on Gauss' theorem and linear interpolations to compute the adjoint quantities at grid faces. In this work, Standard CA stands for the publicly available continuous adjoint solver of OpenFOAM©(programmed by the group of authors), [17]. The latter agrees well with FDs only if an adequately fine grid is used. The TDDC adjoint computes SDs which are in perfect agreement with FDs irrespective of the grid size (an accuracy of 6 up to 9 significant digits is obtained, even on the coarsest grid); this verifies that the proposed TDDC adjoint achieves its goal.



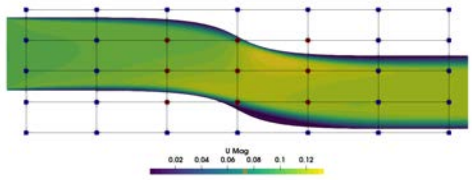
(a) *Case I* (b) *Case III*

Figure 2: Progressively refined grids used to assess the accuracy of the computed SDs for *Cases I, III*. For *Case I*, they consist of 200 (top), 800 (center), and 2000 (bottom) cells. The first (coarsest) one is quite inappropriate for a laminar flow prediction. For *Case III*, grids of ~ 1500 (top), ~ 6500 (center), and ~ 27000 (bottom) nodes are used.

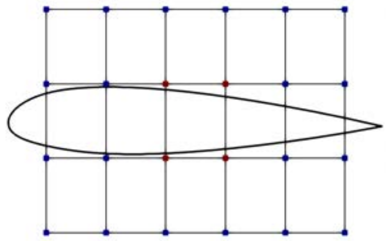
In *Case II*, the pressure–based TDDC adjoint solver, verified in the previous case, is used to optimize the shape of an isolated airfoil. The SDs of C_D and C_L for the reference airfoil, as computed by the TDDC adjoint, are compared against FDs, fig. 5. An accuracy of enough significant digits is obtained, respectively, verifying the accuracy of the proposed discretization schemes. The optimization is performed using the sequential quadratic programming (SQP) method using interior point methods to solve the QP problem, [18]. The convergence of the objective function during the optimization is shown in fig. 6; a reduction in C_D by $\sim 5\%$ is obtained, while the C_L coefficient remains within the imposed bounds throughout the optimization. The area in the optimized airfoil is reduced by $\sim 5.1\%$ compared to the reference one, so the corresponding constraint is also met. The reference and optimized airfoils, as well as the distributions of the static pressure coefficient as a function of the chord's percentage, are presented in fig. 7.

4.2 Assessment of the TDDC adjoint for compressible hyperbolic solvers (PUMA)

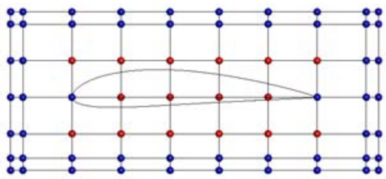
In *Case III*, the shape optimization of the NACA4415 isolated airfoil is carried out using the compressible fluid flow solver PUMA for min. C_D with the constraint that C_L remains close to the reference value (within $\pm 1‰$). The SDs of C_D and C_L computed based on the Standard



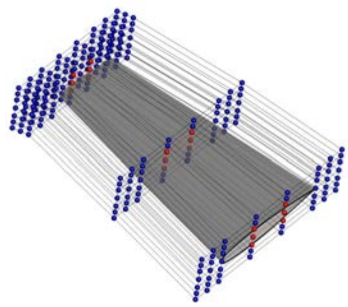
(a) *Case I*



(b) *Case II*



(c) *Case III*



(d) *Case IV*

Figure 3: Parameterization of shapes and grids. Control points in blue are fixed while red ones are allowed to move.

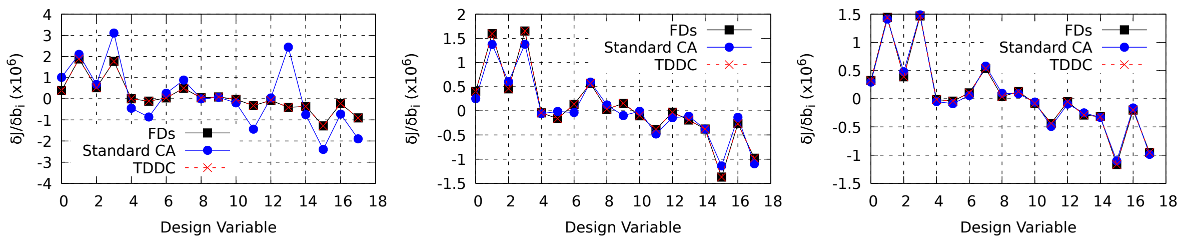


Figure 4: *Case I*: SDs of total pressure losses computed at the three grids shown in fig 2a (in the same order from left to right) by FDs (black), the Standard CA method (blue) and the TDDC adjoint (red) to the pressure–based flow solver of OpenFOAM.

CA (can be found in [10]) and the TDDC adjoint are compared with FDs in fig. 8 for the three different grids, coarse to fine (left to right). The SDs based on the TDDC adjoint have, at least, a six-decimal digit accuracy, regardless the grid quality; this verifies the accuracy of the proposed discretization schemes. On the other hand, small discrepancies exist when the Standard CA is used; these are more intense for the C_D value and, as expected, decrease as grid becomes finer.

The convergence history of the optimization is presented in fig. 9 for the medium-sized grid; an $\sim 80\%$ reduction is achieved, maintaining the C_L value close to the reference one. The volume of the optimized airfoil is reduced by 12.3%. The Mach number fields around the reference and optimized airfoils are presented in fig. 10; the shock strength is reduced and so does C_D.

In *Case IV*, the same solver is used for the shape optimization of a transonic isolated wing for

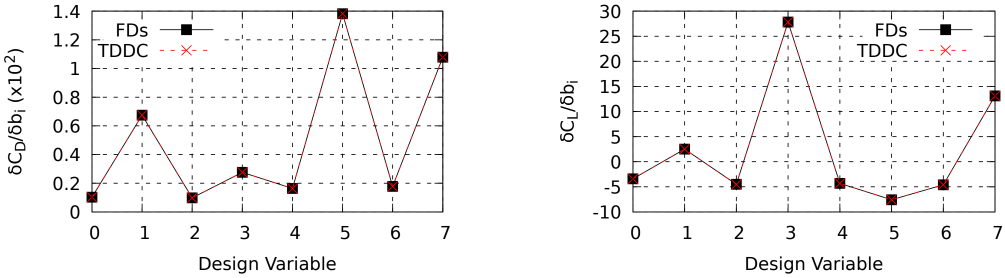


Figure 5: *Case II*: SDs of C_D (left) and C_L (right) computed using FDs (black) and the TDDC adjoint (red).

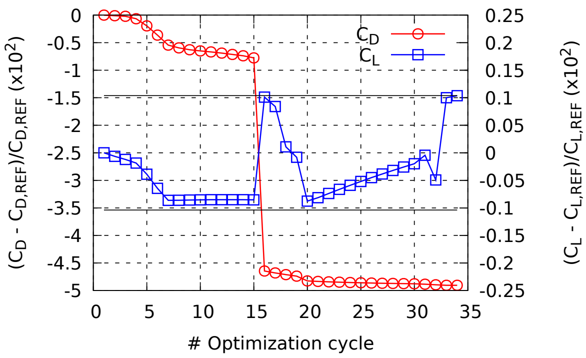


Figure 6: *Case II*: Evolution of C_D (objective function) and C_L (constraint) during the optimization.

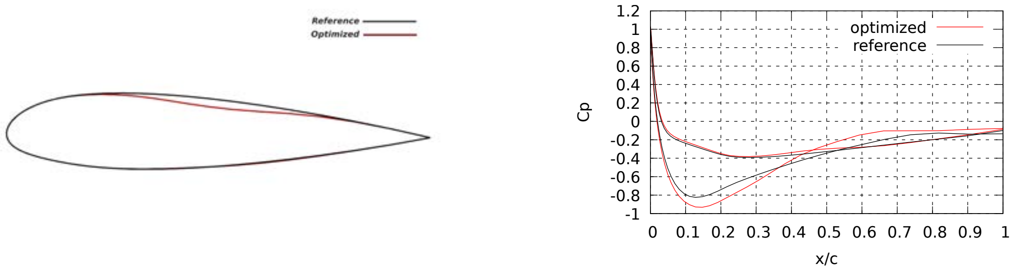


Figure 7: *Case II*: Shape of the reference and optimized airfoil (left). Distributions of the static pressure coefficient (C_P) computed on the surface of the reference and optimized airfoil plotted as a function of the chord percentage (right).

max. C_L, with the constraint that C_D is less or equal to that of the reference wing. Comparison of the SDs for C_D and C_L computed based on the TDDC adjoint and FDs, as well as the optimization convergence history are presented in fig. 11. An increase by $\sim 20\%$ in C_L is obtained for the optimized wing with the drag being close to that of the reference wing. The Mach number fields for the reference and optimized wings are presented in fig. 12.

5 CONCLUSIONS

A new continuous adjoint scheme (to be referred to as the *Think-Discrete-Do-Continuous* or *TDDC adjoint*) for use in gradient-based optimization was presented for two solvers: a pressure-

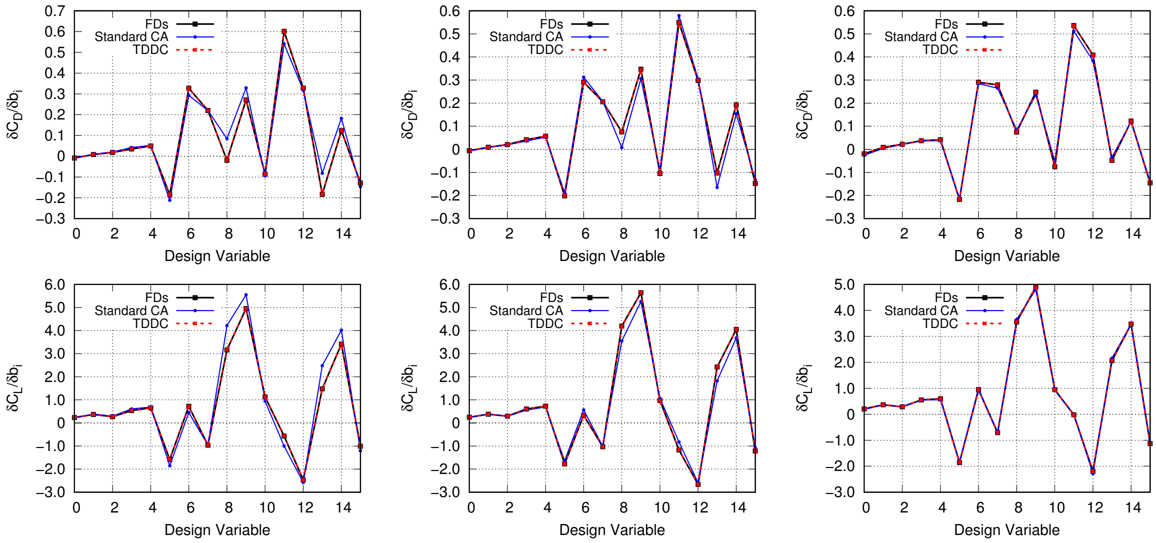


Figure 8: *Case III*: SDs of C_D (top) and C_L (right) computed at the three grids shown in fig. 2b using FDs (black), Standard CA (blue) and TDDC adjoint (red).

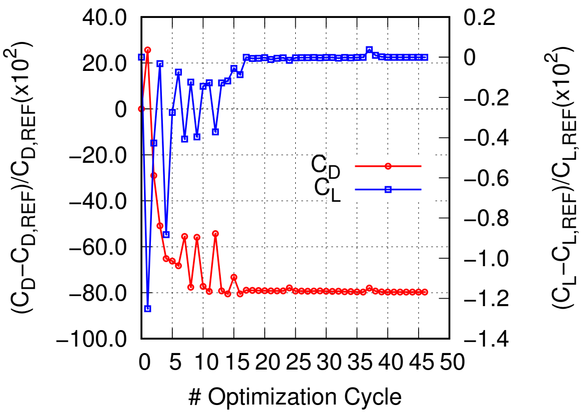


Figure 9: *Case III*: Evolution of the optimization and constraint functions.

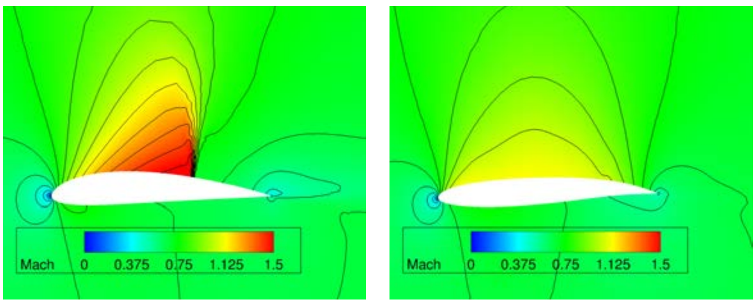


Figure 10: *Case III*: Mach number iso-areas for the reference (left) and optimized airfoils (right).

based one for incompressible flows (OpenFOAM©) and a time-marching hyperbolic-type solver for compressible flows (the in-house PUMA code). Both are based on the finite-volume method: the former is cell-centered, the latter is vertex-centered. The development of the TDDC adjoint starts by the development of the discrete adjoint (hand-differentiated) and, then, builds dis-

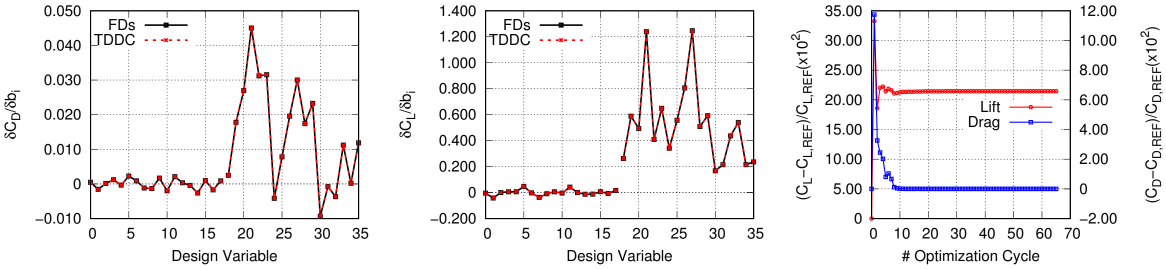


Figure 11: *Case IV*: SDs of C_D (left), C_L (center) computed using FDs (black) and the TDDC adjoint (red). Optimization convergence history (right).

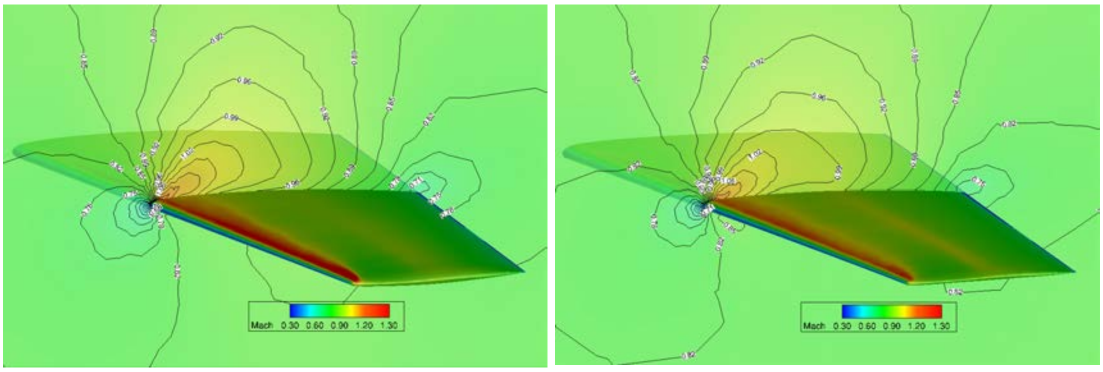


Figure 12: *Case IV*: Mach number iso-areas for the reference (left) and the optimized wing (right). Spanwise cuts at 50% of the wing span.

cretization schemes for the adjoint PDEs which reproduce the former. This ensures high accuracy in the computed gradient (as in discrete adjoint) with a "clear" code and minimal memory footprint of the adjoint code (as in continuous adjoint). A key advantage of the TDDC adjoint is that the developer of the method understands the physical meaning of the discretization schemes.

6 ACKNOWLEDGEMENT

This work is supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers" (Project Number: 3821).

REFERENCES

[1] O. Pironneau, *Optimal Shape Design for Elliptic Systems*. Springer-Verlag, New York, USA, 1982.

[2] A. Jameson, Aerodynamic design via control theory. *Journal of Scientific Computing*, **3(3)**, 233-–260, 1988.

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

[4] J. Elliott, J. Peraire, Practical three-dimensional aerodynamic design and optimization using unstructured meshes. *AIAA Journal*, **35(9)**, 1479-1485, 1997.

[5] E.J. Nielsen, W.K. Anderson, Aerodynamic design optimization on unstructured meshes using the Navier-Stokes equations. *AIAA Journal*, **37(11)**, 1411-1419, 1999.

[6] M. Giles, M. Duta, J.D. Muller, Adjoint code developments using the exact discrete approach. *15th AIAA Computational Fluid Dynamics Conference*, Anaheim, CA, USA, June 11–14, 2001.

[7] J. Peter, R. Dwight, Numerical sensitivity analysis for aerodynamic optimization. A survey of approaches, *Computers & Fluids*, **39(3)**, 373-391, 2010.

[8] P. He, Ch. Mader, J. Martins, K. Maki, An aerodynamic design optimization framework using a discrete adjoint approach with OpenFOAM. *Computers & Fluids*, **168**, 285-303, 2018.

[9] OpenFOAM-v2212, www.openfoam.com.

[10] X.S. Trompoukis, K.T. Tsiakas, V.G. Asouti, K.C. Giannakoglou, Continuous adjoint-based shape optimization of a turbomachinery stage using a 3D volumetric parameterization. *International Journal for Numerical Methods in Fluids*, DOI: 10.1002/fld.5187, 2023.

[11] A. Stuck, T. Rung, Adjoint implementation to viscous finite-volume pressure-correction methods. *Journal of Computational Physics*, **248**, 402–419, 2013.

[12] C.M. Rhie, W.L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal*, **21(11)**, 1525–1532, 1983.

[13] S.V. Patankar, *Numerical Heat Transfer and Fluid Flow*. Taylor & Francis, Hemisphere, NY, 1980.

[14] P. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, **43(2)**, 357–372, 1981.

[15] I.S. Kavvadias, E.M. Papoutsis-Kiachagias, K.C. Giannakoglou, On the proper treatment of grid sensitivities in continuous adjoint methods for shape optimization. *Journal of Computational Physics*, **301**, 1–18, 2015.

[16] V. Schmitt, F. Charpin, Pressure distributions on the ONERA M6 wing at transonic Mach numbers, experimental data base for computer program assessment. *Technical Report AGARD*, **38**, 1979.

[17] E.M. Papoutsis-Kiachagias, V.G. Asouti, K.C. Giannakoglou, K. Gkagkas, S. Shimokawa, E. Itakura, Multi-Point aerodynamic shape optimization of cars based on continuous adjoint. *Structural and Multidisciplinary Optimization*, **59(2)**, 675–694, 2019.

[18] J. Nocedal, S.J. Wright, *Numerical Optimization*. Springer, New York, NY 10013, USA, 2001.