Sensitivity computations by automatic differentiation of a CFD code based on spectral differences

J.I. CARDESA[a], C. AIRIAU[b]

a. jcardesa@imft.fr
b. christophe.airiau@imft.fr
Institut de Mécanique des Fluides de Toulouse (IMFT)
2 Allée du Professeur Camille Soula, 31400 Toulouse, France

Résumé :

Le contrôle et l'optimisation en mécanique des fluides dépendent de l'évaluation de sensibilités. Dans ce travail, on calcule des sensibilités en modifiant un code de simulation en dynamique des fluides d'ordre élevé basé sur une discrétisation spatiale en différences spectrales. L'approche est complètement discrète, en faisant appel à la différentiation algorithmique du code source original. L'outil employé, TAPENADE, peut alors générer deux codes source, un par mode de différentiation : tangent et adjoint. Les deux codes calculent des sensibilités pour un cas test d'écoulement bidimensionel et incompressible dans un carré périodique avec un double profil de cisaillement initial. Les sensibilités obtenues par les deux codes différentiés se distinguent par une erreur de l'ordre de la précision machine, et sont en bon accord avec celles approximées par différences finies (DF). Une comparaison et une discussion sur les temps de calcul sont menées, qui dans notre cas où l'on a un système à une seule entrée et sortie penchent naturellement en faveur du mode tangent. Ce dernier nécessite même un temps de calcul inférieur aux DF.

Abstract :

Control and optimization in fluid mechanics rely on the computation of flow sensitivities. We compute flow sensitivities by modifying a high-order computational fluid dynamics code which is spatially-discretized with spectral differences. Our approach is fully discrete, relying on algorithmic differentiation of the original source code. We obtain two transformed codes by using TAPENADE, one for each differentiation mode : tangent and adjoint. Both codes compute sensitivities in an unsteady test case of two-dimensional incompressible flow over a periodic cube with an initial double-shear profile. We find that the sensitivities from both differentiation modes agree to within machine accuracy, and compare well with those approximated by finite difference (FD) computations. We compare and discuss execution times, which in our case of a single-input/single-output problem are naturally in favor of the tangent code. The latter even outperforms the FD computation time.

Mots clefs : Sensitivity analysis, Flow Control, Adjoint Methods, Automatic Differentiation, Computational Fluid Dynamics

Introduction

The computation of sensitivities is a prerequisite to the implementation of optimization or flow control tools in simulation codes for aerodynamics. One route to estimate these sensitivities is to derive a new set of continuous equations, which need to be discretized in some way and added to the original simulation code [7]. An alternative path is to derive the discrete sensitivity equations by differentiating the discrete equations in the original simulation code. Two main advantages are often attributed to the second approach, which are the knowledge of the exact gradient of the discrete objective function [5] and the easier generalization to higher-order derivatives [6]. Its main disadvantage, however, is that differentiating spatially and temporally discretized equations by hand quickly becomes impractical beyond trivial schemes.

High-order methods adapted to compressible/incompressible fluid flow computations on complex geometries have been recently developed, making them suitable candidates to become industrial tools in the near future [1]. One such code is JAGUAR [3], developed at C.E.R.F.A.C.S., and which has been validated against compressible and incompressible test cases. Its excellent scalability, its ability to handle structured or unstructured grids, its optimized 6-step time integration scheme as well as its high-order spatial discretization based on spectral differences [8, 9] are all positive features which inevitably come at a price : the code is long and complex. Its differentiation for sensitivity computations is impractical. This is why we resorted to algorithmic differentiation software in order to automate this task.

Algorithmic Differentiation

Sometimes referred to as automatic differentiation (AD), it is a tool that takes computer code as input and provides computer code as output. The output code is in the same programming language as the original code. If the original code computed a set of dependent variables Y_i based on independent variables X_j, then the new code will compute those derivatives dY_i/dX_j selected by the user. Since JAGUAR is written in modern FORTRAN, we chose an AD tool compatible with key features from recent FORTRAN standards and which is currently still supported by a team of developers. This tool is called TAPENADE [2], developed by I.N.R.I.A. It can differentiate code in one of two modes : tangent or adjoint. The tangent mode is most suitable when one seeks derivatives of many Y_i with respect to few X_j, while the adjoint mode is mandatory whenever few Y_i are to be differentiated with respect to a large number of variables X_j. The transformations inflicted by the AD tool on the original code will be radically different between the two differentiation modes, with the adjoint mode producing a new code which is the most challenging to both generate and recognise.

Test Case

Two-dimensional double shear layer in a periodic square

We consider a two-dimensional incompressible flow in a square periodic domain spanning $L = 1$ in the streamwise (x) and vertical (y) directions. The velocity field at the initial instant t_0 is given by

$$u = U \tanh\left[r\left(y - 1/4\right)\right], \quad y \leq 1/2 \tag{1}$$

$$u = U \tanh\left[r\left(3/4 - y\right)\right], \quad y > 1/2 \tag{2}$$

$$v = U\delta \sin\left[2\pi\left(x + 1/4\right)\right], \tag{3}$$

case name	*'r40'*	*'r80'*	*'r160'*
r	40	80	160
U	1	0.7072	0.5001

TABLE 1 – The 3 shear parameters r considered in this study and their corresponding perturbation amplitudes U found from Equation (7). In all cases, the initial enstrophy is $\Omega_{ref} = 53.36$.

where all quantities are made non-dimensional with L and the streamwise reference velocity $U_0 = 1$ as follows :

$$t = \tilde{t}\, U_0/L, \quad y = \tilde{y}/L, \quad x = \tilde{x}/L, \quad U = \tilde{U}/U_0, \quad r = \tilde{r}\, L. \tag{4}$$

The parameters of the problem are U, r and δ. These are the streamwise velocity amplitude, the shear parameter and the ratio of vertical to streamwise velocity amplitudes, respectively. We set $\delta = 0.05$ for the remainder of this study so that it is no longer a free parameter. We analyze the evolution of the overall enstrophy Ω, defined as

$$\Omega = \int_0^1 \int_0^1 \frac{1}{2}\omega_z^2 \, dx \, dy, \tag{5}$$

where $\omega_z = \partial_x v - \partial_y u$ is the vorticity. It can be readily shown from Equations (1)-(3) and (5) that at $t = t_0$,

$$\Omega = \frac{U^2}{3}\left[6r\tanh(r/4) - 2r\tanh^3(r/4) + 3\delta^2\pi^2\right], \tag{6}$$

and we choose $r_{ref} = 40$ and $U_{ref} = 1$ to yield the initial enstrophy level $\Omega_{ref} = 53.36$ for all our considered test cases. This implies $U = U(r)$, which can be computed by re-arranging Equation (6) as follows :

$$U(r) = \sqrt{3\Omega_{ref}} \,/\, \left[6r\tanh(r/4) - 2r\tanh^3(r/4) + 3\delta^2\pi^2\right]^{1/2} \tag{7}$$

The initial Reynolds number $Re_0 = U_0 L/\nu = 1.176 \times 10^4$ was identical for all our test cases outlined on Table 1. We show snapshots of ω_z for the case $'r160'$ at four different instants on Figure 1.

Simulations

The Navier-Stokes equations are solved for the flow described in the previous section using JAGUAR, with the Mach number set to zero and using 360^2 degrees of freedom based on a regular square grid. The flux point locations followed Legendre collocation in transformed space, the CFL was kept constant at 0.5 and the Riemann solver was based on the Roe scheme. We compared the output from JAGUAR against a fully spectral in-house code for periodic incompressible flows (MatSPE), which allowed us to do grid resolution studies and to validate our simulations. We compare $\Omega(t)$ between the fully spectral code and JAGUAR on Figure 2 (*left*). The agreement between the two codes is extremely good. We note that the present study is based on the output and analysis of a serial version of JAGUAR, which we consider an essential first step before working on the parallel version of JAGUAR.

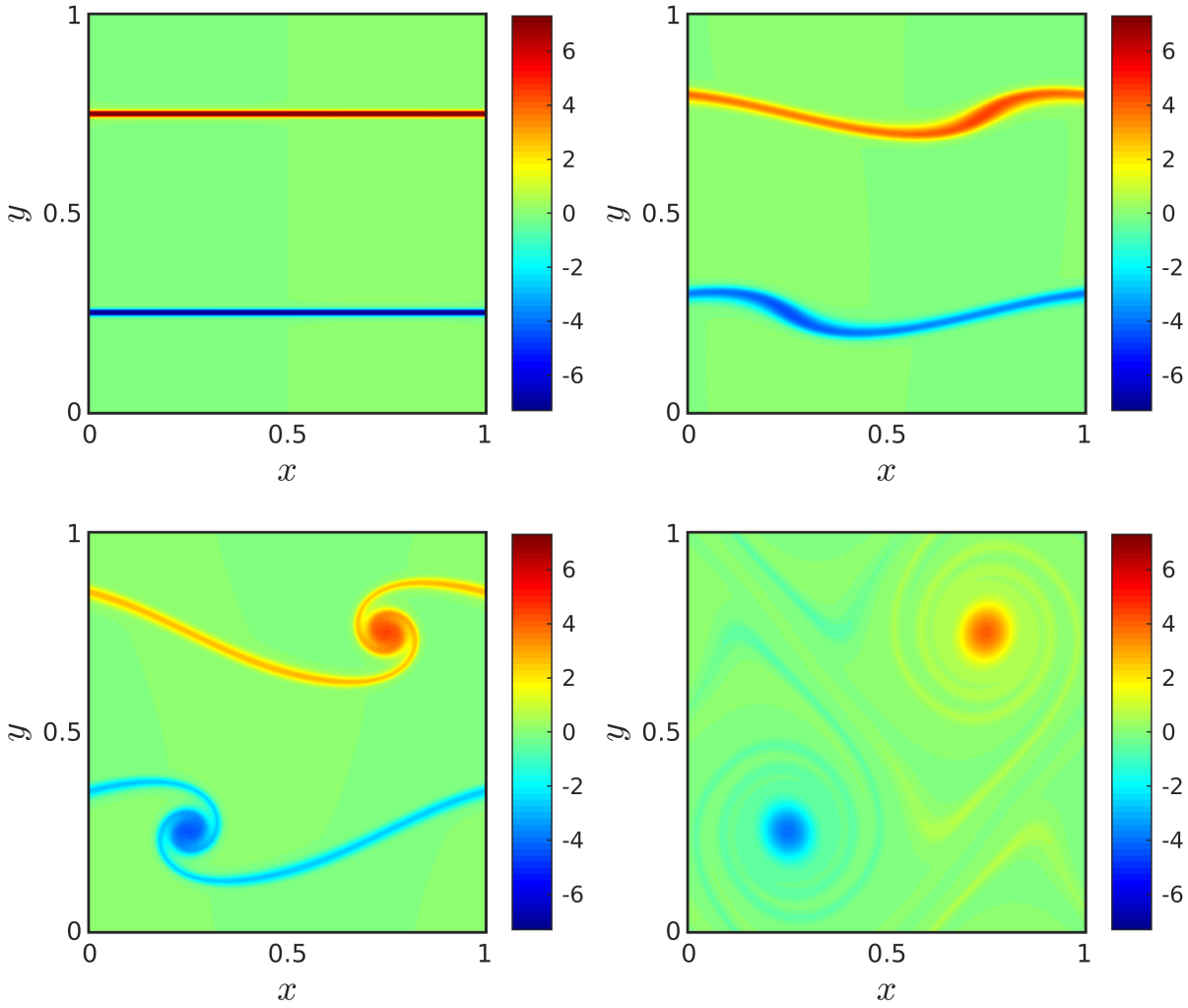

FIGURE 1 – Spatial distribution of $\omega_z/\sqrt{\Omega_{ref}}$ for case $'r160'$ at 4 instants $t\sqrt{\Omega_{ref}} = \{0, 7, 10, 23\}$ in the following respective order : top left, top right, bottom left and bottom right.

Sensitivities

We define the following two quantities

$$J_1 = \Omega(t) \tag{8}$$

$$J_2 = \int_0^T \Omega(t)\, dt \tag{9}$$

which, when differentiated with respect to r, yield the two sensitivities we compute by means of AD. dJ_1/dr is a time-dependent sensitivity, while dJ_2/dr is not. This implies dJ_2/dr can be computed through AD in both tangent and adjoint modes, while dJ_1/dr can only be computed in tangent mode. Both sensitivities will be compared against the finite difference (FD) estimates, computed with 2 independent realisations at r and $r + dr$, where $dr/r = 10^{-5}$.

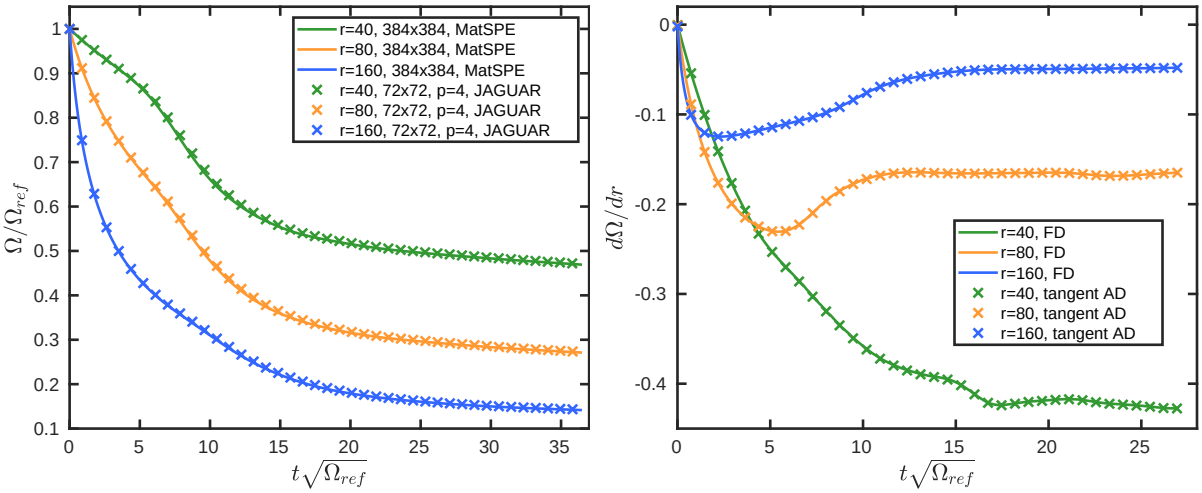

FIGURE 2 – (***left***) Evolution of $\Omega(t)$ at 3 different values of r, JAGUAR *vs.* fully spectral and incompressible code MatSPE. The legend includes the number of Fourier modes used in each direction for the MatSPE simulation (of which 1/3 are zero-padded for dealiasing), while the JAGUAR data includes parameter p which is the selected order of the spatial discretization of the spectral difference scheme. The grid used in the JAGUAR simulation was a 72×72 structured mesh which, together with the setting $p = 4$, yields 360 degrees of freedom (DoF) per spatial direction. So we are effectively comparing 256^2 DoF with MatSPE against 360^2 DoF with JAGUAR. (***right***) Sensitivities dJ_1/dr, computed with both finite differences (FD) and the tangent-differentiated code.

Results and discussion

We modify JAGUAR in order to wrap the part of the code which takes r as input and computes $J_1(t)$ as output. In this form, TAPENADE differentiates the wrapped portion of the code into one which computes both J_1 and dJ_1/dr (in tangent mode). We compare this output to the estimate of dJ_1/dr based on FD in Figure 2 (*right*). The agreement is excellent. The differentiated code is 1.9 times slower than the non-differentiated code. Since the FD computation takes exactly twice the execution time of the non-differentiated code, it appears that the higher accuracy of sensitivity computations from AD comes with the added benefit of a faster computation.

Similarly, we wrap the part of the code which takes r as input and computes $J_2(t)$ as output. We use TAPENADE to differentiate this version of the code in tangent and adjoint modes. The temporal integration is carried out from $t = 0$ to the $n - th$ iteration of the time integration loop in JAGUAR, with $n = 10^4$ and $n = 2 \times 10^4$. We show the output of both code sensitivities used with the 2 different time integration bounds on Table 2, together with the corresponding estimate computed with FD. The results for dJ_2/dr computed with tangent and adjoint derivation modes agree to within numerical round-off error, while the FD estimate agrees less with the previous two due to the lesser accuracy of FDs. Table 3 shows the time taken by the computations. Both the tangent and the adjoint codes that were run were kept as output by TAPENADE, so that no further code optimization was carried out. The adjoint mode is very greedy on time. Initially we ran out of memory since in adjoint mode, a forward code execution is first carried out where snapshots of the data buffers are stored at every iteration in the time integration loop. In order to remedy this problem, we use an option of TAPENADE to specify the number of snapshots one is willing to store, at the expense of additional computation time. This compromise reflects the challenge of using the adjoint mode for unsteady simulations as we do. We underline that

$r = 160$	$T = 10^4$ steps	$T = 2 \cdot 10^4$ steps
FD	-1.41688**7299988**E-003	-4.5418**73618868**E-003
Tangent	-1.4168835482492**68**E-003	-4.541868152846**180**E-003
Adjoint	-1.4168835482494**64**E-003	-4.541868152845**813**E-003

TABLE 2 – Estimates of dJ_2/dr according to 3 different computation methods. Only computed for the case with $r = 160$, and for 2 integration intervals.

$r = 160$	$T = 10^4$ steps	$T = 2 \cdot 10^4$ steps
FD	$1.8\ hrs(\times 2)$	$3.8\ hrs(\times 2)$
Tangent	$3.1\ hrs$	$6.2\ hrs$
Adjoint	$58.5\ hrs$	$123.2\ hrs$

TABLE 3 – Computation time taken by the 3 methods to compute dJ_2/dr. A factor of 2 appears in the finite difference (FD) case since 2 independent runs are required. Only computed for the case with $r = 160$, and for 2 integration intervals.

most adjoint CFD computations - for instance using commercial software [10] - are done in stationary situations through iterative solution procedures with a set number of iterations (of the order of a few hundred). In these cases, one can do away in adjoint mode by storing only the final fixed-point solution - see the F.A.Q. section in [11]. This is in stark contrast with our approach, where nonstationarity is intrinsic to our problem and the backwards time integration must be carried out based on storing and/or recomputing the intermediate results. Developing strategies to overcome this *computational barrier* is the object of current research [4]. We report an execution time in adjoint mode which is longer than in tangent mode by a factor of 20. We emphasize that in adjoint mode the computation time would remain approximately constant as we increase the number of variables J_2 is differentiated with respect to. In tangent mode, it would grow linearly with the number of additional variables J_2 is differentiated with respect to. Hence the adjoint mode could still be preferable in those cases where the sensitivity of J_2 is needed with respect to many more parameters.

Conclusions and future work

The maturity of TAPENADE to cope with a modern FORTRAN code for computational fluid dynamics has allowed us to differentiate modified code in order to obtain sensitivity estimates in a simple test case. Many lessons have been learnt on the way the source code should be modified for a correct differentiation by TAPENADE. Even though the differentiation process is automatic, it should not be regarded as a black box procedure. The output code requires careful analysis and checking, by means of comparison with FD or by using tools supplied with TAPENADE.

The simplicity of the test case, from a fluid mechanics point of view, does not preclude more complicated simulations from being amenable to the same type of code differentiation, since it is the complexity of the code rather than that of the flow which modulates the performance of TAPENADE. However, more ambitious simulations will inevitably require the differentiation of the parallel version of JAGUAR. We are currently working towards differentiating this parallel version, with message passing interface (MPI) directives being supplied to TAPENADE for tangent- and adjoint-mode differentiation. Our target is set on the implementation of an optimal control loop.

Acknowledgements

This work has been financed by a grant from the STAE Foundation for the 3C2T project, managed by the IRT Saint-Exupery. J.I.C. acknowledges funding from the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement n. PCOFUND-GA-2013-609102, through the PRESTIGE programme coordinated by Campus France. We are very grateful for the help provided by Laurent Hascoët and Valérie Pascual on how to use TAPENADE.

Références

[1] F.D. Witherden, A. Jameson, "Future Directions in Computational Fluid Dynamics", in : 23rd AIAA Computational Fluid Dynamics Conference, p. 3791, 2017.

[2] L. Hascoët, V. Pascual, "The Tapenade Automatic Differentiation tool : Principles, Model and Specification", in : ACM Transactions On Mathematical Software, Vol. 39 (3), 2013.

[3] A. Cassagne, J.F. Boussuge, N. Villedieu, G. Puigt, I. D'Ast, A. Genot, "JAGUAR : a new CFD code dedicated to massively parallel high-order LES computations on complex geometry", in : The 50th 3AF International Conference on Applied Aerodynamics (AERO 2015), 2015.

[4] J.C. Hückelheim, J.D. Müller, "Checkpointing with Time Gaps for Unsteady Adjoint CFD", in : Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences, Springer International Publishing, pp. 117–130, 2019.

[5] M. Giles, N. Pierce, "An introduction to the adjoint approach to design", in : Flow, turbulence and combustion, Vol. 65(3-4), pp. 393-415, 2000.

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

[7] B. Spagnoli, C. Airiau, "Adjoint analysis for noise control in a two-dimensional compressible mixing layer", in : Computers & Fluids, Vol. 37(4), pp. 475–486, 2008.

[8] Y. Liu, M. Vinokur, Z.J. Wang, "Spectral difference method for unstructured grids I : basic formulation", in : Journal of Computational Physics, Vol. 216(2), pp. 780–801, 2006.

[9] D.A. Kopriva, "A staggered-grid multidomain spectral method for the compressible Navier-Stokes equations", in : Journal of Computational Physics, Vol. 143(1), pp. 125–158, 1998.

[10] J. Munoz-Paniagua, J. García, A. Crespo, F. Laspougeas, "Aerodynamic optimization of the nose shape of a train using the adjoint method", in : Journal of Applied Fluid Mechanics, Vol. 8(3), pp. 601–612, 2015

[11] https ://www-sop.inria.fr/tropics/tapenade.html