5^{th} GRACM International Congress on Computational Mechanics
Limassol, 29 June - 1 July, 2005

OPTIMAL GRID ADAPTATION THROUGH A POSTERIORI ERROR ANALYSIS

K.N. Gkagkas, D.I. Papadimitriou, K.C. Giannakoglou

Lab. of Thermal Turbomachines,
National Technical University of Athens,
P.O. Box 64069, 15710, Athens, Greece,
e-mail: kgianna@central.ntua.gr

Keywords: A Posteriori Error Analysis, Adjoint Method, Grid Adaptation, Aerodynamics, Turbomachinery

Abstract

A discrete adjoint approach to grid adaptation is presented. In particular, this paper is concerned with the prediction of integral flow quantities, such as the forces acting upon isolated or cascade airfoils, with user–defined accuracy. The aim is to achieve this accuracy through a small number of computations on successively adapted coarse grids. On each grid the flow and adjoint equations are solved. The adaptation sensor on each grid is computed in terms of flow and adjoint variables and residuals. The method application is considered to be succesful if the overall computational cost is less than that required to solve the problem on a very fine grid, safely adequate to reach the same accuracy but, unfortunately, not known in advance. An a posteriori error analysis formulation, that is the tool guiding the grid adaptation, is adjusted to an upwind compressible flow solver, investigated with respect to its parameters and extended to cascade flows.

1 INTRODUCTION

Often, the conclusive output of a flow analysis is one (or more) integral quantities which need to be computed with acceptable accuracy for engineering applications. Typical examples of integral outputs in aerodynamics or turbomachinery are the lift and drag of an isolated airfoil or the peripheral force acting on a cascade blade. This occurs frequently in design optimization problems, where the aerodynamic shape with the minimum or maximum value of an integral quantity (minimum drag, maximum lift, maximum loading in a peripheral cascade, etc) is sought. Since search methods (in particular, those based on evolutionary algorithms) require a great amount of evaluations to reach the optimal solution, the CPU cost per evaluation needs to be as low as possible. One way to minimize the cost is by reducing the grid size, without however damaging the prediction accuracy, at least for the integral output of interest.

In view of the above, the accuracy with which the entire flow field is calculated is of importance only so far as this affects this integral output. It is known that flow equation models, discretization schemes and grid resolution are the main issues which determine the accuracy in CFD computations. In what follows the inviscid flow equations are solved and the discretization scheme is a vertex–centered finite–volume method for unstructured grids, [8]. The inviscid fluxes are computed by means of the Roe's approximate Riemann solver [1] with second–order spatial accuracy [2]. In this framework, the integral output needs to be computed with user–defined accuracy through computations made on the coarser possible grid, i.e. with the minimum CPU cost.

This can be done through formulating and solving an appropriate adjoint (dual) problem. Using the flow and adjoint variables and residuals, a measure of the expected contribution of each grid node to the error in the integral output is estimated, [3, 4, 5, 6, 7]. This measure, in the form of a scalar sensor field over the grid edges, is used to selectively enrich the coarse grid in error inducing areas. The adaptation leads to a new grid on which the flow and adjoint equations are solved again and this procedure goes up as

long as high error inducing grid subsets are identified. The computation on the finally adapted grid yields the integral output with the desired accuracy. In the expense of this iterative algorithm, which relies upon the numerical solution of flow and adjoint equations on sequentially adapted grids, engineers overcome the necessity to generate extremely fine grids and the CPU cost reduces. We will show that the cost of solving the direct and adjoint equations on successive grids is lower than that of solving the flow equations on a very fine mesh.

The structure of this paper is as follows: The formulation of the adjoint problem and the definition of grid adaptation sensors are presented first. More about the method can be found in any of the aforementioned works on a posteriori error analysis. In the results section, the method is applied to flow problems concerned with the accurate prediction of lift and drag in isolated airfoils and the peripheral force in a compressor cascade. Through these cases, particular features of this method are highlighted.

2 A POSTERIORI ERROR ANALYSIS

Let $f(U)$ be an integral flow quantity, resulting from the integration of the flow variables U over a part of the domain boundary. A specific accuracy level is required for $f(U)$. Let us also consider two computational grids, namely the coarse (index H) and the fine (index h) ones, over the flow domain. U_H and U_h can be calculated through satisfying the flow equations, $R_H(U_H) = 0$ or $R_h(U_h) = 0$, on each grid. Then, $f_H(U_H)$ and $f_h(U_h)$ can be computed through the same integration scheme. Note that U_H and $f_H(U_H)$ are obtained using low–cost computations and are not so accurate since both the solution of the flow equations and integration are carried out using the coarse grid. In contrast, the solution of $R_h(U_h) = 0$ is computationally expensive and, practically, undesirable. Finally, starting from U_H, one may interpolate it onto the fine grid by means of a prolongation operator I_h^H to get

$$U_h^H = I_h^H U_H \tag{1}$$

and, $f_h(U_h^H)$, through integration over the fine grid.

By expanding the first–order Taylor series about $f_h(U_h^H)$ and $R_h(U_h^H)$, we get (= is used instead of $\approx$)

$$f_h(U_h) = f_h(U_h^H) + \frac{\partial f_h}{\partial U_h}|_{U_h^H}(U_h - U_h^H) \tag{2}$$

$$R_h(U_h) = R_h(U_h^H) + \frac{\partial R_h}{\partial U_h}|_{U_h^H}(U_h - U_h^H) = 0 \tag{3}$$

where $\frac{\partial f_h}{\partial U_h}|_{U_h^H}$ and $\frac{\partial R_h}{\partial U_h}|_{U_h^H}$ are computed using the prolongated field U_h^H. According to eq. 3, $U_h - U_h^H$ is given by

$$U_h - U_h^H = -\left[\frac{\partial R_h}{\partial U_h}|_{U_h^H}\right]^{-1} R_h(U_h^H) \tag{4}$$

which, upon substitution into eq. 2, provides an estimate of the integral functional as follows

$$f_h(U_h) = f_h(U_h^H) - \frac{\partial f_h}{\partial U_h}|_{U_h^H}\left[\frac{\partial R_h}{\partial U_h}|_{U_h^H}\right]^{-1} R_h(U_h^H) \tag{5}$$

The matrix inversion in eqs. 4 and 5 can be handled by introducing the adjoint variables Ψ, satisfying the so–called adjoint equations

$$\left[\frac{\partial R_h}{\partial U_h}|_{U_h^H}\right]^T \Psi_h|_{U_h^H} = \left(\frac{\partial f_h}{\partial U_h}|_{U_h^H}\right)^T \tag{6}$$

In terms of the adjoint variables, the functional $f_h(U_h)$ is merely expressed as

$$f_h(U_h) = f_h(U_h^H) - \left(\Psi_h|_{U_h^H}\right)^T R_h(U_h^H) \tag{7}$$

Eq. 7 can be considered as a better approximation to $f_h(U_h)$, compared to $f_h(U_h^H)$. However, solving for Ψ_h should be avoided, as we did for any other computation on the fine grid. So, instead of solving eq. 6, the adjoint equations are written and solved on the coarse grid, i.e.

$$\left[\frac{\partial R_H}{\partial U_H}\right]^T \Psi_H = \left(\frac{\partial f_H}{\partial U_H}\right)^T \tag{8}$$

Ψ_H is then interpolated over the fine grid nodes, through the prolongation operator J_h^H,

$$\Psi_h^H = J_h^H \Psi_H \tag{9}$$

So, instead of eq. 6, the following equation can be used

$$\tilde{f}_h(U_h) = f_h(U_h^H) - (\Psi_h^H)^T R_h(U_h^H) \tag{10}$$

3 GRID ADAPTATION

The last term in eq. 10 stands for a correction term (error) through which a better (than $f_h(U_h^H)$) estimate of f can be obtained, without however solving any partial differential equation on the fine grid. This is possible in the expense of additionally solving the adjoint equations on the coarse grid, i.e. with as much as twice the cost of computing U_H. However, if higher accuracy is needed, the coarse grid can be adapted, particularly in areas which induce the maximum error in the integral output and the same procedure is repeated. The grid adaptation must be driven by a sensor which is proportional to the aforementioned error. In [7], it is demonstrated that this error can be written as either

$$f_h(U_h) - f_h(U_h^H) = \left(\Psi_h^H\right)^T R_h(U_h^H) + \left(\Psi_h|_{U_h^H} - \Psi_h^H\right)^T R_h(U_h^H) \tag{11}$$

or

$$f_h(U_h) - f_h(U_h^H) = \left(\Psi_h^H\right)^T R_h(U_h^H) + \left\{R_h^\Psi\left(\Psi_h^H\right)\right\}^T (U_h - U_h^H) \tag{12}$$

where the residual of the adjoint equation on the fine grid is

$$R_h^\Psi(\Psi) \equiv \left[\frac{\partial R_h}{\partial U_h}|_{U_h^H}\right]^T \Psi - \left(\frac{\partial f_h}{\partial U_h}|_{U_h^H}\right)^T \tag{13}$$

From the above relations, an adaptation criterion, which takes into consideration the errors associated with both the flow and adjoint equations, is, [7]

$$\epsilon_k = \frac{1}{2}\sum_{l(k)} \left| \left[Q_h^H \Psi_H - L_h^H \Psi_H\right]_{l(k)}^T \left[R_h(L_h^H U_H)\right]_{l(k)} \right| + \left| \left[Q_h^H U_H - L_h^H U_H\right]_{l(k)}^T \left[R_h^\Psi(L_h^H \Psi_H)\right]_{l(k)} \right| \tag{14}$$

where L and Q are linear and quadratic interpolation schemes. Considering that the fine grid h is defined based on the existing coarse grid, ϵ_k is assigned to each coarse grid element (here, edge) and the summation over $l(k)$ takes into account any fine grid element associated with the coarse grid element k. Finally, the local adaptation parameter is defined by

$$\eta_k = \frac{\epsilon_k}{e_o} \tag{15}$$

where e_o is a user–defined allowed error for the integral output. Grid edges marked with $\eta_k > 1$ should be refined. The grid adaptation cycles terminate when there are no more edges marked for refinement.

4 THE OVERALL ALGORITHM – PRACTICALITIES

The repetitive grid algorithm which leads to the computation of $f(U)$ with prescribed accuracy includes the following steps.

1. Solve the flow and adjoint equations using the same coarse unstructured grid and compute $f(U_H)$.
2. Generate the fine grid (usually the "quadruple" grid, created by subdividing each coarse grid triangle into four triangles) and compute $L_h^H U_H$, $Q_h^H U_H$, $L_h^H \Psi_H$ and $Q_h^H \Psi_H$ over its nodes.
3. Compute the flow and adjoint equation residuals at the fine grid nodes, using $L_h^H U_H$ and $L_h^H \Psi_H$

4. Calculate the adaptation sensor η_k for each coarse grid edge and refine accordingly the coarse grid.

5. Update the coarse grid and return to step (1).

The numerical solution of the Euler equations is carried out through the aforementioned time–marching solver. The numerical inviscid fluxes crossing the interface between any pair of adjacent node–centered control volumes, are computed through the Roe's approximate Riemann solver, [1]. They become second–order accurate through variable extrapolation, [2], which requires the primitive variable gradient at the grid nodes. Gradients are computed over the triangular elements and, then, scatter–added to the nodes. The discretized system of equations is solved using the pointwise implicit Jacobi method. The matrix coefficient is filled in by considering only first–order convection terms; thus, the non–zero pattern of the coefficient matrix, in graph theory terms. coincides with that of the adjacency matrix in graph theory.

The previous assumption concerning the formation of the matrix coefficient is important. During the solution of eq. 8, $\left[\frac{\partial R_H}{\partial U_H}\right]^T$ is set equal to the transpose of the aforementioned coefficient matrix. Thus, in the sake of computational convenience, only the first–order terms are taken into account in the discrete adjoint equation. The r.h.s. term in the same equation, $\left(\frac{\partial f_H}{\partial U_H}\right)^T$ is expressed according to the trapezoidal integration law. In the present problems, the trapezoidal rule is exclusively used to compute integral outputs. Switching to a more accurate integration formula is possible and can be employed in a straightforward manner; however, this does not affect the conclusions drawn below.

5 RESULTS–DISCUSSION

The first case is concerned with the study of the flow developed around the isolated RAE2822 profile. The flow is inviscid with $M_\infty = 0.50$ and $\alpha_\infty = 3^o$. The maximum Mach number over the airfoil is about 0.92 and drag should approach zero. The initial grid is generated through the advancing front method after defining 112 nodes along the airfoil contour, clustered close to the leading and trailing edges. The initial grid (URG1) is fully unstructured, with as many as 1448 nodes and 2763 triangles. No particular care concerning the location of nodes is taken.

A couple of computations on successively refined grids are first made to obtain some reference results, concerning lift and drag coefficients. Starting from URG1, a grid (URG2) with as many as four times its triangles is generated by splitting each one of its triangles into four. The generation of the so–called *uniformly refined grids* (URG) is repeated three times. Over these four grids, the flow solver is used to predict the flow field and, through integration, to compute lift (C_l) and drag (C_d) coefficients. These values are tabulated in table 1 which also shows the CPU cost of each computation. All four computations are made with the same initialization; even if the computation on any fine grid could start from the converged solution on the previous (coarser) grid, this is avoided in the sake of fairness in the comparison of CPU costs. A solution is considered to be converged if the maximum residual becomes lower than 10^{-20} and this criterion determines the CPU cost of each computation. All computations are made on an Intel Pentium M processor at $1.80GHz$. Note that C_d approaches zero without reaching it, due to numerical diffusion.

Grid	Nodes	Triangles	C_l	C_d	CPU secs.
URG1	1448	2763	0.6674	0.008744	13.3
URG2	5659	11052	0.6907	0.004243	111.0
URG3	22370	44208	0.6975	0.003551	1124.1
URG4	88948	176832	0.6998	0.003463	11331.2

Table 1: Flow around the RAE2822 airfoil: Computed (reference) C_l and C_d values using four uniformly refined grids.

The grid adaptation with a posteriori error analysis is employed four times, with the same starting grid (URG1). Each time, a different e_o value is used; the e_o values used are listed in fig. 1 which shows the changes of C_l during the successive grid refinements in terms of grid size (left) and CPU cost (right). In all cases, a small number of adaptation cycles (around five) is needed. After the first or second cycle, the grid size increases slightly. On the other hand, the CPU cost of the flow analysis on each refined grid is, more or less, the same, since the starting flow field is interpolated from the converged solution on the coarser grid. All four runs show the correct trend of the C_l value computed over the finally refined grids.

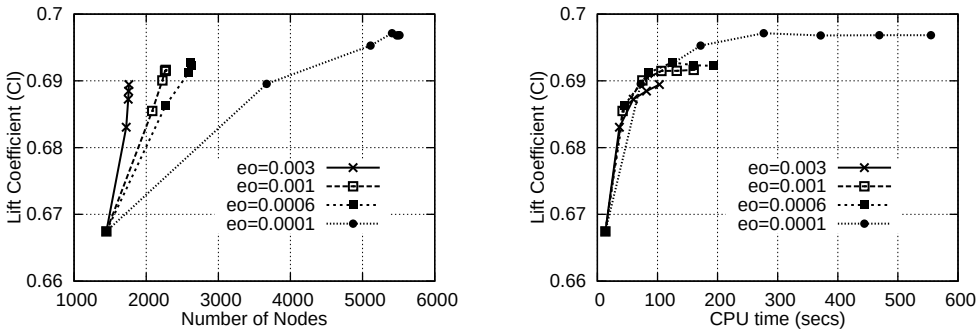

Figure 1: Flow around the RAE2822 airfoil, with user–defined accuracy in C_l: Computed C_l values during the grid adaptation driven by η_k, plotted in terms of the number of nodes (left) and CPU cost (right). The curves shown correspond to four different e_o values.

Fig. 2 compares the cost for obtaining the final C_l value, with user–defined accuracy, with that of using the URGs of table 1. Using the two runs with the lower e_o values, useful conclusions can be drawn. The $e_o = 0.0006$ run computes a better (slightly higher, i.e. closer to that of URG4) C_l value than that computed using URG2. This computation leads to an adapted grid with 2634 nodes (URG2 possesses 5659 nodes which, despite that, yields a slightly worse C_l value). The cost for the computation using $e_o = 0.0006$ is about 193 secs. Using the more strict C_l criterion ($e_o = 0.0001$), the C_l value computed using URG3 is obtained. However, fig. 2, the cost of this run (555 secs) is about half of the cost for URG3 (1124.1 secs). The economy in CPU cost is, in fact, noticeable.

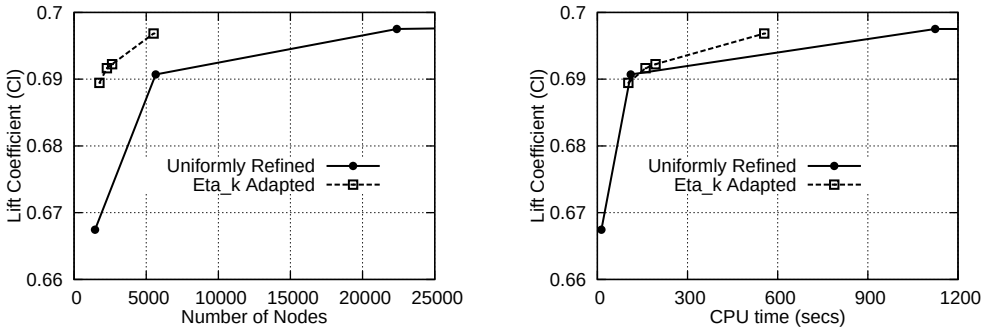

Figure 2: Flow around the RAE2822 airfoil, with user–defined accuracy in C_l: Comparison of C_l values computed using (a) a posteriori error analysis with grid adaptation and (b) the URGs, table 1.

Two more comparisons for the same case can be made using the plots shown in fig. 3. First, eq. 10 is used to correct the C_l computed on each grid during the successive adaptation cycles. Compared to fig. 1 (left) a slight additional improvement is shown in fig. 3 (left). Also, the C_l values on the successively refined grids (for $e_o = 0.0001$) are extrapolated according the Richardson's formula and this offers an additional way of exploiting the a posteriori error analysis method, fig. 3 (right).

The same airfoil, with the same infinite flow conditions and starting grid (URG1) is used for four new computations. This time, the target is the computation of C_d with prescribed accuracy. Four e_o values are used. The results obtained are shown in figs. 4, 5, according to the previously used presentation mode. The conclusions that can be drawn are similar. For instance, fig. 5 (left) shows that the same accuracy level can be obtained using a much coarser grid. From fig. 5 (left) and the CPU costs listed in table 1, it is also obvious that the lower e_o value ($e_o = 0.0001$) reaches the best value for C_d in about 3000 secs compared to the four times more expensive computation using URG4. As in the case of C_l correction, the correction of C_d through eq. 10 or Richardson extrapolation, (not shown here in the interest of space) can be used to further improve the C_d predictions. The starting grid as well as two finally adapted grids for $e_o = 0.0001$ used as threshold for C_l and C_d are shown in fig. 6.

The last case is concerned with the accurate prediction of the peripheral force coefficient in a compressor cascade. Here, a 2D controlled diffusion airfoil cascade is utilized and the integral quantity that needs to be computed with desired accuracy is the non-dimensional force component in the pitchwise direction.

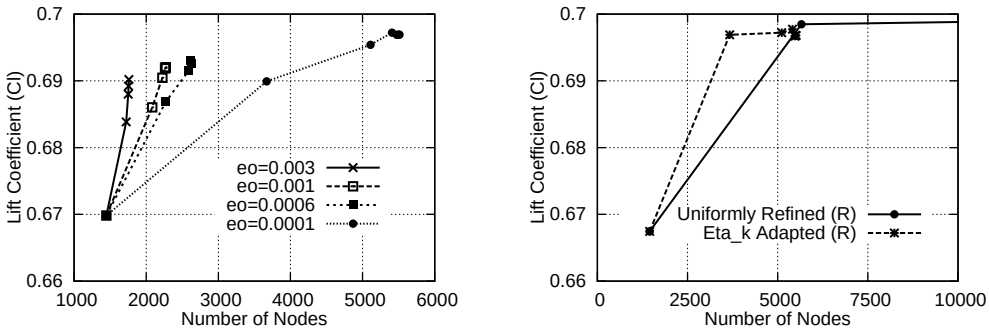

Figure 3: Flow around the RAE2822 airfoil, with user–defined accuracy in C_l: Corrected through eq. 10 (left) and Richardson extrapolated (right, for $e_o = 0.0001$) C_l values computed using (a) a posteriori error analysis with grid adaptation and (b) the URGs, table 1.

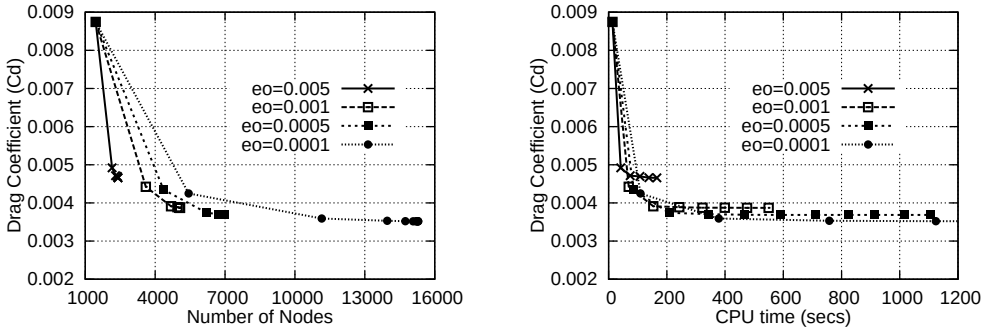

Figure 4: Flow around the RAE2822 airfoil, with user–defined accuracy in C_d: Computed C_d values during the grid adaptation driven by η_k, plotted in terms of the number of nodes (left) and CPU time (right).

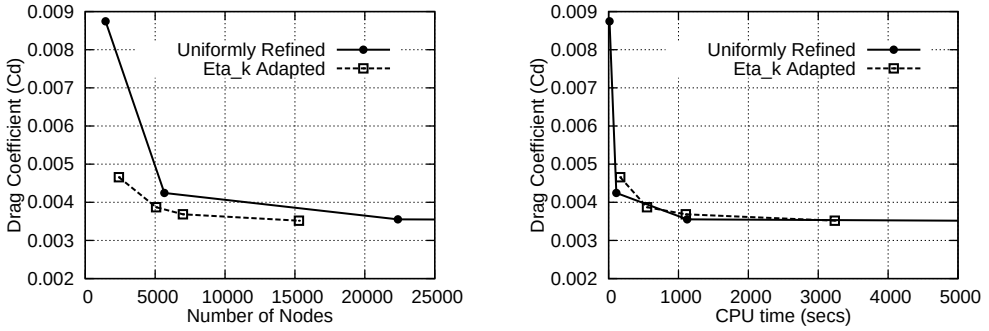

Figure 5: Flow around the isolated RAE2822 airfoil, with user–defined accuracy in C_d: Comparison of C_d values computed using (a) a posteriori error analysis with grid adaptation and (b) the URGs, table 1.

Since the flow is inviscid, this is derived through the integration of the pressure distribution around the airfoil and projection in the pitchwise direction. The coefficient of interest is denoted by C_{Fy}. The flow conditions are: $M_{2,is} = 0.37$ and $\alpha_1 = 47^o$. In all computations, the Jacobi method is used with CFL number equal to 50. The convergence criteria are the same as previously. Of course, modifications in the adjoint method are necessary, in order to account for periodicity.

Table 2 shows the three URGs used to get reference C_{Fy} values so as to compare the expected gain from the use of grid adaptation based on the a posteriori error analysis with uniformly (and, thus, uncontrollably expensive) grid refinement. URG1 is the starting grid in every subsequent calculation.

Fig. 7 shows the results of five computations with a wide span of e_o values. In contrast to the monotonic curves previously shown, in all cases, an overshooting in C_{Fy} is observed before reaching its final value, on the finally adapted grid. According to table 2, we may assume that the desired C_{Fy} value is about 0.033. We can see from fig. 7 that the lower the desired accuracy threshold e_o, the closer to 0.033 the terminal C_{Fy} value is captured.

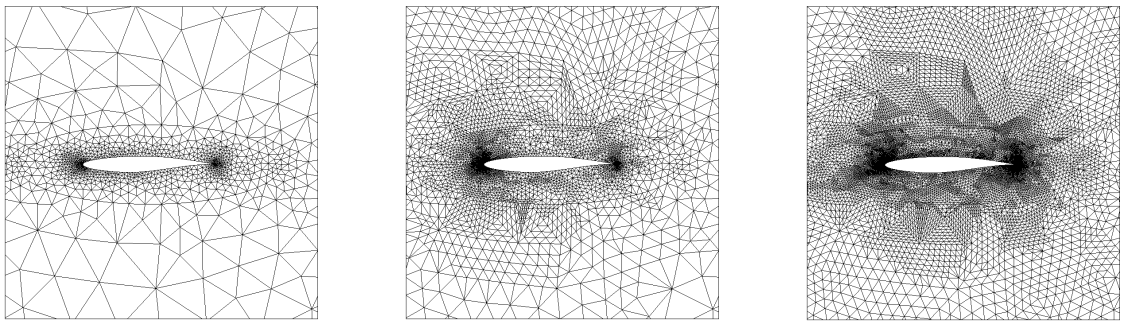

Figure 6: Flow around the RAE2822 airfoil: Left: starting grid (URG1), Middle: finally adapted grid (5571 nodes, 10777 triangles), computed using $e_o = 0.0001$ for C_l. Right: finally adapted grid (15290 nodes, 30170 triangles), computed using $e_o = 0.0001$ for C_d.

Grid	Nodes	Triangles	C_{Fy}	CPU secs.
URG1	1809	3327	0.032880	24.6
URG2	6945	13308	0.032988	189.8
URG3	27198	53232	0.033014	1472.6

Table 2: Flow in a 2D compressor cascade: Computed (reference) peripheral force coefficient values using three uniformly refined grids.

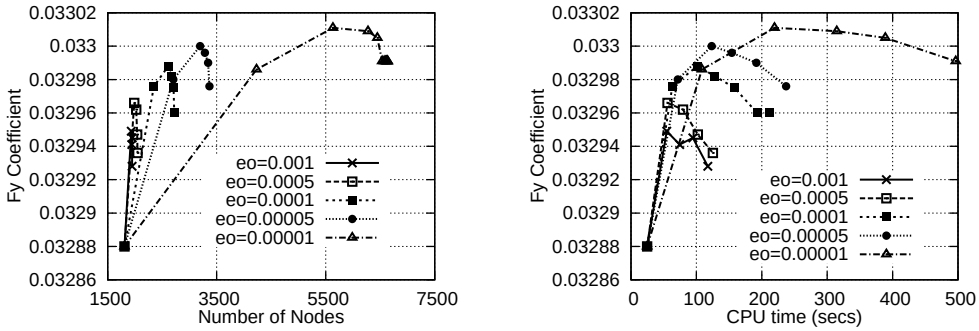

Figure 7: Flow in a 2D compressor cascade, with user–defined accuracy in C_{Fy}: Computed C_{Fy} values during the grid adaptation driven by η_k, plotted in terms of the number of nodes (left) and CPU cost (right), for five different e_o values.

Fig. 8 compares the results of grid adaptation through a posteriori error analysis with those obtained using the three URGs. In terms of grid size, the a posteriori error analysis based adaptation leads to much smaller numbers of grid nodes. The difference is not that important in terms of CPU cost; however, this depends on the convergence criteria used during the repetitive solution of the flow and adjoint equations and could be improved through the selection of different values. Fig. 9 presents the finally adapted grids for $e_o = 0.0005$ and $e_o = 0.00005$. It is obvious that the latter leads to a considerably finer grid close to the blade airfoil.

6 CONCLUSIONS

In this paper, the a posteriori error analysis method combined with grid adaptation techniques was presented, as a tool that ensures the computation of integral quantities with user–defined accuracy. The expected gain, which was confirmed by the examined cases, is that our goal can be achieved without using uncontrollably fine grids; local criteria can be used to drive the grid refinement in areas which induce the higher errors in the computation of the integral output of interest. The repetitive solution of the flow and adjoint equations are carried out on comparatively coarse grids, so the overall CPU cost reduces. Correction formulaes, based on either the adjoint formulation or Richardson extrapolations can be used to improve the prediction accuracy, with the same computing cost. In this paper, the a posteriori error analysis method is extended to periodic turbomachinery flows.

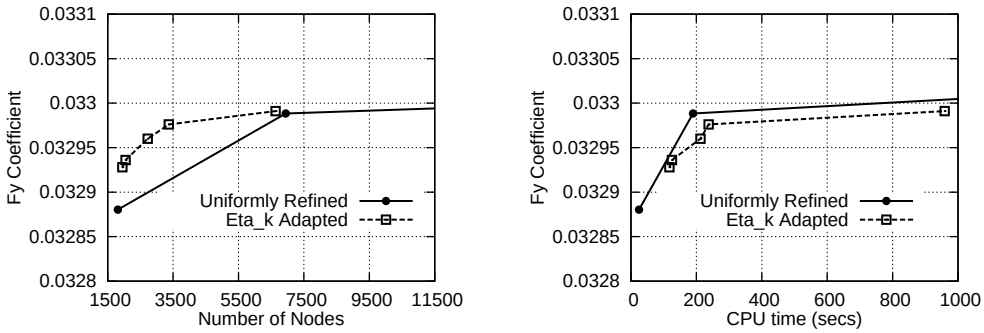

Figure 8: Flow in a 2D compressor cascade, with user–defined accuracy in C_{Fy}: Comparison of C_{Fy} values computed using (a) a posteriori error analysis with grid adaptation and (b) the URGs, table 2.

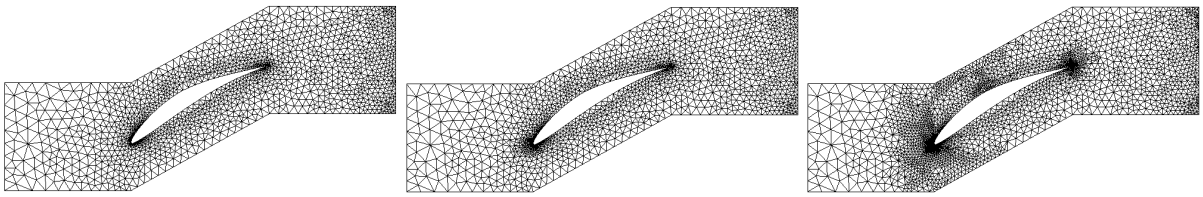

Figure 9: Flow in a 2D compressor cascade, with user–defined accuracy in C_{Fy}: Left: Initial coarse grid (URG1, 1809 nodes, 5136 triangles). Middle: finally adapted grid (2048 nodes, 3786 triangles), computed using $e_o = 0.0005$. Right: finally adapted grid (3368 nodes, 6354 triangles), computed using $e_o = 0.00005$.

ACKNOWLEDGMENT

This work was supported by the Iraklitos Basic Research Project, funded by the Hellenic Ministry of Education.

References

[1] Roe, P., Approximate Riemann Solvers, Parameter Vectors and Difference Schemes, J. of Comp. Phys., 43, 357-372, 1981.

[2] van Leer, B., Flux vector Splitting for the Euler Equations, Lecture Notes in Physics, Vol. 170, 405-512, 1982.

[3] Giles, M.B., On adjoint equations for error analysis and optimal grid adaptation in cfd, tech. rep., Oxford University Computing Laboratory, Numerical Analysis Group, Wolfson Building, Parks Road, Oxford, England, Report no. 97/11, September 1997.

[4] Venditti, D.A. and Darmofal, D.L., A multilevel error estimation and grid adaptive strategy for improving the accuracy of integral outputs, AIAA 99-3292, 1999.

[5] Venditti, D.A. and Darmofal, D.L., Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimension flow, Journal of Computational Physics, vol. 164, pp. 204-227, 2000.

[6] Venditti, D.A. and Darmofal, D.L., A grid adaptive technology for functional outputs of compressible flow simulations, AIAA 2001-2659, 2001.

[7] Venditti, D.A. and Darmofal, D.L., Grid adaptation for functional outputs: Application to two-dimensional inviscid flows, Journal of Computational Physics, vol. 176, pp. 40-69, 2002.

[8] Koubogiannis, D.G., Poussoulidis, L.C., Rovas, D.V. and Giannakoglou, K.C., Solution of Flow Problems Using Unstructured Grids on Distributed Memory Platforms, Computer Methods in Applied Mechanics and Engineering, 160, pp. 89-100, 1998.