INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2002; **40**:1551–1560 (DOI: 10.1002/fld.411)

An algebraic 2-level domain decomposition preconditioner with applications to the compressible Euler equations

Marzio Sala[*,†]

Département de Mathématiques, EPF-Lausanne, CH-1015 Lausanne, Switzerland

SUMMARY

A domain decomposition technique to solve large-scale aerodynamics problems on unstructured grids is investigated. The linear system, arising when an implicit time-advancing scheme is used, is preconditioned using a Schwarz-based method. The key idea of the Schwarz preconditioner is to solve (approximately) a linear problem on each subdomain, then to exchange information only with neighbouring subdomains. Unfortunately, the performance of the Schwarz-type preconditioner deteriorates as the number of processors grows. So, in this case, a key element for obtaining a scalable preconditioner is to provide a *coarse level* operator. Since many of the coarse operators proposed in literature are difficult to implement on unstructured 2D and 3D meshes, a purely algebraic procedure, that requires the entries of the matrix only, has been developed. This procedure may be seen as an Algebraic MultiGrid (AMG) method applied as a coarse grid correction operator. The key idea is to take advantage of the use of local data of domain decomposition preconditioners, and of the automatic coarsening procedures of AMG methods.

Two possible schemes to introduce the coarse grid operator will be described. Both cases have been implemented and tested in a distributed parallel environment, using the MPI library. It will be shown that for suitable values of the rank of the coarse grid operator it is possible to obtain a considerable reduction in the number of iterations compared to the Schwarz preconditioner without coarse operator. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: domain decomposition; Schwarz-type preconditioner; compressible Euler equations

1. INTRODUCTION

Let us consider the solution of the following linear system

$$A\mathbf{u} = \mathbf{f} \tag{1}$$

[*] Correspondence to: M. Sala, Département de Mathématiques, EPF-Lausanne, CH-1015 Lausanne, Switzerland.
[†] E-mail: Marzio.Sala@epfl.ch

Contract/grant sponsor: OFES; contract/grant number: BRPR-CT97-0591.

Received 1 November 2001
Revised 12 June 2002

where $A \in \mathbb{R}^{n\times n}$ is a (possibly) non-symmetric large and sparse real matrix, which is in general ill-conditioned. Consequently, standard Krylov subspace iterative methods converge slowly (or may even diverge if not properly preconditioned).

Although it is difficult to find a general purpose preconditioner, there is an interest in developing a class of preconditioners that is suitable for a large group of applications. In particular, since parallel computers are currently widespread, there is an interest in devising parallel preconditioners, which in this context should also guarantee, as far as possible, the scalability of the iterative solver with respect to the number of processors employed.

In a parallel setting the perhaps simplest preconditioner is obtained using a block-Jacobi procedure, where each block is allocated to a processor and is possibly approximated by an incomplete factorization [1]. This approach may work well for simple problems, yet its performance degrades rapidly as the size of the matrix increases, and it has a poor scalability [2]. A possible improvement consists in enriching the factorization, to use a multi-level method [3], to use approximate inverses or to resort to AMG methods [4–6].

In this work an algebraic preconditioner based on the Schwarz method has been developed. It can be used as a black-box tool which requires the matrix entries and the topology of the domain partition and it can in principle be applied to any non-singular matrix. The performances of the proposed preconditioners are reported for a simple test case, as well as for problems arising from compressible Euler equations in three dimensions. In the latter case the equations have been discretized using a second-order multidimensional upwind technique, while pseudo-time stepping is performed by an implicit Euler scheme.

The paper is organized as follows. Section 2 gives a brief description of the Schwarz method and Section 3 introduces the coarse operator based on an agglomeration procedure. Numerical results are given in Section 4, where some insight on the discretization scheme adopted for the solution of the fluid dynamics problem is provided. Conclusions are drawn in Section 5.

2. THE SCHWARZ PRECONDITIONER

The Schwarz method is a well-known parallel technique based on a domain decomposition strategy. It is in general a rather inefficient solver, however it is a quite popular parallel preconditioner. Its popularity derives from its generality and simplicity of implementation. The procedure is as follows. Let us decompose the computational domain Ω into M subdomains $\Omega^{(i)}$, $i = 1,\dots,M$, such that $\bigcup_{i=1}^{M}\Omega^{(i)} = \Omega$ and $\Omega^{(i)}\cap\Omega^{(j)} = \emptyset$ for $i \neq j$. To introduce a region of overlap, these subdomains are extended to $\tilde{\Omega}^{(i)}$ by adding to each $\Omega^{(i)}$ all the elements of Ω that have at least one node in $\Omega^{(i)}$. In this case, the overlap is minimal. More overlap can be obtained by recursively repeating this procedure. A parallel solution of the original system is then given by an iterative procedure involving local problems in each $\tilde{\Omega}^{(i)}$, where on $\partial\tilde{\Omega}^{(i)}\backslash\Omega$ Dirichlet conditions are applied by imposing the latest values available from the neighbouring subdomains. The increase of the amount of the overlap among subdomains has a positive effect on the convergence of the iterative procedure, at the price of a more computationally expensive method. Furthermore, the minimal overlap variant may exploit the same data structure used for the parallel matrix–vector product in the outer iterative solver, thus allowing a very efficient implementation with respect to memory requirements (this is usually not anymore true for wider overlaps). In the numerical results, presented later in this paper, the minimal overlap strategy has been used, that is, an overlap of width of one element only. See References [7, 8] for more details.

3. THE AGGLOMERATION COARSE OPERATOR

The use of an agglomeration procedure to build the coarse operator for a Schwarz preconditioner has been investigated in Reference [9] for elliptic problems. Here, this technique is extended and generalized and applied it to non-self-adjoint problems.

For the sake of simplicity, let us consider the discrete variational problem:

$$\text{find } u_h \in V_h \text{ such that:}$$

$$a(u_h, v_h) = (f, v_h) \quad \text{for } \forall v_h \in V_h$$

where $u_h, v_h, f : \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, $a(\cdot,\cdot)$ is a bi-linear form and V_h is a finite-dimensional Hilbert space of (possibly vector) functions in Ω. (u, v) denotes the L_2 scalar product, i.e. $(u, v) = \int_\Omega uv \, d\Omega$. In the following V_h will be the space of finite element functions defined on a triangulation of Ω and $\phi_k^{(i)}$ is the basis function associated to the kth mesh node of $\Omega^{(i)}$. $n^{(i)}$ is the number of basis function associated to $\Omega^{(i)}$. V_h is a finite-dimensional space, equipped with piecewise linear basis functions.

A *coarse space* is built as follows. Let us consider for each sub-domain the set $\{\boldsymbol{\beta}_s^{(i)} \in \mathbb{R}^{n^{(i)}}, s = 1, \ldots, l^{(i)}\}$ of linearly independent nodal weights $\boldsymbol{\beta}_s^{(i)} = (\beta_{s,1}^{(i)}, \ldots, \beta_{s,n^{(i)}}^{(i)})$.

The value $l^{(i)}$ represents the (local) dimension of the coarse operator on the sub-domain $\Omega^{(i)}$, while M is the number of subdomains. Clearly it is required that, for all $i, l^{(i)} \leqslant n^{(i)}$ and, in general, $l^{(i)} \ll n^{(i)}$. l indicates with the global dimension of the coarse space, $l = \sum_{i=1}^M l^{(i)}$.

With the help of the vectors $\boldsymbol{\beta}_s^{(i)}$, it is possible to define a set of local coarse space functions as linear combination of basis functions of V_h, i.e.

$$\mathscr{V}_H^{(i)} = \left\{ z_s^{(i)} : \Omega \to \mathbb{R} \mid z_s^{(i)} = \sum_{k=1}^{n^{(i)}} \beta_{s,k}^{(i)} \phi_k^{(i)}, \; s = 1, \ldots, l^{(i)} \right\}$$

It is immediate to verify that the functions in $\mathscr{V}_H^{(i)}$ are linearly independent. Finally, the set $\mathscr{V}_H = \bigcup_{i=1}^M \mathscr{V}_H^{(i)}$ is the base of the global coarse grid space V_H, i.e. $V_H = \text{span}\{\mathscr{V}_H\}$.

Note that $V_H \subset V_h$ as it is built by linear combinations of function in V_h. Any function $W_H \in V_H$ may be written as

$$W_H = \sum_{i=1}^{M} \sum_{s=1}^{l^{(i)}} W_s^{(i)} z_s^{(i)} \tag{2}$$

where the $W_s^{(i)}$ are the 'coarse' degrees of freedom.

Finally, the coarse problem is built as

$$\text{find } U_H \in V_H \text{ such that}$$

$$a(U_H, W_H) = f(W_H) \quad \forall W_H \in V_H \tag{3}$$

The interpolation operator $R_H^{\mathrm{T}}: V_H \to V_h$ is defined as a rectangular $n \times l$ matrix whose columns correspond to the nodal weight $\boldsymbol{\beta}_s^{(i)}$, i.e.

$$R_H^{\mathrm{T}} = [\boldsymbol{\beta}_1^{(1)} \dots \boldsymbol{\beta}_{l^{(1)}}^{(1)} \dots \boldsymbol{\beta}_{l^{(M)}}^{(M)}] \tag{4}$$

where the $\boldsymbol{\beta}_s^{(i)}$ are column vectors. The restriction operator will simply be the transpose of R_H^{T}. It is clear from (4) that R_H strongly depends on the choice of the nodal weights.

Furthermore, the coarse matrix and the right-hand side of problem (3) can be written as $A_H = R_H A R_H^{\mathrm{T}}$ and $\mathbf{f}_H = R_H \mathbf{f}$.

The condition imposed on the $\boldsymbol{\beta}_s^{(i)}$ guarantees that the columns of R_H are linearly independent. Thus, if A is non-singular, symmetric and positive definite, then also A_H is non-singular, symmetric and positive definite.

For the numerical results later presented, the construction of the $\boldsymbol{\beta}_s^{(i)}$ has been obtained using the following procedure. Each sub-domain $\tilde{\Omega}_i$ has been partitioned into N_{parts} connected parts, indicated in the following as $\omega_s^{(i)}$ with $s = 1, \dots, N_{\text{parts}}$. The coarse matrix is built by taking for all sub-domains $l^{(i)} = N_{\text{parts}}$, while the elements of $\boldsymbol{\beta}_s^{(i)}$ are defined following the rule

$$\beta_{s,k}^{(i)} = \begin{cases} 1 & \text{if node } k \text{ belongs to } \omega_s^{(i)} \\ 0 & \text{otherwise.} \end{cases}$$

As already explained, the coarse grid operator is used to ameliorate the scalability of a Schwarz-type parallel preconditioner P_{S}. By notation, P_{ACM} indicates a preconditioner augmented by the application of the coarse operator (ACM stands for agglomeration coarse matrix). In particular, two possible strategies for its construction are here illustrated.

A one-step preconditioner, $P_{\mathrm{ACM},1}$, may be formally written as

$$P_{\mathrm{ACM},1}^{-1} = P_{\mathrm{S}}^{-1} + A_{\mathrm{ACM}}^{-1}$$

where $A_{\mathrm{ACM}}^{-1} = R_H^{\mathrm{T}} A_H^{-1} R_H$. This preconditioner corresponds to an additive application of the coarse operator.

An alternative formulation adopts the following two-step Richardson method:

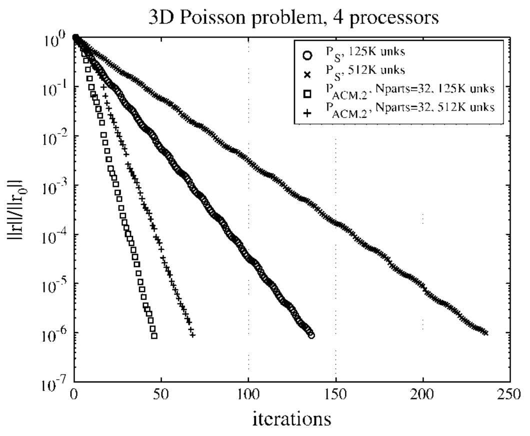
$$\begin{aligned} \mathbf{u}^{n+1/2} &= \mathbf{u}^n + A_{\mathrm{ACM}}^{-1} \mathbf{r}^n \\ \mathbf{u}^{n+1} &= \mathbf{u}^{n+1/2} + P_{\mathrm{S}}^{-1} \mathbf{r}^{n+1/2} \end{aligned} \tag{5}$$

where $\mathbf{r}^n$ and $\mathbf{u}^n$ are, respectively, the residual and the approximate solution at the n-step of the outer iterative solver. The corresponding preconditioning matrix can be formally written as

$$P_{\mathrm{ACM},2}^{-1} = P_{\mathrm{S}}^{-1} + A_{\mathrm{ACM}}^{-1} - P_{\mathrm{S}}^{-1} A \, A_{\mathrm{ACM}}^{-1} \tag{6}$$

4. NUMERICAL RESULTS

The numerical tests have been conducted on a SGI Origin 3000 computer, equipped with 32 MIPS R14000 processors with 500 Mhz and 512 Mbytes of memory. The communicator is MPI. For the solution of the linear system the AZTEC library [10] has been used. This



N_{parts}	N_{procs} 1	2	4
P_S	25.72	18.82	10.46
2	33.17	22.64	11.88
4	31.30	22.15	9.99
8	31.32	22.14	8.10
16	32.30	19.54	7.53
32	32.77	17.33	7.48
64	26.61	15.31	6.36
128	23.64	13.70	5.15
256	20.17	11.31	4.72

Figure 1. 3D Poisson problem. Comparison among P_S and $P_{ACM,2}$. On the left, convergence history using 4 processors and 125.000 and 512.000 unknowns. On the right, CPU-time in seconds using P_S and $P_{ACM,2}$ with increasing dimension of the coarse matrix. P_S with 1 processor means that the preconditioner is an ILU(0) decomposition of matrix A.

library has been extended to include the preconditioners previously described; these extensions are freely available contacting the author. As regards the partitioning software, the package METIS [11] have been used.

For the Schwarz preconditioner described in Section 2, and always indicated with P_S, an overlap of 1 element among the sub-domains have been used. The local problems are solved inexactly using an ILU(0) decomposition.

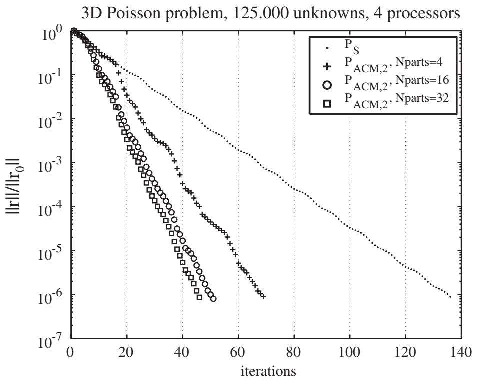
4.1. 3D Poisson problem

Consider the following Poisson problem:

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned} \tag{7}$$

where $\Omega = (0,1) \times (0,1) \times (0,1)$. The finite element discretization makes use of piece-wise constant function on a regular mesh. The linear system has been solved using GMRES(60), up to a tolerance on the relative residual $\|\boldsymbol{r}\|/\|\boldsymbol{r}_0\|$ of 10^{-6}. The right-hand side is made up of 1's.

The picture on the left of Figure 1 compares the performances of P_S and $P_{ACM,2}$, keeping fixed the number of processors. As one may notice, the convergence is significantly faster using the algebraic coarse correction than only using P_S. However, the CPU-time needed to solve the linear system may increase, as reported in the table on the right side of Figure 1 for different values of N_{parts}. 1, 2 and 4 processors have been used in the computations. One may note that the coarse correction may lead to better CPU-time even in a serial implementation (that is, added to an incomplete factorization of the whole matrix). Figure 2 shows the influence of the coarse matrix size $l = N_{parts} \times M$ on the converge history. The total number of unknowns is kept fixed and it is equal to 125.000 (for the picture on the left) and to 512.000 (for the picture on the right). As expected, increasing the value of l leads to a faster convergence of the iterative method.



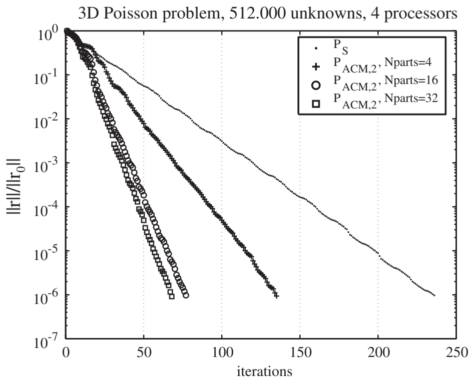


Figure 2. 3D Poisson problem with 125.000 unknowns (on the left) and 512.000 unknowns (on the right). Comparison between P_S and $P_{ACM,2}$ for some values of N-parts and using 4 processors.

4.2. The compressible Euler equations

Before presenting the numerical results concerning the Euler equations, a brief insight will be given about the application problem considered in this paper, namely inviscid compressible flow around aeronautical configurations, and the numerical scheme adopted.

The Euler equations govern the dynamics of compressible inviscid flows and can be written in conservation form as

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{d} \frac{\partial \mathbf{F}_j}{\partial x_j} = 0 \quad \text{in } \Omega \subset \mathbb{R}^d, \ t > 0 \tag{8}$$

with the addition of suitable boundary conditions on $\partial\Omega$ and initial conditions at $t = 0$. Here, $\mathbf{U}$ and $\mathbf{F}_j$ are the vector of conservative variables and the flux vector, respectively defined as

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u_i \\ \rho E \end{pmatrix}, \quad \mathbf{F}_j = \begin{pmatrix} \rho u_j \\ \rho u_i u_j + p\delta_{ij} \\ \rho H u_j \end{pmatrix}$$

with $i = 1, \dots, d$, $d = 2, 3$. $\mathbf{u} = (u_1, \dots, u_d)$ is the velocity vector, ρ the density, p the pressure, E the specific total energy, H the specific total enthalpy and δ_{ij} the Kronecker symbol.

Any standard spatial discretization applied to the Euler equations leads eventually to a system of ODE in time, which may be written as $\mathrm{d}U/\mathrm{d}t = R(U)$, where $U = (U_1, U_2, \dots, U_n)^{\mathrm{T}}$ is the vector of unknowns with $U_i = U_i(t)$ and $R(U)$ the result of the spatial discretization of the Euler fluxes. An implicit two-step scheme applied to (8), for instance a backward Euler method, yields

$$U^{n+1} - U^n = \Delta t R(U^{n+1}) \tag{9}$$

where Δt is in general the time step but may also be a *diagonal matrix* of local time steps when the well-known ‘local time stepping’ technique is used to accelerate convergence to steady-state. The non-linear problem (9) may be solved, for instance, by employing a Newton

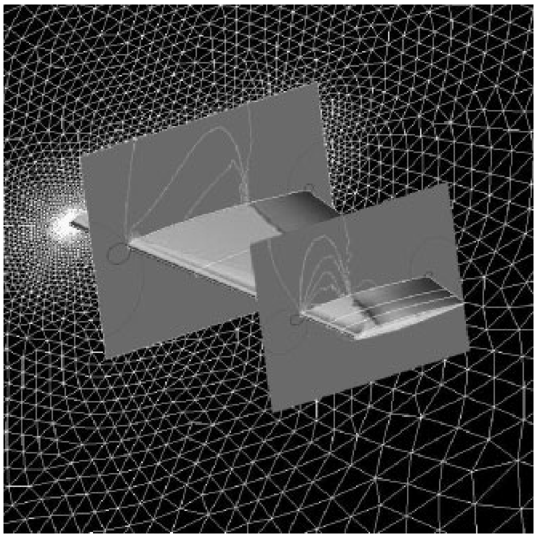


Figure 3. Mach number contours around a M6 wing. The computation has been performed using a grid composed by 316 275 nodes. The free-stream Mach number is 0.84 and the angle of attack is 3.06. A lambda shock is formed near the tip of the wing.

iterative procedure, which computes successive approximations $U_{(k)}$ of U^{n+1} by solving

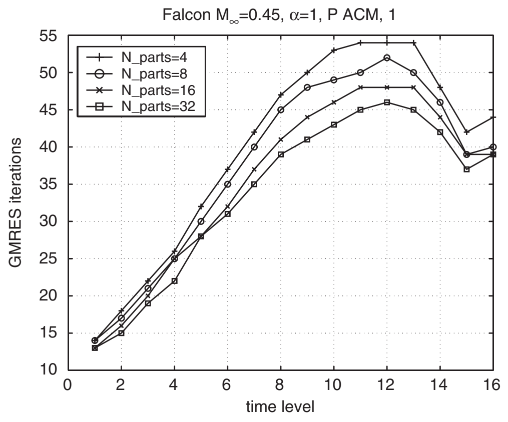
$$\left[\Delta t \frac{\partial R}{\partial U}(U_{(k)}) - I\right](U_{(k+1)} - U_{(k)}) = U_{(k)} - U^n - \Delta t R(U_{(k)}), \quad k = 0, \dots \tag{10}$$

with $U_{(0)} = U^n$. Thus, the original non-linear problem gives rise to a series of linear systems which will be finally tackled by the proposed parallel techniques. In this work, being interested in the steady state solutions only, just a single iteration of the Newton procedure (10) is performed.

The code THOR was used for the numerical experiments. This code, developed at the von Karman Institute (Belgium), uses for the spatial discretization the multidimensional upwind finite element scheme [12].

The following test cases have been considered: FALCON_45k, with the free-stream Mach number $M_\infty = 0.45$ and an angle of attach $\alpha = 1$, on a grid made up of 45 387 nodes, M6_23k, M6_42k, M6_94k and M6_316k with $M_\infty = 0.84$ and $\alpha = 3.06$, with grids of respectively, 23 008, 42 305, 94 493 and 316 275 nodes. FALCON_45k concerns the flow field around a Falcon aircraft. Although the mesh is rather coarse, this test case is interesting since it corresponds to a complete configuration of an airplane, with fuselage, wings, nacells and aleirons. The resulting flow field is subsonic. The ONERA-M6 wing is a classic CFD validation case for external flows because of its simple geometry combined with complexities of transonic flow [13]. For the prescribed boundary conditions, the flow around the ONERA-M6 wing is transonic, and a lambda shock is formed near the tip of the wing. Mach number countours are depicted in Figure 3.

For all of them, the starting solution is the constant-field, and the CFL number varies from 10 to 10^{+5}, multiplied each time step by a factor 2. A first-order scheme for the spatial



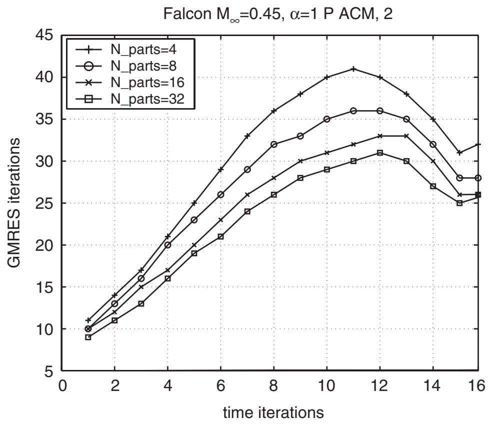
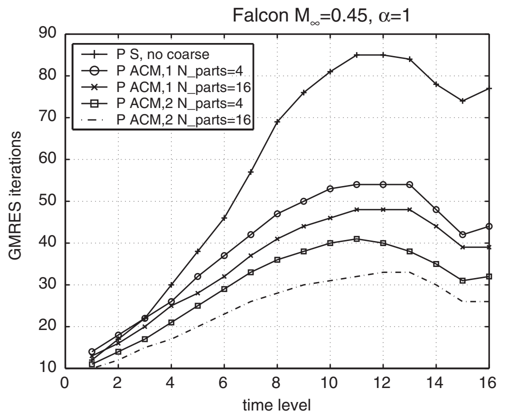


Figure 4. FALCON_45k. Iterations to converge with different values of N-parts for $P_{\text{ACM},1}$ (left) and $P_{\text{ACM},2}$ (right). Sixteen processors have been used in the computations.



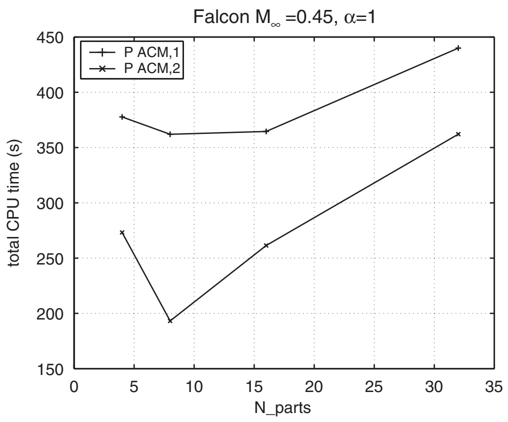
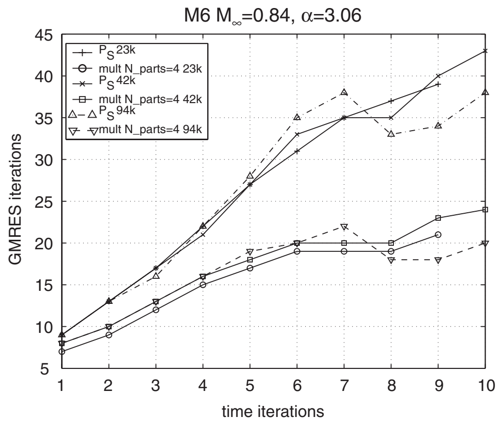


Figure 5. FALCON_45k. Iterations to converge (left) and CPU-time (right) using different preconditioners and different values of N-parts. Sixteen processors have been used in the computations.

discretization has been used. The computations are stopped when the 2-norm of the density residual is less than 10^{-6}. As previously described, at each time step a linear system has to be solved. This is done using GMRES(60), up to a tolerance on the relative residual $\|\boldsymbol{r}\|/\|\boldsymbol{r}_0\|$ of 10^{-6}. The starting solution is the zero vector.

In Figure 4 the influence of the local dimension of the coarse grid N_{parts} is reported for $P_{\text{ACM},1}$ and $P_{\text{ACM},2}$, using 16 processors. The value of N_{parts} does not affect remarkably the convergence of GMRES, unless the CFL number is large enough (as previously stated, the CFL number is multiplied by 2.0 at each time level). The two-level preconditioner seems a better choice from the point of view of both iterations to converge and CPU time, especially for small values of N_{parts}, as one may notice in Figure 5.



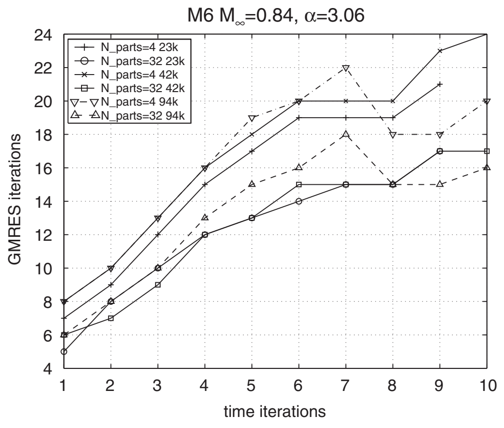


Figure 6. M6_94k. Iterations to converge with P_S and $P_{ACM,2}$ (left), and iterations to converge with $P_{ACM,2}$ using two different values of N_{parts} (right). In the computations 16 processors have been used. 23 k, 42 k and 94 k refer to the test cases M6_23k, M6_42k and M6_94k, respectively.

Finally, from Figure 6, one can note that, for the two-level preconditioner, the positive influence of the agglomeration procedure on the number of iterations to converge is not affected by the mesh size.

5. CONCLUSIONS

A coarse correction operator based on an agglomeration procedure that requires the matrix entries only has been presented. This procedure does not require the construction of a coarse grid, a step that can be difficult or computationally expensive for real-life problems on unstructured grids. A single-level and a two-level preconditioner which adopt this coarse correction have been presented. The latter seems to be a better choice from the point of view of both iterations to converge and CPU time. Results have been presented for problems obtained from the three-Dimensional compressible Euler equations where the procedure has shown a good performance. The proposed coarse operator is rather easy to build and may be applied to very general cases, and it may be generalized by employing different strategies for building the weights $\boldsymbol{\beta}$.

ACKNOWLEDGEMENTS

The author acknowledges the support of the OFES under contract number BRPR-CT97-0591.

REFERENCES

1. Saad Y. *Iterative Methods for Sparse Linear Systems*. Thompson: Boston, 1996.
2. Saad Y, Zhang Y. BILUTM: a domain-based multi-level block ILUT preconditioner for general sparse matrices. *Technical Report UMSI* 98/118, Minnesota Supercomputing Institute, University of Minnesota, Minneapolis, MN, 1998.

3. Zhang J. A grid based multilevel incomplete LU factorisation preconditioning technique for general sparse matrices. *Technical Report* No. 283-99, Department of Computer Science, University of Kentucky, Lexington, KY, 1999.
4. Vanek P, Brezina M, Mandel J. Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*. 2001; **88**:559–579.
5. Carré G, 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. Mavriplis D, Venkatakrishnan V. A 3D agglomeration multigrid solver for the Reynolds-averaged-Navier–Stokes equations on unstructured meshes. *International Journal for Numerical Methods in Fluids* 1996; **23**:527–544.
7. Quarteroni A, Valli A. *Domain Decomposition Methods for Partial Differential Equations*. Oxford University Press: Oxford, 1999.
8. Smith BF, Bjorstad P, Gropp W. *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press: New York, 1996.
9. L. Formaggia, A. Scheinine, A. Quarteroni. A numerical investigation of Schwarz domain decomposition techniques for elliptic problems on unstructured grids. *Mathematics and Computer in Simulations* 1997; **44**:313 –330.
10. Hutchinson SA, Shadid JN, Tuminaro RS. AZTEC user's guide. *Technical Report SAND95-1559*, Sandia National Laboratories, Albuquerque, NM, 1995.
11. Karypis G, Kumar V. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM Journal on Scientific Computing* 1998; **20**:359–392.
12. Bastin J, Roge G. A multidimensional fluctuation splitting scheme for the three dimensional Euler equations. *Mathematical Modelling and Numerical Analysis* 1999; **33**:1241–1259.
13. Schmitt V, Charpin F. Pressure distributions on the ONERA-M6-Wing at transonic mach numbers. Experimental data base for computer program assessment. *Report of the Fluid Dynamics Panel Working Group* 04, AGARD AR 138, May 1979.