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)

A PARALLEL AGGLOMERATION MULTIGRID METHOD FOR THE ACCELERATION OF COMPRESSIBLE FLOW COMPUTATIONS ON 3D HYBRID UNSTRUCTURED GRIDS

GEORGIOS N. LYGIDAKIS[*] AND IOANNIS K. NIKOLOS[†]

[*,†]School of Production Engineering and Management
Technical University of Crete
University Campus, GR-73100 Chania, Greece
[*]e-mail: glygidakis@isc.tuc.gr
[†]e-mail: jnikolo@dpem.tuc.gr

Key Words: *Compressible Flow, Node-Centered Finite-Volume Scheme, Reynolds-Averaged Navier-Stokes (RANS) equations, Agglomeration Multigrid, Parallelization.*

Abstract. Although the unstructured grids may accurately describe a complex geometry along with reduced requirements of user interaction for their generation/adaptation, the corresponding solvers appear to be relatively inefficient comparing to the structured ones. This drawback can be significantly alleviated by the employment of the agglomeration multigrid method, based on the solution of the flow problem on successively coarser grids, derived from the initial finest one through the fusion of the adjacent finite control volumes. In this study a parallel agglomeration multigrid methodology is presented, developed to enhance an existing academic CFD code, named *Galatea*, which employs the RANS (Reynolds-Averaged Navier-Stokes) equations along with appropriate turbulence models, to simulate inviscid and viscous laminar or turbulent compressible fluid flows. The fusion strategy considers the neighboring control cells' merging on a topology-preserving framework, which resembles the advancing front technique, while depending on the flow type (inviscid, laminar or turbulent) and subsequently on the type of the initial finest grid (tetrahedral or hybrid), an isotropic agglomeration or a directional one can be performed, either selected by the user. The multigrid accelerated iterative solution of the flow and turbulence models is achieved via the Full Approximation Scheme (FAS) in a V-cycle process, incorporated in the Full Multigrid (FMG) Scheme, considering as such two stages, the preliminary and the main one. The proposed methodology is validated against well-documented test cases, which concern inviscid, laminar and turbulent compressible flow over a rectangular wing with NACA0012 airfoil and the DLR-F6 aircraft model, demonstrating its capability for improved computational performance.

1 INTRODUCTION

During the last decades the CFD (Computational Fluid Dynamics) algorithms have been revealed to be a valuable tool in the aerospace industry as they allow for the prediction of the aerodynamic behavior of complete aircrafts in a relatively short period of time. Their popularity was significantly augmented by the utilization of the unstructured grids, enhancing them with the capability of describing complex geometries, i.e. aircraft configurations, along with the minimum user interaction for their generation/adaptation[1,2]. Nevertheless, the unstructured grid solvers remain inferior in terms of efficiency compared to the structured mesh ones[3-7]; a remedy to this shortcoming is the multigrid method, originally introduced to increase the convergence rate of the numerical solution of elliptic problems[8,9], but since then a significant effort has been also exerted for the compressible flow hyperbolic ones [1,3]. Its main idea derives from the observation that most of the well-established iterative methods converge more slowly on finer spatial resolutions due to the relatively inefficient damping of low frequency errors; the solution of the flow problem on a coarser grid results to the transformation of these errors in high frequency ones and subsequently to the increase of the convergence rate. Various types of the pre-mentioned methodology have been developed, whose differences are mainly identified on the generation of the coarser grids, i.e. geometrical (nested or non-nested) multigrid[3], agglomeration multigrid[1,3,6], etc. and the associating relation between the utilized grids, such as the FMG[1] and the FAS[3].

In this work a parallel agglomeration multigrid technique is developed to enhance an existing academic CFD code, named *Galatea*[10], which employs the RANS equations along with appropriate turbulence models (k-ε, k-ω and SST) for the prediction of inviscid, laminar and turbulent compressible fluid flows on hybrid unstructured grids, composed of tetrahedral, prismatic and pyramidical elements. The fusion of the adjacent control cells is achieved in a way analogous to the advancing front technique; the procedure begins with the isotropic fusion of the boundary nodes and extends to the internal ones either isotropically or directionally, depending on the flow type[6]. According to pre-defined rules, each node is examined in order its control volume to be merged with its neighboring (not yet agglomerated) cells and create a new virtual *supernode* of the coarser polyhedral control volume grid. Special care is required for the *ghost* nodes at the overlapping region between adjacent sub-domains (for parallelized solution), to be fused or remain singletons according to the agglomeration of their corresponding *core* nodes at the neighboring partition[11]. For the iterative procedure, the FAS is implemented via a V-cycle strategy[3,5,6,7], which considers the solution of approximated flow and turbulence models' equations for the coarser levels, utilizing the edge-based structure of the algorithm. The pre-mentioned scheme is implemented in two main stages, the preliminary and the main one, resembling in that way the FMG method[1,3]; the solution begins from the coarsest grid (preliminary stage) and, as the cycles advance, extends to the finer levels up to the finest one (main stage). The interaction between two successive spatial resolutions is obtained via the restriction of the conservative variables and the corresponding flux balances from the finer to the coarser grid, as well as the prolongation of the same variables' corrections from the coarser to the finer level[3,5,6,7]. The proposed methodology is evaluated against benchmark test cases, which consider inviscid, laminar and turbulent compressible flow over a rectangular wing with NACA0012 airfoil and the DLR-F6 aircraft model, revealing its capability for improved computational performance.

2 FLOW SOLVER

For the prediction of compressible fluid flows *Galatea* employs the RANS equations, which are obtained by Favre averaging the Navier-Stokes PDEs (Partial Differential Equations) and described in differential form as[3]

$$\frac{\partial \vec{W}}{\partial t}+\frac{\partial \vec{F}^{inv}}{\partial x}+\frac{\partial \vec{G}^{inv}}{\partial y}+\frac{\partial \vec{J}^{inv}}{\partial z}-\frac{\partial \vec{F}^{vis}}{\partial x}-\frac{\partial \vec{G}^{vis}}{\partial y}-\frac{\partial \vec{J}^{vis}}{\partial z}=\vec{S} \tag{1}$$

where $\vec{W}$ is the conservative variables' vector, $\vec{F}^{inv},\vec{G}^{inv},\vec{J}^{inv}$ and $\vec{F}^{vis},\vec{G}^{vis},\vec{J}^{vis}$ are the inviscid and viscous flux vectors respectively, and $\vec{S}$ is the vector of the source term which is assumed to be equal to zero in this study[1,3,10]. The computation of these vectors is performed using the dimensionless form of the primitive variables, density ρ, velocity components u, v, w and pressure p, as well as the stress and thermal tensors based on the Boussinesq assumption along with the dynamic viscosity μ, the latter evaluated via the Sutherland law[3,11]. The set of flow equations is completed with the perfect gas state equation[3,10,11]. Finally, for turbulent flow simulations, three two-equation models[3,10], namely k-ε, k-ω and SST, have been incorporated, interacting with the RANS PDEs mainly via the turbulent dynamic viscosity μ_t, while no additional mechanism has been incorporated for transition modeling yet.

The discretization of the computational field is succeeded employing a node-centered finite-volume method for both the flow and turbulence models; the median-dual control volume of a node is constructed by connecting lines defined by edge midpoints, barycenters of faces and barycenters of elements, sharing this node[2,3,10]. Considering this approach, equation (1) is integrated at each control volume as

$$\left(\frac{d\vec{W}}{dt}\right)_P V_P+\sum_{Q\in K_N(P)\,and\,Q_{out}}\vec{\Phi}_{PQ}^{inv}-\sum_{Q\in K_N(P)}\vec{\Phi}_{PQ}^{vis}=\vec{0} \tag{2}$$

where V_P is the volume of the control volume of node P, while the second and the third terms represent the inviscid and the viscous fluxes, calculated at the internal and boundary interfaces of the examined polyhedral control volume.

For the computation of the inviscid fluxes, a one-dimensional Riemann problem is considered at each edge (at the interface of the adjacent control cells), while its solution is achieved implementing the Roe's approximate Riemann solver[12], along with a second-order spatial accurate scheme, based on the MUSCL method and jointed with appropriate slope limiters (Van Albada-Van Leer, Barth-Jespersen and Min-mod, either selected by the user)[2,3,10]. An edge-based data structure is utilized, resulting in the evaluation of the pre-mentioned fluxes with a single edge-loop. For the computation of the viscous fluxes, the gradients of the velocity components and temperature have to be previously defined at the middle of each edge. Two methods have been incorporated in *Galatea*: an element based one, considering the gradients' evaluation at edge-dual volumes consisted of all the neighboring elements of the examined edge[3,10], and a nodal-averaging one, using the gradients at the end-points of each edge to calculate the corresponding derivatives at the mid-point[3]. The fluxes for the turbulence models are calculated in the same way, except for the convective ones, for which a first-order accurate simple upwind scheme is employed as the diffusive term is the sovereign one[3,10]. The contributions of the boundary surfaces are added to the flux balances

of the corresponding nodes; for solid walls a free-slip condition is assumed in case of an inviscid flow and a no-slip one in case of a viscous problem, while for inlet/outlet regions a similar treatment to the internal edges is used for both models, except for the inlet area where Dirichlet conditions are imposed for the turbulent variables.

Since the flux balances of the computational nodes are defined, equation (2) is iteratively relaxed, by implementing either an explicit scheme with a four-stage Runge-Kutta method (RK(4)) or a point implicit one, using the Jacobi or the Gauss-Seidel algorithm[3,10]. For the employment of these schemes, the pre-mentioned equation is transformed as follows for the current iteration *n*

$$-V_P \frac{\Delta \vec{W}_P^{n+1}}{\Delta t_P} = \vec{R}_P^n \tag{3}$$

where Δt_P is the pseudo-time step, evaluated with a local time-stepping technique[3].

In addition, *Galatea* solver is enhanced with a parallelization capability, in order to efficiently encounter flows requiring large grids with millions or tens of millions of nodes for their representation. The parallelization strategy is based on the domain decomposition approach along with the MPI (Message Passing Interface) protocol[2,10,13]; it begins with the division of the computational domain into a number of sub-grids, using the METIS application, which derives a number of sets of non-common nodes, named as *core* nodes. The division extends accordingly to the elements of the initial mesh; nevertheless some of them, positioned at the interface of the partitions remain incomplete, as not all of their nodes belong to the same sub-domain. The missing nodes are added as *ghost* nodes to each partition, completing these elements and constructing the overlapping area, which allows at next for the interaction between the adjacent partitions via the exchange of data (flow variables, turbulence model variables and nodal gradients) through the MPI protocol.

3 AGGLOMERATION MULTIGRID METHOD

3.1 Agglomeration strategy

Either the isotropic or the directional agglomeration are performed in each partition on a topology-preserving framework resembling the advancing front technique[6], limited though by pre-defined rules concerning the implementation of the boundary conditions as well as the fusion of the *ghost* nodes at the overlapping region. More specifically, such rules are the following: a) An internal node can be agglomerated only with another internal node to create a new *supernode*. b) Similarly, the control volume of a boundary node can be merged only with an adjacent control volume, belonging to the same boundary surface. c) A node belonging to two or more boundary-condition-type closures isn't agglomerated and remains singleton at the next multigrid level[6]; for example a node belonging simultaneously to an inlet and a solid wall boundary surface can't be agglomerated. Nevertheless, nodes whose one boundary-condition-type closure concerns symmetry boundary conditions, are excluded from this limitation and they can be merged with other nodes with the same two boundary condition types. d) A boundary node belonging to two or more boundary slope discontinuities, such as a node placed at the corner of two different surfaces, isn't agglomerated and remains singleton at the next multigrid level[6]. e) Only the *core* nodes at

the overlapping region (sending data) of each sub-domain are considered for agglomeration during the main procedure; since this procedure at each agglomeration level is completed the *ghost* nodes (receiving data) are fused accordingly to the agglomeration of their corresponding *core* nodes at the adjacent sub-domains.
Based on these limitations, the isotropic agglomeration procedure (in case of a tetrahedral initial grid) is performed in the following sequence: a) The constrained (by the pre-mentioned limitations) nodes remain singletons and are simply transferred to the next multigrid level, while the new *supernodes* are also marked for limited agglomeration. b) The nodes belonging to solid wall boundary surfaces are marked for agglomeration, creating the list of the so-called *seed* nodes[6,11]; if a sub-domain doesn't include such nodes, the *core* nodes at the overlapping region are marked and included in this list instead. c) The main agglomeration procedure begins with looping over the *seed* nodes, to examine whether their adjacent (not yet agglomerated) nodes can be agglomerated, by checking the pre-mentioned limitations. If no limitation is identified their control volumes are merged with this of the *seed* node, creating the polyhedral control volume of a new *supernode* (Figure 1); if a limitation is identified or all the neighboring nodes have been already agglomerated, the examined *seed* node becomes a singleton.

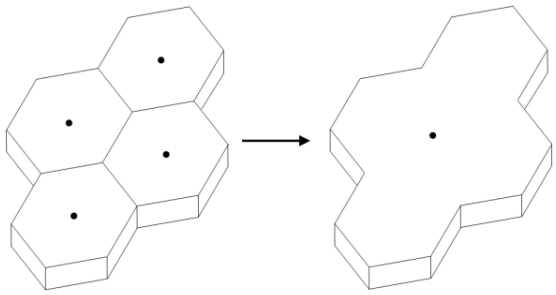

Figure 1: Generation of a *supernode* via agglomeration at the prismatic region of a hybrid grid

d) Another loop is performed in order the derived unconstrained singletons to be examined for fusion with their adjacent *supernode* with the less number of merged nodes, as well as the *supernodes* completely surrounded by another *supernode* to be identified and merged with the surrounding one[11]. e) A new list of *seed* nodes is constructed, with the nodes touched by the agglomeration front, which are actually the non-agglomerated neighboring nodes of the lastly fused ones. The nodes are imported in this list according to a priority hierarchy, based on the number of times a node is touched by the agglomeration front and the number of its adjacent control cells. Thus, the set of nodes with the maximum touch number will be examined first, with the procedure beginning from the node with the maximum number of neighboring nodes. f) Steps (c) to (e) are repeated until all the *core* nodes are agglomerated, or simply transferred as singletons to the next multigrid level. g) The *ghost* nodes at the overlapping region of each partition are agglomerated or become singletons according to the corresponding *core* nodes' fusion at the adjacent sub-domain; virtual *ghost supernodes* are generated in that way, as the number of their merged nodes may differ from this of the corresponding *core supernodes*. h) The new virtual *superedges*, connecting the *supernodes* are constructed; for evaluating the new vectors of the *superinterfaces/superedges*, the internal vectors at each super control volume are deleted while the external ones are simply summed. i) The main procedure is

concluded with the identification of the derived singletons and the marking of their adjacent *supernodes* to become also singletons at the next multigrid level, in order steep irregularities to be avoided in the next agglomerated grid. k) In case an even coarser mesh is required the previous steps are repeated.

For viscous flow simulations, implemented on a hybrid unstructured grid, directional agglomeration is performed instead, adding some extra steps to the pre-mentioned isotropic fusion process. Before the beginning of the main procedure the boundary prismatic nodes (belonging to boundary prismatic elements) are identified and assigned an index, i.e. an ascending number. The index of each boundary node is assigned then to the corresponding nodes of the next prismatic layers, resulting in that way in lines of prismatic nodes with the same index (implicit lines)[6]. Utilizing such indexes, five more steps are added before step (b) of the previously described isotropic procedure, in the following sequence: 1) A loop over the boundary prismatic nodes is performed, in order their control volumes to be merged with these of their adjacent nodes belonging to the same surface (bottom of implicit lines), similarly to step (c) of the isotropic agglomeration; the indexes of the merged nodes of each *supernode* are stored in a data structure in order the nodes of the next layers to be fussed accordingly. 2) A list of *seed* nodes is constructed, including the nodes touched by the agglomeration front, which are actually the prismatic nodes of the next layer. 3) The neighboring nodes of the *seed* ones are examined for agglomeration similarly to step (c) of the isotropic strategy, limited though by another constraint defining that a node can be fussed with its adjacent one only if their indexes are associated, namely if their corresponding boundary nodes (at the bottoms of the implicit lines) have been merged. 4) Steps 2 and 3 are repeated until all the prismatic nodes are agglomerated or transferred as singletons to the next multigrid level. In that way, nodes belonging to the same implicit line at two successive prismatic layers are fussed together. In addition, depending on the examined problem, an extra constraint may be imposed, limiting the number of merged nodes included in a *supernode*[14]; for example, at each fusion level a maximum of 4 (or less) nodes are allowed to be merged on the solid wall boundaries. In order to preserve the topology of the initial finest grid, up to the coarsest generated mesh, after the first agglomeration the nodes of the lowest prismatic layer are not allowed to be merged with the nodes of the next prismatic layer; the same stands for the nodes of the second prismatic layer after the second agglomeration, and so forth (Figure 2). 5) Steps 1 to 4 are performed for the prismatic nodes, whose implicit line bottom nodes belong to an adjacent partition, assuming as bottom node the *core* one at the overlapping region.

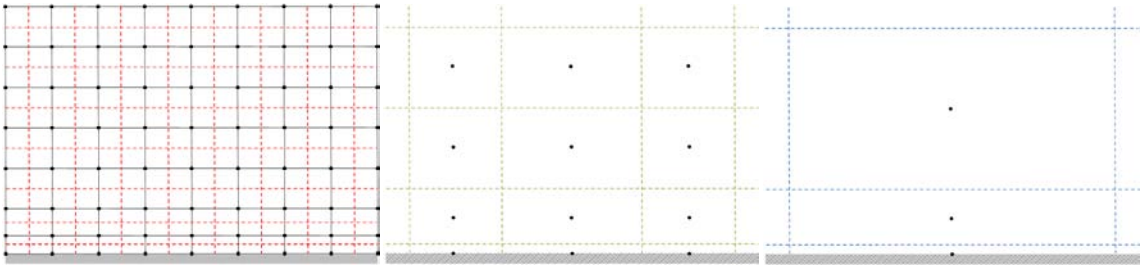

Figure 2: Agglomeration at the quadrilateral region of a hybrid grid (level 1-red, level 2-green and level 3-blue)

Since the steps of the directional agglomeration are complete the isotropic procedure is

implemented for the rest of the computational field; the initial *seed* list includes the adjacent tetrahedral nodes (touched by the agglomeration front) of the already fussed prismatic ones.

3.2 Flux computation and iterative solution

For the multigrid accelerated iterative solution of the flow and turbulence models' equations the FAS[1,3,6] is implemented; equation (3) is solved at the initial finest grid, while at the coarser ones an approximate version of the same equation is used instead[3]. Each cycle begins with the relaxation of equation (3) at the initial mesh (denoted with subscript *h*), while, since the models' conservative variables are updated, they are restricted by means of interpolation to the coarser resolution (denoted with subscript *H*), using the restriction operator $I_{W,h}^{W,H}$ as

$$\vec{W}_{P,restricted}^{H} = I_{W,h}^{W,H}\vec{W}_{p}^{h} = \sum \vec{W}_{p}^{h} \cdot V_{p} \big/ V_{P} \tag{4}$$

where $\vec{W}_{P,restricted}^{H}$ is the volume-weighted sum of the conservative variables' vectors of nodes *p* included in the *supernode P*. Except for the conservative variables, the corresponding flux balances are also restricted to the coarser grid, using a similar restriction operator $I_{R,h}^{R,H}$ as

$$\vec{R}_{P}^{H} = I_{R,h}^{R,H}\vec{R}_{p}^{h} = \sum \vec{R}_{p}^{h} \tag{5}$$

where $\vec{R}_{P}^{H}$ is actually the sum of flux balances of the merged nodes *p* to the *supernode P*. Since restriction is completed approximate equation (3) is solved for the coarser mesh, which is obtained by substituting the right hand side term with the following one[3]:

$$\vec{R}_{P,FAS}^{H} = \vec{R}_{P}^{H}\left(\vec{W}_{P}^{H}\right) + \underbrace{I_{R,h}^{R,H}\vec{R}_{p}^{h} - \vec{R}_{P}^{H}\left(I_{W,h}^{W,H}\vec{W}_{p}^{h}\right)}_{A_H} \tag{6}$$

The first term represents the current flux balance of *supernode P* in the coarse grid, while A_H is the forcing function based on the restricted flux balance as well as the balance computed using the restricted conservative variables. Considering the edge-based structure of *Galatea* the evaluation of the convective and diffusive fluxes in the coarser mesh is a straightforward procedure; a first-order accurate spatial scheme is employed for the inviscid terms, while for the viscous ones the necessary gradients are either computed by the nodal averaging technique or volume-weighted restricted from the finest grid. The restriction and relaxation procedures are implemented similarly up to the coarsest generated grid.

The corrections of the conservative variables are transferred then to the finer grid using a simple point injection scheme, namely the prolongation operator, as

$$\Delta\vec{W}_{p}^{h} = I_{W,H}^{W,h}\Delta\vec{W}_{P}^{H} = \Delta\vec{W}_{P}^{H} = \vec{W}_{P}^{H} - \vec{W}_{P,restricted}^{H} \tag{7}$$

used at next to update the corresponding computed variables at the finer mesh. If an even finer mesh exists the prolongation continues similarly. In that way a $V(\nu_1,\nu_2)$ cycle[3,6] is completed, where ν_1 denotes the number of iterative relaxations before the restriction and ν_2 the corresponding number after the prolongation; for inviscid and laminar flows a *V(1,0)* cycle is employed with the explicit scheme, while the *V(2,1)* is utilized with the implicit one[3]. In case of a turbulent flow the *V(2,1)* and *V(3,3)* cycles are applied for the explicit and the implicit schemes respectively[6]. Since the FAS for the flow model is completed, it is implemented in

the same way to update the conservative variables of the turbulence model.

The FAS[1,3,6] is incorporated in the FMG[1] scheme[14]; therefore the solution is divided in two stages, namely the preliminary and the main one. At the preliminary stage the governing equations are solved beginning from the coarsest generated grid and, as the iterative cycles accumulate, the FAS extends successively to the finer meshes up to the finest initial one, at which point the main stage begins[14].

4 NUMERICAL RESULTS

4.1 Inviscid flow over a rectangular wing with a NACA0012 airfoil

The first test case concerns inviscid flow with Mach number *0.8* and angle of attack 0^o over a rectangular wing with NACA0012 airfoil. The utilized grid consists of *625,250* nodes and *3,500,243* tetrahedrons, while for parallel computation on a workstation with an AMD FX$^{(tm)}$-8120 eight-core processor at *3.1* GHz it was divided in four sub-domains. For the multigrid accelerated iterative solution three coarser grids were created via isotropic agglomeration. In Figure 3 the initial (non-agglomerated) and the first level agglomerated volume grid are illustrated, along with the obtained distribution of pressure coefficient *Cp* at the mid-span of the wing, compared with the corresponding one of the reference solver[15]. For the evaluation of the proposed multigrid methodology both an explicit (Runge-Kutta) and an implicit scheme (Jacobi) were employed along with a second order accurate scheme jointed with the Min-mod limiter. In Figure 4 the convergence histories for density (per number of iterations and time) are presented for the single-grid simulation as well as for the multigrid one, using the explicit scheme with a constant CFL number equal to unity; an iterations and time speed-up coefficient equal to ~11.0 and ~9.5 respectively was achieved for a final residual equal to 1.0E-10. Using a linearly increased CFL number from 1.0 to 5.0 the corresponding history for the implicit scheme is illustrated in Figure 5; a time speed-up coefficient equal to ~4.0 was obtained along with a significantly reduced computation time (~3.5 hours).

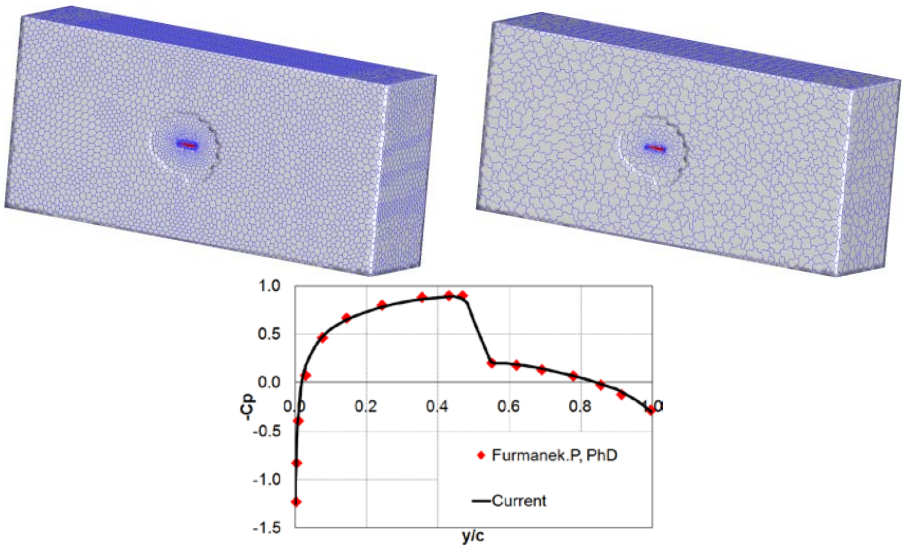

Figure 3: Non-agglomerated and first-level agglomerated volume grid; distribution of *Cp* at the mid-span

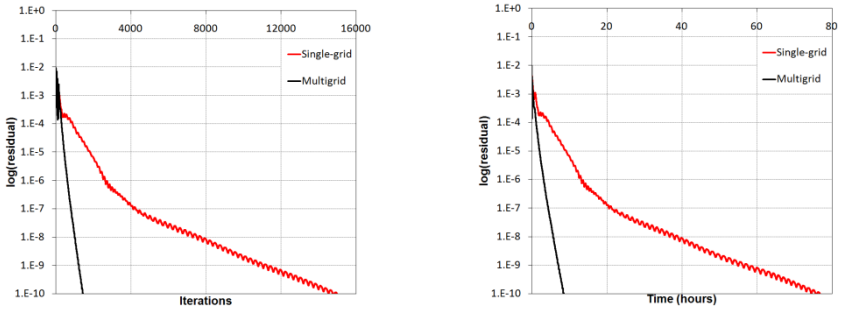

Figure 4: Convergence history (density), per number of iterations and time, for the explicit scheme

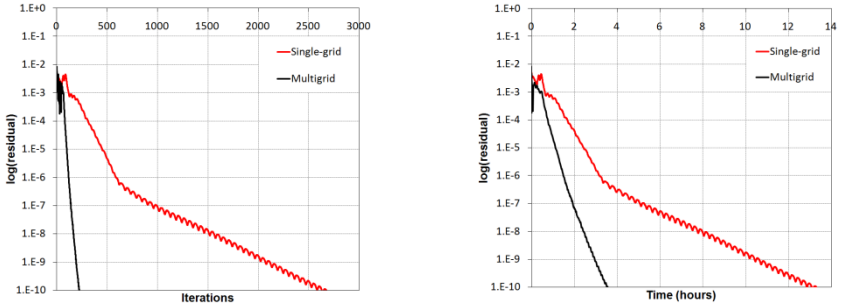

Figure 5: Convergence history (density), per number of iterations and time, for the implicit scheme

4.2 Laminar flow over a rectangular wing with a NACA0012 airfoil

The second benchmark problem considers a rectangular wing with a NACA0012 airfoil against a laminar flow with 0^o angle of attack, Mach number *0.85* and Reynolds number *500*. The utilized hybrid grid is composed of *305,978* nodes, *566,245* tetrahedrons and *394,760* prisms, while for parallelization (on the same computational system) it was divided in four partitions. The explicit scheme was employed with a CFL number equal to *0.5*. Three coarser grids were generated via the directional agglomeration method. In Figure 6 the symmetry surface of the initial and the first level agglomerated volume grid is presented along with the obtained distribution of *Cp* at the mid-span, compared with the corresponding one of a reference solver[16].

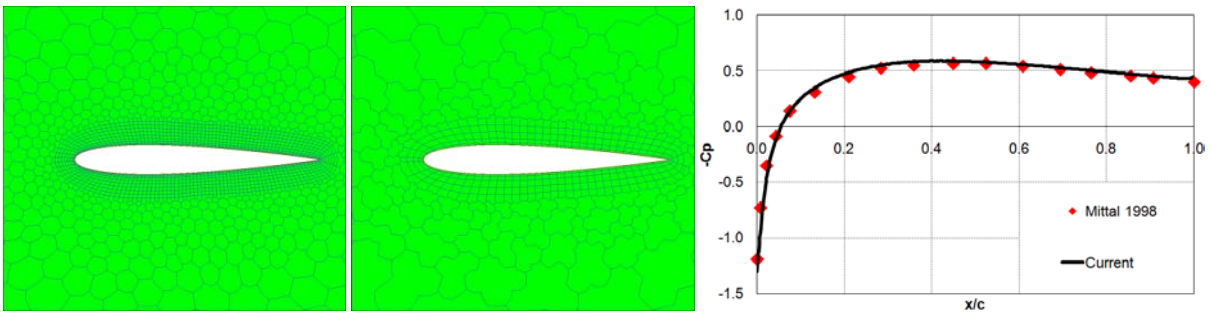

Figure 6: Symmetry surface of the initial and first level agglomerated volume grid; *Cp* distribution at mid-span

Two simulations were performed, without any difference in their final results; the first utilized (for the velocity and temperature gradients' computation) the element-based method,

while the second one the nodal-averaging scheme. In Figure 7 the convergence history for density (per number of iterations and time) is presented for the single-grid simulation as well as for the multigrid one, using the element-based method; a time speed-up coefficient equal to ~10.5 was achieved for a final residual 1.0E-10. The corresponding convergence history for the nodal-averaging scheme is illustrated in Figure 8; a temporal speed-up coefficient equal to ~7.0 was succeeded along with a considerably reduced computation time (~10.0 hours).

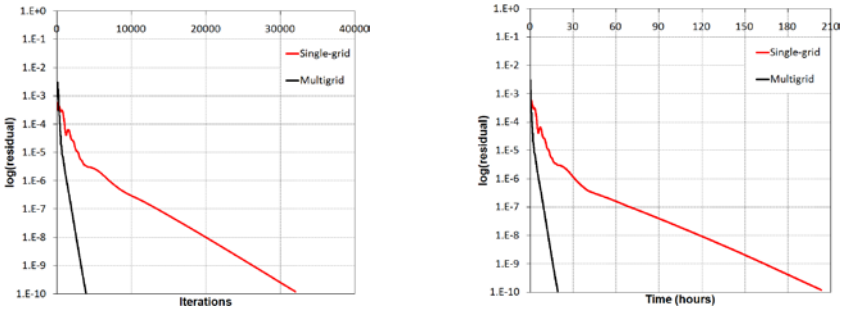

Figure 7: Convergence history (density) per number of iterations and time for the element-based method

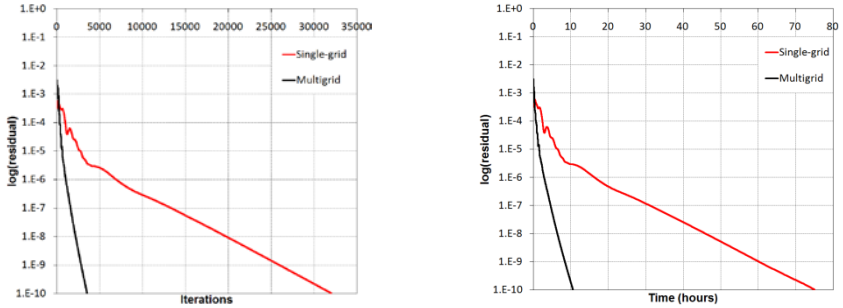

Figure 8: Convergence history (density) per number of iterations and time for the nodal-averaging scheme

4.3 Turbulent flow over the DLR-F6 aircraft

The last test case concerns the simulation of the steady-state fully turbulent flow (Mach number *0.75*, Reynolds number *3E+6* and angle of attack *0.275*^{o}) over the DLR-F6 configuration without engine nacelles (wing, body - WB configuration), which was examined in the second AIAA Drag Prediction Workshop (DPW) in Orlando, in June 2003[17,18]. A relatively coarse mesh, provided by NASA Langley Research Center (LARC) site, was used for the evaluation of the proposed multigrid methodology; it consists of *622,445* nodes, *1,217,387* tetrahedrons, *790,934* prisms and *78* pyramids, while for parallel computation on a workstation with an AMD FX$^{(tm)}$-8350 eight-core processor at *4.0* GHz it was divided in four sub-domains. The four-stage Runge-Kutta method was utilized with a CFL number equal to *0.5*, while for turbulence prediction the SST model, without any transition scheme, was employed along with an implicit source term treatment. For the multigrid accelerated solution, three coarser grids were generated via the directional agglomeration method; in Figure 9 the symmetry surfaces of the initial, first and second level agglomerated volume grid are illustrated. In addition, Figure 10 presents the convergence history for density per number of iterations and time (upper) as well as the corresponding ones for the turbulent kinetic energy (lower); a time speed-up coefficient equal to ~4.2 was achieved for the final residual 1.0E-8.

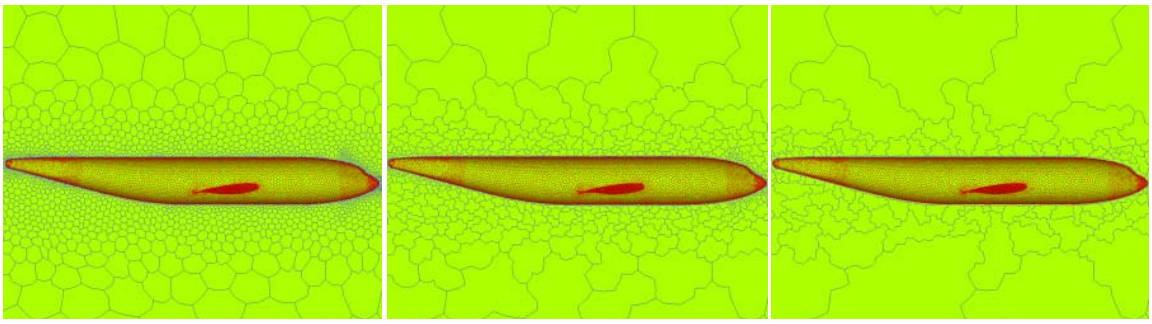

Figure 9: Symmetry surface of the initial (non-agglomerated), first and second level agglomerated volume grid

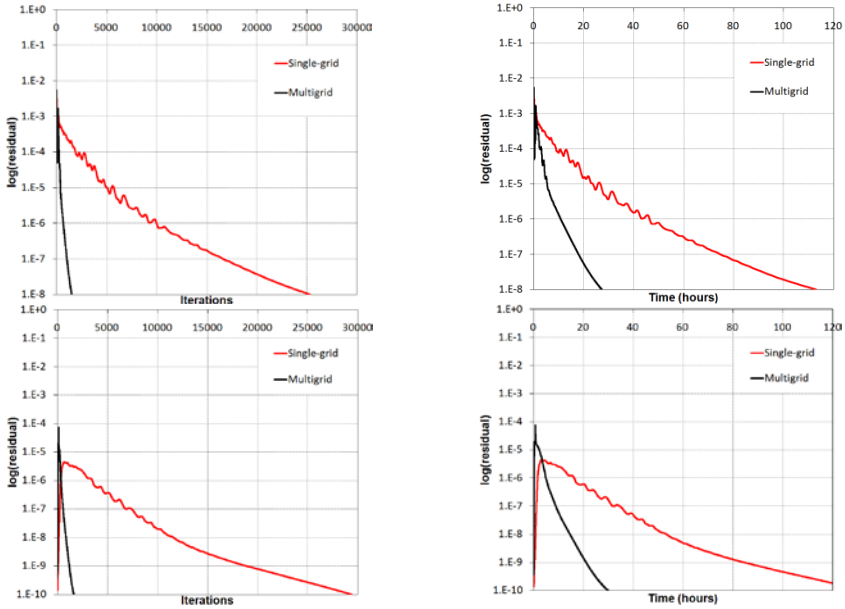

Figure 10: Density (upper) and turbulent kinetic energy (lower) convergence histories per iterations and time

As the utilized grid was coarse to accurately predict the flow over the aircraft, a finer mesh was also used, composed of *5,666,335* nodes, *4,175,553* tetrahedrons, *9,768,149* prisms and *2,587* pyramids. In Figure 11, the obtained distributions of pressure coefficient at the span-wise sections *33.1%* and *84.7%* are provided, compared to these of a reference solver[17].

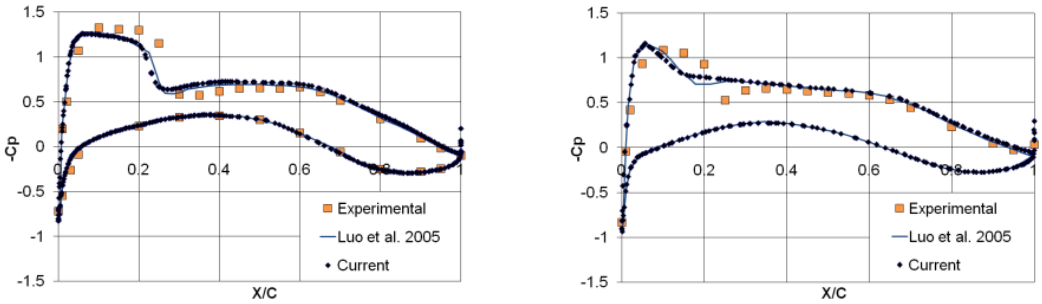

Figure 11: Distribution of *Cp* at the span-wise sections *33.1%* (left) and *84.7%* (right)

REFERENCES

[1] Ferziger, J.H. and Peric, M. *Computational methods for fluid dynamics*. Springer, 3rd Edition, (2002).

[2] Lygidakis, G.N. and Nikolos, I.K. Using a high-order spatial/temporal scheme and grid adaptation with a finite-volume method for radiative heat transfer. *Num. Heat Transfer Part B: Fundamentals* (2013) **64**:89-117.

[3] Blazek, J. *Computational fluid dynamics: principles and applications*. Kidlington, Elsevier Science, (2001).

[4] Carre, G., Fournier, L. and Lanteri, S. Parallel linear multigrid algorithms for the acceleration of compressible flow calculations, *Comput. Methods Appl. Mech. Engrg.* (2000) **184**:427-448.

[5] Carre, G. and Lanteri, S. Parallel linear multigrid by agglomeration for the acceleration of 3D compressible flow calculations on unstructured meshes, *Numerical Algorithms* (2000) **24**: 309-332.

[6] Nishikawa, H. and Diskin, B. Development and application of parallel agglomerated multigrid methods for complex geometries (AIAA 2011-3232), *20th AIAA Computational Fluid Dynamics Conference*, Honolulu, Hawaii, 27 - 30 June, (2011).

[7] Mavriplis, D.J. and Pirzadeh S. Large-Scale Parallel unstructured mesh computations for 3D high-lift analysis, *NASA ICASE Report*, No. 99-9, (1999).

[8] Brandt, A. *Multigrid techniques with applications to fluid dynamics: 1984 guide*, VKI Lecture Series, (1984).

[9] Mavriplis, D.J. Directional coarsening and smoothing for anisotropic Navier-Stokes problems, *Electronic Transactions on Numerical Analysis* (1997) **6**:182-197.

[10] Lygidakis, G.N. and Nikolos, I.K. Evaluating a parallel node-centered finite-volume algorithm, named Galatea, in simulation of 3D compressible flows, *Proc. 10th HSTAM International Congress on Mechanics*, Chania, Greece, 25 - 27 May, (2013).

[11] Sorensen, K.A. et al. A mutigrid accelerated hybrid unstructured mesh method for 3D compressible turbulent flow, Computational Mechanics (2003) **31**:101-114.

[12] Roe, P. Approximate Riemann solvers, parameter vectors and difference schemes, Journal of Computational Physics (1981) **43**:357-371.

[13] Lygidakis, G.N. and Nikolos, I.K. Using the finite-volume method and hybrid unstructured grids to compute radiative heat transfer in 3-D geometries. *Num. Heat Transfer Part B: Fundamentals* (2012) **62**:289-314.

[14] Lambropoulos, N.K. et al. Acceleration of a Navier-Stokes solver for unstructured grids using agglomeration multigrid and parallel processing, *Comput. Methods Appl. Mech. Engrg.* (2004) **193**:781-803.

[15] Furmanek, P. Numerical solution of steady and unstedy compressible flow, PhD Thesis, Czech Technical University in Prague, (2008).

[16] Mittal, S. Finite element computation of unsteady viscous compressible flows, *Comput. Methods Appl. Mech. Engrg.* (1998) **157**:151-175.

[17] Luo, H. et al. High-Reynolds number viscous flow computations using an unstructured-grid method. *Journal of Aircraft* (2005) **42**:483-492.

[18] Langtry, R.B. et al. Drag prediction of engine-airframe interference effects with CFX-5. *Journal of Aircraft* (2005) **42**:1523-1529.