11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
E. Oñate, J. Oliver and A. Huerta (Eds)

UNSTEADY CONTINUOUS ADJOINT METHOD USING POD FOR JET-BASED FLOW CONTROL

C. K. VEZYRIS*, I. S. KAVVADIAS*, E. M. PAPOUTSIS-KIACHAGIAS* AND K. C. GIANNAKOGLOU*

*National Technical University of Athens
Iroon Polytexneiou 9, 157 73, Zografou, Athens
e-mail: kgianna@central.ntua.gr

Key words: Unsteady continuous adjoint, Proper Orthogonal Decomposition, Incremental Singular Value Decomposition, Flow Control, Optimization

Abstract. An approximation method based on the Proper Orthogonal Decomposition (POD) method is used for the storage of the primal flow fields, needed during the solution of the unsteady adjoint equations in aerodynamic optimization problems. Here, without loss in generality, the presentation is restricted to flow control optimization; its extension to aerodynamic shape optimization is straightforward.

In this paper, the use of the POD, as an alternative to the check-pointing technique to handle unsteady flows, is demonstrated. POD approximates the time-evolution of the flow variables at each grid node, instead of repetitively re-computing them during the solution of the unsteady adjoint equations, while marching backwards in time. The solutions obtained with the POD method are compared to those reached by the check-pointing method, in terms of accuracy and overall simulation time. A parametric investigation of the POD implementation is carried out.

Based on the approximated primal flow fields, a flow control optimization, using pulsating jets, is performed using the unsteady continuous adjoint method. The jet positions are fixed whereas their amplitudes are optimized aiming at minimal time-averaged drag.

1 INTRODUCTION

The most efficient way to compute the gradient of an objective function with respect to (w.r.t.) a set of design variables is the adjoint method [13, 8].

In this paper, the continuous adjoint [12, 9, 5] method, where the adjoint PDEs are firstly derived and then discretized, is used. The state equations are the unsteady Navier-Stokes equations for incompressible fluids and, for unsteady flow problems, time-averaged performance metrics are used as objective functions. The flow is considered to be laminar, though previous works by the same group (such as [15]) guarantee that the method may readily accommodate exactly differentiated turbulence models.

In unsteady problems, adjoint information travels backwards w.r.t. to the primal one. The adjoint wake is formed upwind while the adjoint time progresses from the last to the first instant of the simulation. Also, for the numerical solution of the unsteady adjoint equations, the primal fields must be available at each time step. Theoretically speaking, this makes the storage of the primal solution fields mandatory. However, storing the computed primal fields for all time steps is very expensive memory-wise and alternatives are sought.

A common alternative is the binomial check-pointing technique, [7, 14]. In large scale problems, even though the binomial check-pointing technique has been proved to be optimal, it may lead to non-affordable computational cost due to the repetitive solutions of the flow fields as the solution of the adjoint equations progresses. In order to avoid repetitive computations of the primal flow fields, without using excessive amounts of memory, viable alternatives based on approximation of the time-evolution of the flow field can be devised. The approximation can be done with simple models, such as linear interpolation, quadratic models including cubic-splines, Fourier series in case of periodic phenomena, or any other interpolation method. In this paper, the POD technique is applied and assessed in terms of overall accuracy and simulation time.

After briefly presenting the primal and the adjoint equations, the POD method is discussed. Emphasis is laid on the incremental variant of the POD, which is used herein. In standard POD, the decomposition is performed only after the complete snapshot matrix is composed. However, this approach is of no interest since it requires full storage of the computed instantaneous flow fields. Instead, the incremental POD updates the decomposition at every new snapshot without burdening storage requirements. The method is used in flow control optimization problems, using pulsating jets in order to control the drag exerted on a circular cylinder. The same optimizations are also performed using the check-pointing method for the purpose of comparison.

2 FLOW MODEL AND OBJECTIVE FUNCTIONS

The flow is modeled by the Navier–Stokes equations for the unsteady laminar flow of an incompressible fluid. The primal equations are

$$R_i^v = \frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} - \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)\right] + \frac{\partial p}{\partial x_i} = 0 \ , \qquad i = 1, 2(3) \tag{1}$$

$$R^p = -\frac{\partial v_j}{\partial x_j} = 0 \tag{2}$$

where v_i and p stand for the velocity components and the static pressure divided by the density, respectively. To solve the primal equations, the SIMPLE algorithm [4] is used, with a staggered, cell-centered finite-volume discretization scheme for unstructured meshes.

In the optimization problems examined, pulsating jets, [16, 5], are used to minimize the time-averaged drag exerted on the body. The cartesian velocity components of each jet are given by

$$v_\lambda^m = \left(A^m \sin\left(2\pi f^m \left(t - f_0^m\right)\right) - A^m\right) n_\lambda \ , \quad \lambda = 1, 2(3) \tag{3}$$

where m is the jet counter, A^m is the amplitude, f^m the frequency and f_0^m the phase of each jet. Jets are aligned with the outwards, normal to the wall, unit vector n_λ. Positive A^m corresponds to blowing and negative A^m to suction. Frequencies and phases of all jets are fixed, $f^m = \frac{v_\infty}{d}$ and $f_0^m = 0$, as in [9], where v_∞ is the infinite flow velocity and d the diameter of the cylinder. The only design variables are the amplitudes A^m.

The time-averaged (squared) drag force is expressed as

$$J = \frac{1}{2T}\int_T D^2(t)dt \tag{4}$$

where T is the flow period. In the uncontrolled case, the flow period is the Karman vortices' period whereas in the controlled case T stands for the jets' period. D is the time-dependent drag force

$$D(t) = \int_{S_w} \left[p n_i - \nu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) n_j - \left|v_j n_j\right| v_i \right] r_i dS \tag{5}$$

where r_i are the components of the unit vector aligned with the farfield velocity and S_w stands for the solid wall boundary. The last term in eq. 5 stands for the contribution of jets on the forces acting upon the body, at the jets locations.

The derivative of the 'mean drag' objective function w.r.t. b_m is

$$\begin{aligned} \frac{\delta J}{\delta b_m} &= \frac{1}{T}\int_T \int_{S_w} D\Bigg(- \nu\left[\frac{\partial}{\partial x_j}\left(\frac{\partial v_i}{\partial b_m}\right) + \frac{\partial}{\partial x_i}\left(\frac{\partial v_j}{\partial b_m}\right)\right] n_j \\ &+ \frac{\partial v_i}{\partial b_m}\left|v_j n_j\right| + \frac{v_j n_j}{\left|v_j n_j\right|}\frac{\partial v_j}{\partial b_m} n_j v_i + \frac{\partial p}{\partial b_m} n_i \Bigg) r_i dS dt \end{aligned} \tag{6}$$

3 THE CONTINUOUS UNSTEADY ADJOINT METHOD

In order to derive the unsteady continuous adjoint equations, the augmented objective function L is defined as

$$L = J + \int_T \int_\Omega u_i R_i^v \, d\Omega dt + \int_T \int_\Omega q R^p \, d\Omega dt \tag{7}$$

where u_i and q are the adjoint velocities and pressure, respectively.

The derivatives of L w.r.t. the design variables b_m (here $b_m = A^m$), after applying the Leibniz theorem, become

$$\frac{\delta L}{\delta b_m} = \frac{\delta J}{\delta b_m} + \int_T \int_\Omega u_i \frac{\partial R_i^v}{\partial b_m} \, d\Omega dt + \int_T \int_\Omega q \frac{\partial R^p}{\partial b_m} \, d\Omega dt \tag{8}$$

The field adjoint equations are derived from eq. 8, after applying the Green-Gauss theorem to it and eliminating the field integrals depending on variations in the flow variables w.r.t. b_m. These are

$$R^q = \frac{\partial u_i}{\partial x_i} = 0 \tag{9}$$

$$R_i^u = -\frac{\partial u_i}{\partial t} - v_j \frac{\partial u_i}{\partial x_j} + u_j \frac{\partial v_j}{\partial x_i} + \frac{\partial q}{\partial x_i} - \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] = 0 \tag{10}$$

After eliminating the field integrals depending on variations of the flow quantities, eq. 8 becomes

$$\begin{aligned} \frac{\delta L}{\delta b_m} &= \frac{\delta J}{\delta b_m} + \int_\Omega \left[v_i \frac{\partial v_i}{\partial b_m}\right]_0^T d\Omega + \int_T \int_S D_i^u \frac{\partial v_i}{\partial b_m} dSdt + \int_T \int_S D^q \frac{\partial p}{\partial b_m} dSdt \\ &+ \int_T \int_S E_i^u \left[\frac{\partial}{\partial x_j}\left(\frac{\partial v_i}{\partial b_m}\right) + \frac{\partial}{\partial x_i}\left(\frac{\partial v_j}{\partial b_m}\right)\right] n_j dSdt \end{aligned} \tag{11}$$

where S_I is the inlet, S_O the outlet, S_∞ the freestream boundaries of the domain, $S = S_I \cup S_O \cup S_w$ or $S = S_\infty \cup S_w$ is the boundary and $D_i^u = u_i v_j n_j + \nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) n_j - q n_i$, $E_i^u = -\nu u_i$ and $D^q = u_j n_j$.

After substituting eq. 6 into eq. 11, the elimination of the boundary integrals including variations of the flow variables w.r.t. b_m gives rise to the adjoint boundary conditions at every time step. The instantaneous adjoint boundary conditions along S_w, S_I and S_O are S_w: $u_i = -\frac{D(t)}{T} r_i$ and S_∞: $u_i = 0$; for the whole domain Ω, the initial condition at $t = T$ is $u_i|_{t=T} = 0$.

The remaining terms in this development give the sensitivities of J w.r.t. the control variables $b_m = A^m$ which are given by

$$\begin{aligned} \frac{\delta J}{\delta b_m} &= \int_T \int_{S_w} \left[u_i v_j n_j - u_i \left|v_j n_j\right| + \nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) n_j \right. \\ &- \left. q n_i - \frac{v_j n_j}{|v_j n_j|} u_j v_j n_i\right] (sin(2\pi f^m (t - f_0^m)) - 1) n_i dSdt \end{aligned} \tag{12}$$

4 STORAGE OF THE PRIMAL FIELDS

The main difficulty in the solution of the unsteady adjoint equations, which march backwards in time, is that at each time-instant, the corresponding primal field must be available. Since the adjoint information travels backwards in time, the primal fields must be stored during the solution of the varying primal field. Such a full storage of the varying primal field is very expensive memory-wise and is usually replaced by either the check-pointing method [14, 7] or approximation methods.

The check-pointing technique is a compromise between memory consumption and CPU cost. Instead of storing the primal solutions at every time step, only those at a predefined number of time-instants, called check-points, are stored; from them, the primal solution at every other time-instant is re-computed. For a given number of time instants for which the corresponding flow field snapshots can be stored, the check-pointing technique corresponds to the optimal distribution of snapshots in time, which guarantees minimum re-computations of the primal field.

On the other hand, approximation methods offer the option of trading memory consumption with accuracy in the stored (approximated) primal fields, while the CPU cost remains the same. The accuracy of the approximated fields depends on the approximation method used. In this paper, the POD method is used as an approximation method for the primal fields.

4.1 Proper Orthogonal Decomposition

The method of POD is reviewed in [10]. Even though the method is generally know as POD, other names are in use, depending on the scientific field. The main idea derives from the concept of Principal Component Analysis (PCA). Other notable names of the method is the KarhunenLoéve decomposition, Eigenvalue decomposition of $A^T A$ and Singular Value Decomposition of A (A being the snapshot matrix defined below). POD results in a compact representation of the data at hand, while ensures that the representation in a reduced dimension space is optimal. Primarily, POD is used for Reduced Order Modelling (ROM), signifying the projection of a higher dimensional space onto a lower one. Various applications of POD can be found in the literature. A POD method for generating an aerodynamic database through parameter space (comprised by the angle of attack, Mach number and flare base radius range) for several test case analysing the number of modes required is presented in [11]. In [3], 'gappy' data sets are investigated, while afterwards they are used for inverse aerofoil design.

4.1.1 Mathematical formulation of POD/SVD

Since terms POD and SVD [6] can be used indifferently, in what follows the term SVD will mostly be used as it allows an easier connection to the incremental variant of the method. The snapshot matrix A corresponds to all spatial and temporal data, where n is the number of cells and m is the number of snapshots in the time domain. Below, a cell-centered storage of the finite-volume based CFD solver is assumed and the analysis is given separately for each flow variable.

4.2 Singular Value Decomposition

For $A \in \mathbb{R}^{nxm}$, SVD suggests that two orthogonal matrices exist, $U \in \mathbb{R}^{nxn}$ and $V \in \mathbb{R}^{mxm}$, such that $A = U\Sigma V^T$, where $\Sigma \in \mathbb{R}^{nxm}$ is a diagonal matrix, which includes the singular values of A in descenting order $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_{min(n,m)} \geq 0$. The columns of U and V are referred to as the left and right singular vectors of A, respectively.

SVD is used to set-up a Reduced Order Model with reduced storage requirements. To do so, the decomposition of the snapshot matrix is

$$A = U_{n\text{x}k} \Sigma_{k\text{x}k} V_{k\text{x}m}^T \tag{13}$$

where $k << m$, leading to lower storage demand. In the present study rank k is user-defined and this is evaluated in the parametric studies shown.

4.2.1 Incremental Singular Value Decomposition

In its original form, SVD is performed after the complete snapshot matrix A has been created. Therefore, in a problem similar to the one considered in the present paper, significant storage is required, without any apparent advantage. An alternative solution is the continuous update of the matrices generated by the SVD of small scaled data-set. This method is called Incremental SVD. Here, a brief presentation of the method is given, following closely the formulation developed in [2, 1].

Based on the predefined maximum rank k, the initial snapshot matrix is composed. Thus, $A \in \mathbb{R}^{nxk}$, where n is the number of grid cells and k the number of the already performed time steps. Then, a first SVD takes place resulting in $U_o \in \mathbb{R}^{n\text{x}k}$, $\Sigma_o \in \mathbb{R}^{k\text{x}k}$ and $V_o \in \mathbb{R}^{k\text{x}k}$. Since the singular value matrix is in descending order the last entry can be disposed off when the solution for the next $k+1$ time step becomes available. At this point, the corresponding SVD matrices are updated based on the following algorithm. For every new vector, ω, containing the flow variables, it is

$$p' = U^T \omega \tag{14}$$

$$g' = U\omega \tag{15}$$

$$r' = \omega - g' \tag{16}$$

The update of the current decomposition yields,

$$\begin{bmatrix} U\Sigma V^T & \omega \end{bmatrix} = \begin{bmatrix} U & \frac{r'}{\|r'\|} \end{bmatrix} \begin{bmatrix} \Sigma & p' \\ 0 & \|r'\| \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & 1 \end{bmatrix}^T \tag{17}$$

The decomposition

$$\begin{bmatrix} \Sigma & p' \\ 0 & \|r'\| \end{bmatrix} = U' \Sigma' V'^T \tag{18}$$

is then carried out.

The update of the singular value matrix, as well as the left and right subspaces, is expressed by three new matrices as follows

$$U_{new} = \left[U \ \frac{r'}{\|r'\|} \right] U' \tag{19}$$

$$\Sigma_{new} = \Sigma' \tag{20}$$

$$V_{new} = \begin{bmatrix} V & 0 \\ 0 & 1 \end{bmatrix} V' \tag{21}$$

5 Method Demonstration

The developed code was used for the reconstruction of the fields of the flow variables, such as the pressure p and velocity u. The results produced with the use of the incremental SVD algorithm are presented, in comparison with those computed by using binomial check-pointing.

The flow control optimization problem is handled by using 12 pulsating jets as discussed in section 2. Fig. 3 shows a comparison of the computed drag coefficient, by processing the primal solution and that obtained when a different number of maximum rank k is used in the incremental SVD algorithm. It is noticed that the phenomenon is captured adequately by using 10 modes. In contrast, when the first 5 orthogonal modes are used for the reconstruction of the flow variable field, the representation is rather poor. This results in a drag coefficient curve with a different phase and amplitude compared to the curve generated by the primal flow variable field.

Increasing the number of orthogonal bases matches the primal curve even more accurately. In fig. 3, it could be seen that the curve, which corresponds to 6 bases yields similar amplitude for the overall phenomenon, though having a different phase. Further increase in the number of bases may quite accurately capture the curve generated by the primal fields.

Based on the previous comparisons, it is expected that the optimization process, which makes use of the POD method, yields results very close to those obtained when binomial check-pointing is used. Indeed, fig. 4 shows that the three curves are practically identical. The optimization results taken with the utilization of POD have a perfect resemblance to results calculated by considering binomial check-pointing.

6 Conclusions

Incorporating the method of incremental SVD to the overall optimization process has a twofold positive effect. First, the storage required, when 10 bases are considered, is

$\sim 2\%$ of the one that would be bound with 500 check-points. Also, for an approximation of the flow variables with 10 bases the optimization for 25 cycles requires $\sim 31\%$ less computational time compared to the run using the check-pointing.

7 Acknowledgents

The research leading to these results has received funding from the People Programme (ITN Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement n^o317006 (AboutFLOW project).

REFERENCES

[1] L. Balzano and S.J. Wright. On grouse and incremental SVD. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on*, pages 1–4, Dec 2013.

[2] M. Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, 415(1):20 – 30, 2006.

[3] T. Bui-thanh, M. Damodaran, and K. Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, pages 1505–1516, 2004.

[4] L.S. Caretto, A.D. Gosman, S.V. Patankar, and D.B. Spalding. Two calculation procedures for steady three-dimensional flows with recirculation. In *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*, Paris, 1972.

[5] A. Carnarius, F. Thiele, Özkaya E., A. Nemili, and N.R. Gauger. Optimal control of unsteady flows using a discrete and a continuous adjoint approach. *System Modeling and Optimization, IFIP Advances in Information and Communication Technology*, 391:318–327, 2011.

[6] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, 78(7):808–817, 2000.

[7] A. Griewank and A. Walther. Algorithm 799: Revolve: an implementation of check-pointing for the reverse or adjoint mode of computational differentiation. *ACM Transactions on Mathematical Software (TOMS)*, 26(1):19–45, 2000.

[8] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[9] I.S. Kavvadias, G.K. Karpouzas, E.M. Papoutsis-Kiachagias, and K.C. Giannakoglou. Optimal flow control and topology optimization using the continuous

adjoint method in unsteady flows. In *Evolutionary And Deterministic Methods For Design, Optimization And Control*, Gran Canaria, Spain, 2013.

[10] Gaetan Kerschen, Jean-claude Golinval, ALEXANDERF. VAKAKIS, and LAWRENCEA. BERGMAN. The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview. *Nonlinear Dynamics*, 41(1-3):147–169, 2005.

[11] M. J. Mifsud, S. T. Shaw, and D. G. MacManus. A high-fidelity low-cost aerodynamic model using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 63(4):468–494, 2010.

[12] D.I. Papadimitriou and K.C. Giannakoglou. Aerodynamic shape optimization using first and second order adjoint and direct approaches. *Archives of Computational Methods in Engineering (State of the Art Reviews)*, 15(4):447–488, 2008.

[13] O. Pironneau. On optimum design in fluid mechanics. *Journal of Fluid Mechanics*, 64:97–110, 1974.

[14] Q. Wang, P. Moin, and G Iaccarino. Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation. *SIAM Journal on Scientific Computing*, 31(4):25492567, 2008.

[15] A.S. Zymaris, D.I. Papadimitriou, K.C. Giannakoglou, and C. Othmer. Continuous adjoint approach to the Spalart-Allmaras turbulence model for incompressible flows. *Computers & Fluids*, 38(8):1528–1538, 2009.

[16] A.S. Zymaris, D.I. Papadimitriou, E. M. Papoutsis-Kiachagias, K.C. Giannakoglou, and C. Othmer. The continuous adjoint method as a guide for the design of flow control systems based on jets. *Engineering Computations*, 30(4), 2013.

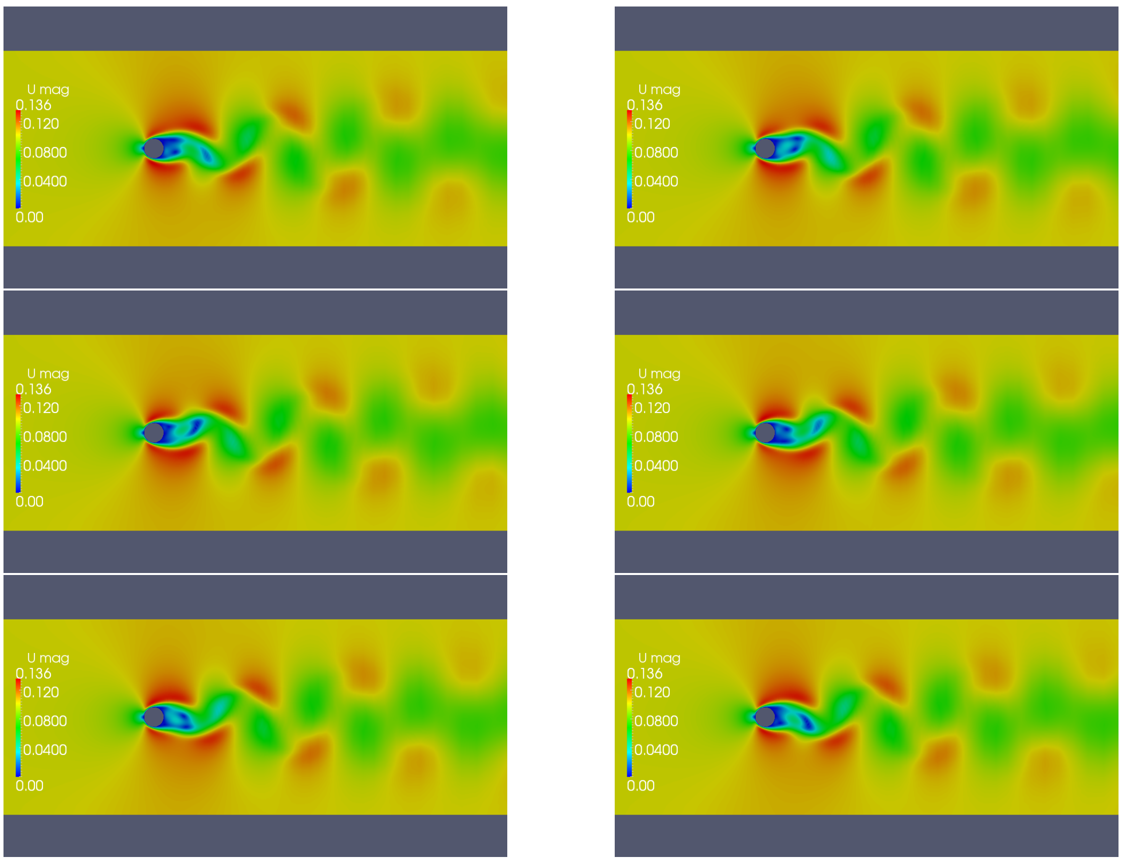

Figure 1: Six snapshots of the primal velocity magnitude calculated over a period of $T_{kv} \approx 0.59s$. The time-step increment for theses snapshots was equal to $T_{kv}/6$. These snapshots were taken once a periodic flow was established. No jets were applied in this case.

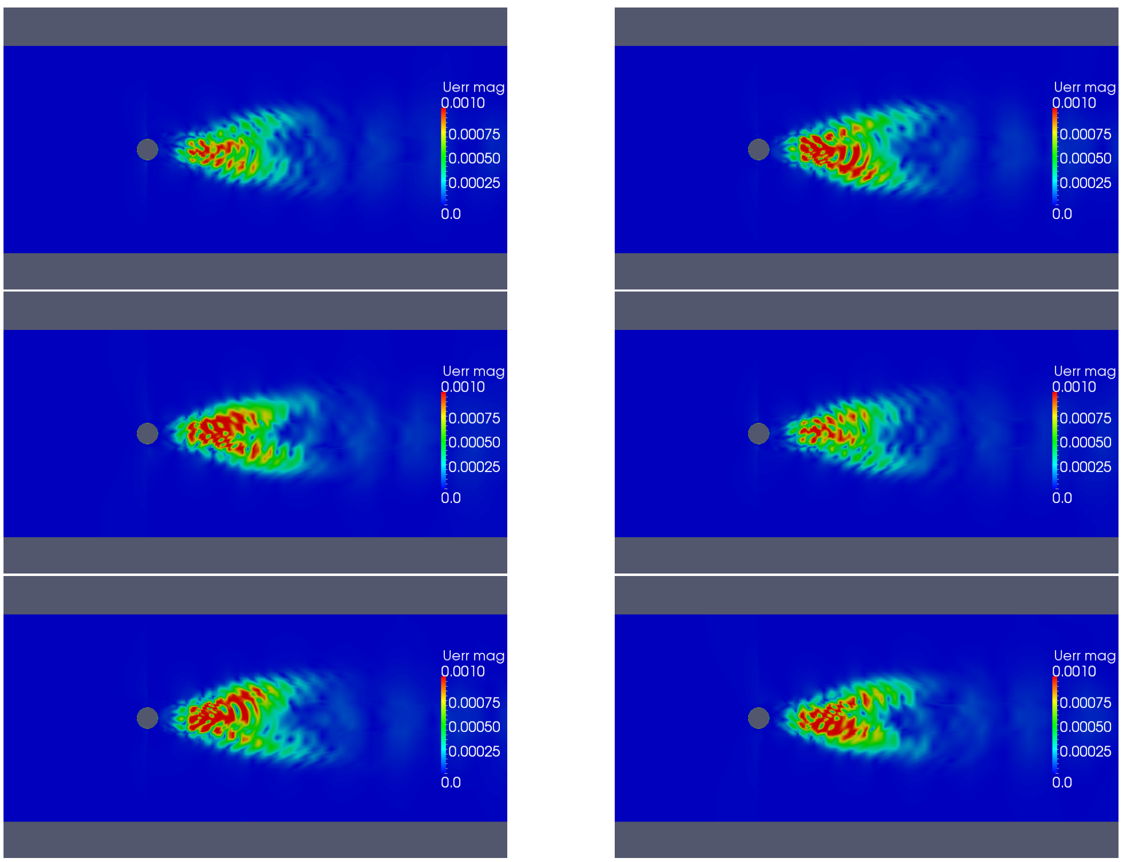

Figure 2: The absolute error in the velocity magnitude, for the 6 figures of fig. 2, between the velocity computed by the primal solver and that reconstructed by the POD method, using 10 bases.

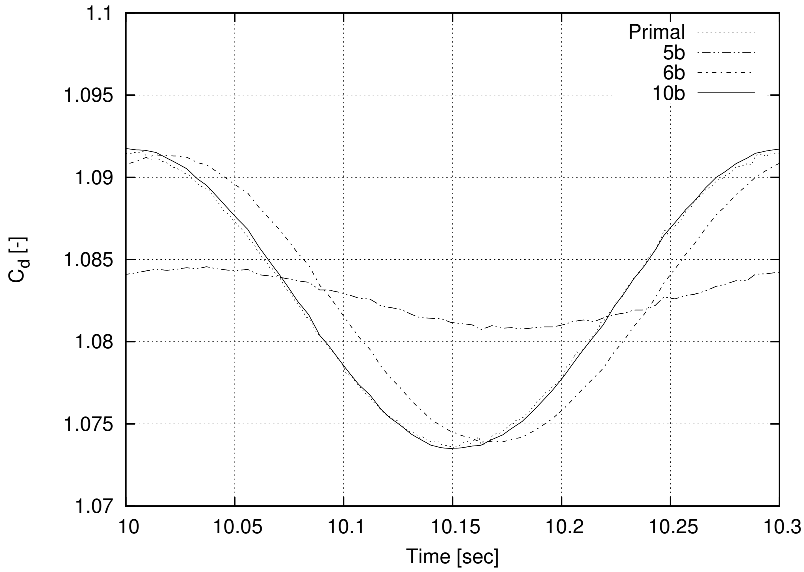

Figure 3: Drag coefficient, over time, as computed by the primal solver and POD approximations for different number of bases. For 5 orthogonal bases the approximation is not satisfactory, while for 10 it is almost perfect. It seems that, using more than 10 bases, no meaningful gain in accuracy is expected.

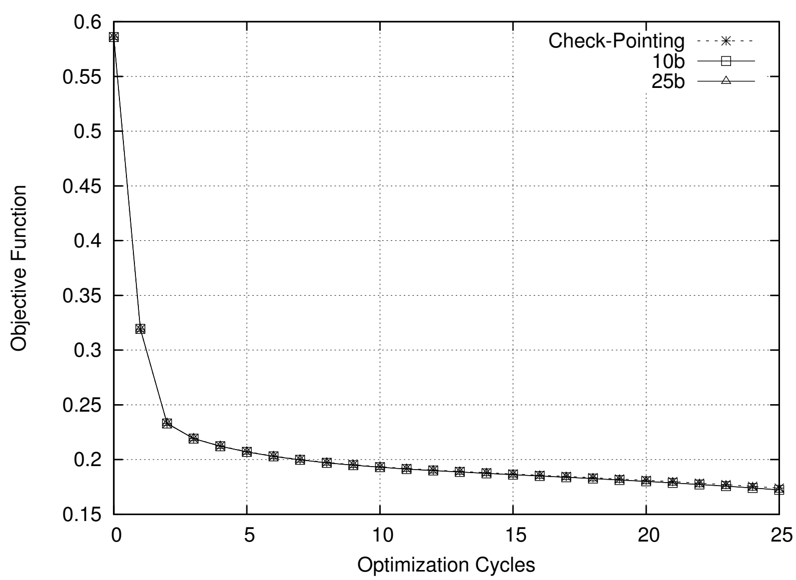

Figure 4: Evolution of the objective function obtained by using binomial check-pointing are approximated closely by the POD method using 10 orthogonal bases. As expected, further increasing the number of orthogonal bases further does not contribute any extra gain.