Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Application of parallel algebraic multilevel domain decomposition preconditioners in large eddy simulations of wall-bounded turbulent flows: first experiments

Andrea Aprovitola, Pasqua D'Ambra,
Filippo Maria Denaro, Daniela di Serafino,
Salvatore Filippone

RT-ICAR-NA-07-02 **07-2007**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) – Sede di Napoli, Via P. Castellino 111, I-80131 Napoli, Tel: +39-0816139508, Fax: +39-0816139531, e-mail: napoli@icar.cnr.it, URL: *www.na.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Application of parallel algebraic multilevel domain decomposition preconditioners in large eddy simulations of wall-bounded turbulent flows: first experiments

Andrea Aprovitola[2], Pasqua D'Ambra[1],
Filippo Maria Denaro[2], Daniela di Serafino[3],
Salvatore Filippone[4]

Rapporto Tecnico N.: ***Data:***
RT-ICAR-NA-07-02 **07-2007**

---

[1] Institute for High-Performance Computing and Networking (ICAR), CNR,Via P. Castellino 111, 80131 Naples, Italy
[2] Department of Aerospace and Mechanical Engineering, Second University of Naples, Via Roma 29, Aversa, Italy
[3] Department of Mathematics, Second University of Naples,Via Vivaldi 43, I-81100 Caserta, Italy
[4] Department of Mechanical Engineering, University of Rome "Tor Vergata",Viale del Politecnico 1, 00133, Rome, Italy

Application of parallel algebraic multilevel domain decomposition preconditioners in large eddy simulations of wall-bounded turbulent flows: first experiments

Andrea Aprovitola [a,1], Pasqua D'Ambra [b], Filippo Maria Denaro [a], Daniela di Serafino [c], Salvatore Filippone [d]

[a] *Department of Aerospace and Mechanical Engineering, Second University of Naples, Via Roma 29, I- Aversa, Italy*

[b] *Institute for High-Performance Computing and Networking, CNR, Via Pietro Castellino 111, I-80131 Naples, Italy*

[c] *Department of Mathematics, Second University of Naples, Via Vivaldi 43, I-81100 Caserta, Italy*

[d] *Department of Mechanical Engineering, University of Rome "Tor Vergata", Viale del Politecnico 1, I-00133, Rome, Italy*

Abstract

In this work we analyze the behaviour of multilevel Domain Decomposition preconditioners, coupled with Krylov solvers, in the solution of linear systems arising in the application of a projection-based method to the numerical simulation of uncompressible wall-bounded turbulent flows. We consider a Large Eddy Simulation (LES) approach, where a high-order filter is applied to the Navier-Stokes equations. In this context, one of the main computational tasks is the solution of an elliptic equation, leading to large and sparse linear systems to be solved at each time step of a simulation. We show the effectivenes of the algebraic multilevel preconditioners implemented in the MLD2P4 software package, in terms of efficiency and scalability, for linear systems arising from a bi-periodical channel flow simulation. This work is the first step towards the development of a scalable parallel code for the LES of wall-bounded flows in real-world applications.

Key words: Parallel Sparse Linear Solvers, Algebraic Multilevel Preconditioners, Computational Fluid Dynamics, Large Eddy Simulation

July 2007

1 Introduction

The hypothesis of fluid incompressibility is traditionally adopted for solving problems in which acoustic waves propagate at a velocity much greater than that of macroscopic advective transport of properties, which is the case of low Mach number flows. This hypothesis remains essentially valid also for flows governed by weak temperature gradients, such as buoyancy-driven flows. Hence, even if the hypothesis of incompressibility leads to simplifying the Navier-Stokes (N-S) equations, the mass-conservation law is reduced to the constraint $\underline{\nabla} \cdot \mathbf{v} = 0$ and the pressure only acts as a Lagrangian multiplier, with no thermodynamic law driving the evolution. This complicates the task of solving the discrete formulation of the governing equations [18]. This form of the N-S equations was historically solved by means of classical methods, such as artificial compressibility, pressure-correction methods or stream function-vorticity formulations (see [16] for a review).

To reduce the computational effort in the solution of incompressible N-S equations, a numerical formulation decoupling the velocity and the pressure gradient, based on the Helmholtz-Hodge decomposition (HHD) theorem [9,13,20], leads to the class of the so-called *projection methods*. A fundamental task of the projection methods is to obtain a divergence-free velocity field in a discrete sense, and the main computational effort is the solution of a discrete elliptic operator which substitutes the continuity equation. This operator allows to project a predicted velocity in the sub-space of divergence-free vector fields. Projection methods may be used for both laminar and turbulent flow simulations, but they are most commonly employed in simulating turbulence, where the computational complexity reaches is highest and requires very high performance solvers.

The study of the incompressible turbulence finds its application in a wide range of scientific and engineering fields, such as oceanography, meteorology, air quality, aircraft design and so on. Unfortunately, the numerical simulation of real-world incompressible turbulent flows is still an open issue due to the high computational effort required to adequately describe them.

As opposed to laminar flows, in which energy and momentum transfers are essentially due to the molecular diffusion process, turbulent flows are charac-

Email addresses: andrea.aprovitola@unina2.it (Andrea Aprovitola),
pasqua.dambra@na.icar.cnr.it (Pasqua D'Ambra),
filippomaria.denaro@unina2.it (Filippo Maria Denaro),
daniela.diserafino@unina2.it (Daniela di Serafino),
salvatore.filippone@uniroma2.it (Salvatore Filippone).
[1] This author has been supported by the ICAR Project "Numerical Simulation of Oceanographic Models", funded by Stazione Zoologica "A. Dohrn".

terized by the inertial transport of vortical structures of very different length scales; these structures produce fluctuations in the velocity field at macroscopic level, increasing the energy and momentum mixing. Thus, kinetic energy is transferred inviscidly from larger scales, where it is produced, down to smaller scales whereas it is finally dissipated. Since the energy production and dissipation mechanisms take place at different length scales, an adequate description of this behavior, entails a resolution of all fluid spatial and temporal scales [16,28], dramatically increasing the cost of a numerical simulation.

The most accurate way to simulate the dynamics of turbulent flows is to solve the N-S equations without any approximations other than those deriving from the numerical scheme. This approach, called Direct Numerical Simulation (DNS), can be considered equivalent to performing a realistic experiment [16,29,30,34]. In a DNS it is assumed that all the fluid structures, responsible of the production, transfer and dissipation of energy, are well resolved. This assumption implies that, in order to perform a DNS, the number of computational grid points has to be of the order of L/η in each coordinate direction, where L is the integral length scale corresponding to the measure of the largest characteristic eddy size, and η is the smallest dissipative length scale, also known as Kolmogorov scale. It is known [34,16,30,29] that this ratio is proportional to $Re^{3/4}$, Re being the Reynolds number of the flow. Another requirement imposed by a DNS is that the time step used in the simulation must be less than the time scale of the smallest resolved scale; therefore, since the ratio of the integral time scale to diffusive time scales is also proportional to $Re^{3/4}$, the total computational cost for performing a DNS is of the order of Re^3. As the Reynolds number of most scientific and engineering applications is in the range $[10^6, 10^9]$, the application of DNS is strongly limited by its computational cost. Moin and Kim in a review article appeared on the Scientific American Magazine in 1997 [29] estimated that for the simulation of the turbulent flow near the surface of an aircraft for 1 second of flight time with a 1 *Tflops* supercomputer several thousand years were needed.

A more affordable approach to accurate numerical simulations of turbulent flows is Large Eddy Simulation (LES). The basic idea of LES is computing only the dynamics of the flow scales which are responsible for the energy transfer, while modelling the dynamics of the scales where the energy dissipation takes place. This is achieved by applying a low-pass filtering procedure to the N-S equations, thereby decomposing the velocity field into a mean field (resolved large scales) and a fluctuation field (unresolved small scales). In order to describe the interaction between the resolved and the unresolved scales a Subgrid Scales Model (SGS) has to be considered. Despite the strong reduction of the computational cost with respect to the DNS technique, the use of LES in realistic applications is still limited by the resolution requirements and it is feasible only through high-performance algorithms and software on high-end supercomputers. Furthermore, LES is very expensive in the case of

wall-bounded simulations, where the anisotropy of the flow results into larger grids and shorter time steps [22].

The projection methods-based numerical integration of the filtered equations requires the solution of large and sparse linear systems; together with the SGS computations they account for most of the execution time of the simulation codes. Therefore, the availability of efficient and reliable linear equation solvers is a key issue for an effective application of LES [30]. The choice of such solvers should be based on a tradeoff among high-performance, scalability and accuracy criteria, since all these features have a strong impact on LES simulations. As pointed out in [19], Krylov solvers with Domain Decomposition (DD) preconditioners, usually in a multilevel framework, are often the methods of choice in large-scale CFD applications, because they fit naturally in a parallel environment and exhibit good convergence properties.

The main focus of this paper is to analyze the behavior of multilevel Domain Decomposition (DD) preconditioners, coupled with Krylov iterative solvers, in the solution of linear systems arising from the application of an LES approach in the numerical simulation of wall-bounded turbulent flows. In this study we use algebraic multilevel DD preconditioners available in MLD2P4 (Multilevel Domain Decomposition Parallel Preconditioners Package based on PSBLAS) [7,10], coupled with Krylov solvers from PSBLAS [17]. These preconditioners have been proven to be effective in the solution of large-scale linear systems arising from model problems and real applications, and have been shown to maintain good scalability properties [7,11].

As a first step in the development of a scalable parallel code for LES of wall-bounded turbulent flows in real-world applications, we analyze here the behaviour of different multilevel preconditioners. We apply them to the linear system arising at the first time step of a simulation; this is a valid approach since the system matrix and the main numerical features of the linear systems do not change during the temporal evolution.

The remaining of the paper is organized as follows: in Section 2 we briefly describe the equations for incompressible flows in the framework of a LES approach. In Section 3 we outline the numerical discretization of the equations, introducing the large and sparse linear systems to be solved at each time step of a computer simulation. In Section 4 we describe the multilevel domain decomposition preconditioners coming from a software package for high-performance solution of unstructured linear systems on parallel and distributed computers. In Section 5 we discuss preliminary results on performance behaviour of different preconditioners, coupled with non-symmetric linear solvers, in the solution of the main computational kernel of a LES for a model problem. Finally, in Section 6 we draw our conclusion and outline future work.

2 Mathematical model

The motion of an incompressible and homothermal flow in a bounded domain V is described by an initial, boundary-value problem for the set of N-S equations. According to an integral approach, the mass and momentum conservation laws, written in non-dimensional form for each Finite Volume (FV) $\Omega(\mathbf{x}) \subseteq V$, are:

$$\int_{\partial\Omega(\mathbf{x})} \mathbf{v} \cdot \mathbf{n} \; dS = 0 \tag{1}$$

$$\frac{\partial \bar{\mathbf{v}}}{\partial t} = -\frac{1}{|\Omega(\mathbf{x})|} \int_{\partial\Omega(\mathbf{x})} \mathbf{F}(\mathbf{v}) \cdot \mathbf{n} \; dS, \tag{2}$$

where $\bar{\mathbf{v}}(\mathbf{x}, t)$ defines the local volume average of the field $\mathbf{v}(\mathbf{x}, t)$ over $\Omega(\mathbf{x})$, $\mathbf{n}$ is the outward-oriented unit vector normal to the boundary $\partial\Omega(\mathbf{x})$, and $\mathbf{F}$ the flux function defined as:

$$\mathbf{F}(\mathbf{v}) = \mathbf{v}\mathbf{v} + \mathbf{I}p - \frac{2}{Re}\nabla^s \mathbf{v},$$

p being the pressure term, Re the Reynolds number and $\nabla^s \mathbf{v}$ the zero-trace symmetric part of the gradient velocity. The governing equations are associated to suitable initial and boundary conditions.

In the LES approach, the velocity field $\mathbf{v}$ is decomposed in two contributions [34]:

$$\mathbf{v}(\mathbf{x}, t) = \bar{\mathbf{v}}(\mathbf{x}, t) + \mathbf{v}'(\mathbf{x}, t),$$

where $\bar{\mathbf{v}}$ is the resolved, or large scale, field and $\mathbf{v}'$ is the non-resolved, or small scale, field. The scale separation is obtained by means of a filtering procedure which is mathematically defined by the convolution product [34]:

$$\bar{\mathbf{v}}(\mathbf{x}, t) = \int_{\mathbb{R}^3} G(\mathbf{x} - \mathbf{x}'; \Delta) \, \mathbf{v}(\mathbf{x}', t) d\mathbf{x}' \equiv G * \mathbf{v},$$

where G is the convolution kernel and Δ the filter width.

Since the volume average operator is mathematically equivalent (for uniform filter width) to the so called top-hat filter [14], the equations (1) and (2) can be considered as filtered model equations. The top-hat filter belongs to the class of smooth filters [14]; therefore it does not produce a sharp separation

between wavenumbers component of the numerical solution [34]. Furthermore, the top-hat filter is not invertible because of the reducibility of the transfer function $\widehat{G}(\mathbf{k})$. A generalized regularization procedure based on Taylor series expansion [14] can be applied in order to obtain an approximate inversion of a filter, as well as a more reliable filtered variable $\tilde{\mathbf{v}}$ [2,14,21]. This variable is defined by means of a m-th order differential deconvolution operator $A_{\mathbf{x}}^{(m)}$, applied to the volume averaged variable $\bar{\mathbf{v}}$:

$$\tilde{\mathbf{v}}(\mathbf{x},\mathrm{t}) = A_{\mathbf{x}}^{(m)}\bar{\mathbf{v}} = A_{\mathbf{x}}^{(m)}\left[\frac{1}{|\Omega(\mathbf{x})|}\int\limits_{\Omega(\mathbf{x})}\mathbf{v}(\mathbf{x}',t)\,d\mathbf{x}'\right] \equiv G_{inv}^{(m)} * \overline{\mathbf{v}}.$$

Let us observe that for non-uniform filter width, the deconvolution operator and the integrals do not commute; to avoid commutation terms, the deconvolution-based equations can be written as [14,21]:

$$\int\limits_{\partial\Omega(\mathbf{x})}\widetilde{\mathbf{v}}\cdot\mathbf{n}\;dS = s, \tag{3}$$

$$\left[G_{inv}^{(m)}\right]^{-1} * \frac{\partial\widetilde{\mathbf{v}}}{\partial t} = \mathbf{f}_{conv} + \mathbf{f}_{diff} + \mathbf{f}_{press} + \mathbf{f}_{sgs}, \tag{4}$$

where

$$\mathbf{f}_{conv} = -\frac{1}{|\Omega(\mathbf{x})|}\int\limits_{\partial\Omega(\mathbf{x})}\widetilde{\mathbf{v}}\widetilde{\mathbf{v}}\cdot\mathbf{n}\;dS$$

$$\mathbf{f}_{diff} = \frac{2}{Re\;\;|\Omega(\mathbf{x})|}\int\limits_{\partial\Omega(\mathbf{x})}(\nabla^s\widetilde{\mathbf{v}})\cdot\mathbf{n}\;dS$$

$$\mathbf{f}_{pres} = -\frac{1}{|\Omega(\mathbf{x})|}\int\limits_{\partial\Omega(\mathbf{x})}p\mathbf{n}\;dS$$

$$\mathbf{f}_{sgs} = \frac{1}{|\Omega(\mathbf{x})|}\int\limits_{\partial\Omega(\mathbf{x})}\left[\frac{2}{Re}\left(\nabla^s\mathbf{v} - \nabla^s\widetilde{\mathbf{v}}\right) + \left(\widetilde{\mathbf{v}}\widetilde{\mathbf{v}} - \mathbf{v}\mathbf{v}\right)\right]\cdot\mathbf{n}\;dS$$

In the following, we disregard the source term s, fix $m = 2$ and write $A_{\mathbf{x}}^{(2)} = A$. The vector $\mathbf{f}_{sgs}$ represents the unresolved terms that can be either disregarded, in case of an implicit LES, or explicitly modelled by means of a SGS model. For details we refer the reader to [14,21,34].

3 Numerical simulation procedure

3.1 Computational grid

The computational domain V is partitioned by means of a structured Cartesian grid. We assume that the flow is homogeneous along the x (stream-wise) and z (span-wise) directions, so that volume averages of the variables are invariant for any translation of the set $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$. Along the homogeneous directions, the centers of the control volumes in which V has been decomposed are uniformly distributed as:

$$x_i = \left(i - \frac{1}{2}\right) \Delta x \ , \ \ (i = 2, \ldots N_x + 1) \, ,$$

$$z_k = \left(k - \frac{1}{2}\right) \Delta z \ , \ \ (k = 2, \ldots N_z + 1) \, ,$$

where $\Delta x = L_x/N_x$ and $\Delta z = L_z/N_z$ are the spatial discretization step sizes, and N_x and N_z the number of control volumes along x e z directions. Along direction y we used a non-uniform grid refined near the wall, following a stretching cosine-law. The coordinate of the centers of the control volumes along this direction are $y_j = \left(y_j^- + y_j^+\right)\big/2$. In the following the coordinates of the control volume faces are written in terms of the coordinates of the cell centers.

$$\begin{aligned} x_i^- &= x_i - \Delta x/2 \ , & x_i^+ &= x_i + \Delta x/2 \\ y_j^- &= y_j - h_y\left(j\right)/2 \ , & y_j^+ &= y_j + h_y\left(j\right)/2 \\ z_k^- &= z_k - \Delta z/2 \ , & z_k^+ &= z_k + \Delta z/2 \ . \end{aligned}$$

so the control volume $\Omega\left(\mathbf{x}\right)$ will be defined as:

$$\Omega\left(x_i, y_j, z_k\right) \equiv \Omega_{ijk} = \left[x_i^-, x_i^+\right] \times \left[y_j^-, y_j^+\right] \times \left[z_k^-, z_k^+\right] ,$$

$$\left|\Omega_{ijk}\right| = \Delta x h_y\left(j\right) \Delta z$$

where $h_y\left(j\right) = y_j^+ - y_j^-$ is the discretization step along y. Let us remark that this construction of the computational grid is associated to a co-location of the flow variables at the centers of each FV.

In this study we adopt an implicit LES modelling by assuming $\mathbf{f}_{sgs} = 0$. As recognized in [21,34], this approach corresponds to supplying an Approximate Deconvolution Model (ADM) for the top-hat based governing equations.

The time integration of equation (4) is performed by applying the classical second order Adams-Bashforth/Crank-Nicolson (AB/CN) semi-implicit scheme [16]. The implicit CN integration is applied for solving diffusion terms along the y direction, in order to obtain wider stability range near the solid walls where a non-uniform grid is used. The explicit AB method is used to solve the convection and diffusion terms in the x, z homogeneous directions. The complete set of the continuity and deconvolved momentum equation is split in time by means of an Approximate Projection Method (APM) [1,3,18,31]. The APM method is obtained by decoupling the velocity from the pressure in the momentum equation. According to the Helmholtz-Hodge Decomposition (HHD) theorem [9,13,20], the unknown velocity field $\tilde{\mathbf{v}}$ at each time step can be evaluated through a prediction-correction approach based on the following decomposition:

$$\tilde{\mathbf{v}}^{n+1} = \mathbf{v}^* - \Delta t \nabla \phi^{n+1}, \tag{5}$$

where an intermediate velocity vector field $\mathbf{v}^*$ (prediction field) and a potential scalar field $\nabla\phi$ (correction field) have to be obtained by solving two different equations.

The equation for the intermediate field $\mathbf{v}^*$ is obtained by performing the AB/CN integration of equation (4), while disregarding the pressure term [2].

The second (projection) step of the APM procedure consists in computing the correction field $\nabla\phi^{n+1}$ needed to obtain a divergence-free velocity field $\tilde{\mathbf{v}}$ in discrete sense. For this purpose we write the elliptic equation:

$$\left(D_1 + D_2 + D_3\right)\phi \;=\; \frac{1}{\Delta t \left|\Omega\left(\mathbf{x}\right)\right|} \int\limits_{\partial\Omega(\mathbf{x})} \mathbf{n} \cdot \mathbf{v}^* \; dS \quad , \tag{6}$$

for which a solution is guaranteed to exist, up to an additive constant, by prescribing non-homogeneous Neumann boundary condition on ∂V, fulfilling the compatibility condition.

Note that it is possible to derive a relation between the pressure p and the field ϕ, showing that $\nabla\phi$ results in an $O(\Delta t)$ approximation of the actual pressure gradient [13,21]. This is the reason why equation (6) is often referred to as the pressure equation.

In this Section we address the details of the discretization of the elliptic equation (6) in the APM procedure. This task requires careful examination, since a very large fraction (usually more than 60%) of the run time of a projection-based LES code is spent in the solution of the resulting linear system.

The surface integrals in (6) are approximated by using the mean value formula and second order central interpolation. Having collocated the velocity field and the ϕ field at the center of the FV, the velocity components normal to the flux sections must be expressed in term of the values at the center of the volume. This can be done by means of a linear interpolation, obtaining the following discrete equation for the ϕ field:

$$
\begin{aligned}
&-\phi_{i,j,k}^{n+1}\left[\frac{2}{\Delta x^2}+\frac{2}{\Delta z^2}+\frac{1}{h_y}\left(\frac{1}{\Delta y_{j+1}}+\frac{1}{\Delta y_j}\right)\right]+\frac{\phi_{i+1,j,k}^{n+1}+\phi_{i-1,j,k}^{n+1}}{\Delta x^2}+\\
&+\frac{\phi_{i,j,k+1}^{n+1}+\phi_{i,j,k-1}^{n+1}}{\Delta z^2}+\frac{\phi_{i,j+1,k}^{n+1}}{h_y\Delta y_{j+1}}+\frac{\phi_{i,j-1,k}^{n+1}}{h_y\Delta y_j}=\frac{u_{i+1,j,k}^{*}-u_{i-1,j,k}^{*}}{2\Delta t\Delta x}+\\
&+\frac{N_{j+1}^{north}v_{i,j+1,k}^{*}+\left(N_j^{north}-N_j^{south}\right)v_{i,j,k}^{*}-N_{j-1}^{south}v_{i-1,j,k}^{*}}{h_y\Delta t}+\frac{w_{i,j,k+1}^{*}-w_{i,j,k-1}^{*}}{2\Delta t\Delta z},
\end{aligned}
\tag{7}
$$

where the terms N_{j-1}^{south}, N_j^{south}, N_j^{north} and N_{j+1}^{north} are the linear shape functions along the non uniform directions y. The existence of the solution to the (7) is ensured by prescribing the boundary condition

$$
\left.\frac{\partial\phi}{\partial y}\right|_{i,jbnd,k}^{n+1}=\frac{1}{\Delta t}\left(v_{i,jbnd,k}^{*}-\tilde{v}_{i,jbnd,k}^{n+1}\right).
$$

Having prescribed the directional derivative of ϕ along the normal to the upper and lower boundary of the domain, the resulting linear system is formally singular. The elliptic problem is built to satisfy the following compatibility condition, which is both necessary and sufficient to obtain a solution:

$$
\int_{\Omega}\nabla^2\phi^{n+1}d\Omega=\frac{1}{\Delta t}\int_{\partial\Omega}\nabla\cdot\mathbf{v}^{*}dS.
$$

This condition is verified in discrete sense too. We also note that $R(A)\cap N(A)=\{0\}$, where A is the linear system matrix and $R(A)$ and $N(A)$ are its Range Space and Null Space, respectively. This property is useful in the choice of the linear equation solver (see Section 5). The coefficient matrix A is

banded, and because of the non uniform discretization along the y direction, it is non-symmetric.

Note that for a fixed computational grid the linear system matrix is constant throughout the simulation; therefore it is possible to build it just once, as well as to spend a substantial amount of time in computing an efficient preconditioners, given that these preprocessin steps will be amortized over a large number of time steps.

4 Multilevel Domain Decomposition Preconditioners

The main idea of this work is to investigate the behaviour of multilevel DD preconditioners, coupled with Krylov methods, in the solution of the pressure equation arising from a LES model. The basic, i.e. one-level, preconditioners considered here are the the Additive Schwarz (AS) ones, that decompose the system matrix A into overlapping submatrices A_i, corresponding to different subdomains, and solve separate linear systems involving these matrices. More precisely, the application of the AS preconditioner M_{AS} to vector r, formally $v = M_{AS}^{-1} r$, is carried out by solving systems of the type $A_i v_i = r_i$, and by combining the solutions v_i to get v. The r_i vectors are obtained from r by using suitable restriction operators, while v is obtained by prolongating and adding the v_i vectors.

Among the AS preconditioners, the variant known as Restricted AS (RAS) is the most efficient on systems coming from elliptic problems, in terms of both convergence rate and execution time [8,15]. A detailed description of these preconditioners is outside the scope of this work; see [35] for more details. Here we only note that the AS preconditioners can be formulated in terms of numerical linear algebra kernels, and hence can be implemented in an efficient, reliable and portable way by using such kernels [6].

AS preconditioners exhibit an intrinsic parallelism, since different submatrices can be processed in parallel; on the other hand, the convergence rate of the preconditioned solvers usually deteriorates as the number of submatrices increases. In order to reduce the dependency of the number of iterations on the degree of parallelism, a multilevel approach can be applied, where a coarse matrix is used to introduce a global coupling among the subsystems. We consider algebraic multilevel preconditioners, which build a sequence of coarser matrices and the related transfer operators by exploiting only the fine grid linear system, i.e. without explicitly using any information on the physical domain, and hence relieve the user from the burden of generating coarse grids and corresponding matrices. In the algebraic case, a coarse matrix A_C is usually built with a Galerkin approach, i.e. $A_C = R_C A R_C^T$, where R_C is

a suitable restriction operator from the fine to the coarse linear space. The coarse space and the restriction operator can be obtained by using a classical AMG approach [32,37] or an aggregation technique [4,39]. A comparison of geometric and algebraic multilevel preconditioners based on aggregation, on different CFD problems, has shown that the algebraic are not worse than the geometric ones, even for problems defined on simple geometries [27].

In this work we use the algebraic multilevel preconditioners implemented in the package MLD2P4 [10]. This package makes available different versions of multilevel preconditioners of Schwarz type, where the coarse matrices and transfer operators are obtained by using a smoothed aggregation technique [39]. The coarse space is built by grouping the vertices of the adjacency graph of the matrix, and the prolongator R_C^T is obtained by applying a smoother to a piecewice constant interpolation operator from the coarse to the fine space, in order to remove spurious oscillatory components from the range of the prolongator.

The construction and the solution of the coarse matrix system as described above is sequential; therefore parallel variants of the aggregation and solution algorithms have been considered to preserve the scalability of the one-level DD algorithms. In MLD2P4 a decoupled aggregation technique [38] is used to build a coarse matrix A_C distributed among the processors, and parallel block-Jacobi iterations, with incomplete or complete LU factorization on the blocks of A_C, are available to (approximately) solve the coarse system. A distributed solution of the coarse system can be obtained also by using the distributed sparse LU factorization provided by SuperLU-DIST [26]. The coarse matrix may be also replicated on the processors and solved by the UMFPACK sequential LU factorization [12]. More details on the multilevel preconditioners available in MLD2P4 can be found in [11]. An analysis of the performance and scalability of the various versions of two-level Schwarz preconditioners implemented in MLD2P4 is reported in [7,11].

5 Numerical experiments

We have applied RAS and the RAS-based two- and three-level preconditioners available in MLD2P4, coupled with Krylov solvers from the PSBLAS library [17], to the pressure system arising in a LES of a bi-periodical channel flow. The Reynolds number of this flow, referred to the shear velocity, is $Re_\tau = 180$; a Poiseuille flow, with a random Gaussian perturbation, is assumed as initial condition. The computational grid, which is non-uniform only in the y direction, has $140 \times 32 \times 45$ nodes, leading to a nonsymmetric pressure system matrix with dimension 201600 and 1398600 nonzero entries. The time step Δt in the numerical integration is set to 10^{-4}, in order to meet stability requirements.

Even though this test is based on a model problem, it provides useful indications on the behaviour of algebraic multilevel Schwarz preconditioners in a more general LES context. Therefore, the present study is a first step towards a deeper understanding of the impact of such preconditioners in the LES of realistic problems.

We discuss here the results obtained by solving the pressure system at the first time step of the simulation procedure. Indeed, the matrix of this system does not change during the whole flow simulation. On the other hand, we have not observed significant differences in the behaviour of the preconditioned solvers during the first 20000 time steps of numerical integration.

All the preconditioners have been applied, as right preconditioners, with RGMRES(30), i.e. Restarted GMRES with restarting parameter equal to 30, and with BiCGSTAB. The null vector has been chosen as starting guess and the Krylov solvers have been stopped when the ratio between the 2-norms of the residual and of the right-hand-side is smaller than 10^{-7}. A maximum number of 3000 iterations has been set. A row-block decomposition of the pressure equation matrix has been considered, with a conformal distribution of the right-hand side and solution vectors. The number of submatrices used in RAS has been set equal to the number of processors.

We note that the choice of GMRES has been made by taking into account that this method is able to compute a solution of a singular system $Ax = b$, if the system is consistent and $R(A) \bigcap N(A) = \{0\}$, where $R(A)$ and $N(A)$ are the Range Space and the Null Space of A [5]. As already observed in Section 3, the pressure system satisfies these requirements. Conversely, BiGSTAB has been considered even if, in our knowledge, a study of its convergence behaviour in the case of singular systems is not available.

All the experiments have been carried out on the *spaci* parallel machine operated by the Naples branch of ICAR-CNR. This machine is a HP XC 6000 Linux cluster with 64 bi-processor nodes. Each node is an Intel Itanium 2 Madison with 1.4 Ghz, running the HP Linux for High Performance Computing, based on Red Hat Enterprise Linux AS 3 with Kernel 2.4.21. Each single node is equipped with 4 GB of RAM. The main interconnection network is Quadrics QsNetII Elan 4, which has a sustained bandwidth of 900 MB/sec. and a latency of about 5 μsec for large messages. The GNU Compiler Collection version 4.2 and the HP MPI implementation, version 2.01, have been used. MLD2P4 and PSBLAS have been installed on top of the BLAS implementation provided by ATLAS 3.6.0 and of BLACS 1.1.

In the following we analyze the results of the numerical experiments in terms of performance, scalability and accuracy. We focus on the two-level and three-level preconditioners which, at the coarsest level, build a distributed linear

system and apply to it four block-Jacobi sweeps, with ILU(0) on the blocks. RAS, with ILU(0) on the local sub-matrices, is used as fine-level smoother. Indeed, on our test problem, these preconditioners generally achieve the best tradeoff between performance and scalability among the various versions of multilevel preconditioners available in MLD2P4. These two- and three-level preconditioners are referred to as 2LDI4 and 3LDI4, respectively. RAS with ILU(0) is also considered as reference one-level preconditioner.

In Tables 1 and 2 we report the number of iterations performed by BICGSTAB and RGMRES (Iterations), and the times, in seconds, to setup the preconditioner (Tprec) and to solve the preconditioned system (Tsolve). We also report the 2-norms of the residuals corresponding to the computed solutions. These results have been obtained using $np = 1, 2, 4, 8, 16, 32$ processors and overlap $ov = 0, 1, 2$ in the RAS smoother.

We observe that, for all preconditioners, the residuals of the solutions computed with RGMRES(30) and BiCGSTAB are of the same order of magnitude. Furthermore, a comparison with a reference solution obtained by running sequential SOR until the ∞-norm of the residual has been driven to double-precision machine accuracy, has shown that the solutions from the Krylov methods differ from the SOR one only by an additive constant, in agreemenbt with the Neumann and periodic boundary conditions prescribed on the flow domain.

Looking at the performance data, we see that in the case of the multilevel preconditioners the setup time accounts for a large part of the total time, while for RAS it is often negligible. On the other hand, the multilevel setup phase shows a good scalability, especially for $ov = 0$, confirming the appropriateness of the choice of distributing the coarsest-level matrix.

Since, as already mentioned, the matrix of the pressure system does not change during the whole flow simulation, the preconditioner has to be built only once and its setup time becomes negligible in a typical simulation requiring about 10^8 time steps. For this reason we focus our analysis on the solution phase of the preconditioned system.

Let us consider first the results obtained with RGMRES. From Table 1 we see that $2LDI4$ and $3LDI4$ significantly reduce the number of iterations with respect to RAS, when up to 16 processors are used. With 2LDI4 the reduction of number of iterations ranges from 70% for $np = 1$ to 78% for $np = 16$; with $3LDI4$ it ranges from 79% for $np = 1$ to 83% for $np = 16$. In this case, the iteration count is approximately constant as the number of processors increases. When more than 16 processors are used, the number of iterations of the multilevel preconditioners has an erratic behaviour. In particular, it is significantly greater than the RAS one for $np = 32$ and $ov = 1, 2$ and for

$np = 64$ and $ov = 2$ (in the latter case the iterative solver does not reach the required accuracy within the maximum number of iterations). A steep increase of the iteration count on 32 and 64 processors can be observed with RAS too, but in this case a larger overlap corresponds to a smaller number of iterations. The behaviour of all the preconditioned solvers on 32 and 64 processors seems to be related to the possible singularity of the sub-matrices to which ILU is applied, at either the fine or the coarse levels, and deserves a careful investigation.

The best solution times are obtained with 3LDI4, which generally outperforms both 2LDI4 and RAS for each value of the overlap, except for $np = 32$. In particular, the smallest solution time (about 0.86 sec.) is obtained with 3LDI4 for $np = 16$ and $ov = 0$. On the other hand, the best behaviour in terms of scalability is shown by the RAS preconditioner, which has the best speedup line, with a maximum value of 11.9 for $np = 32$ and $ov = 0$.

¿From the results reported in Table 2, we see that BiCGSTAB, will all the three preconditioners, has a general behaviour similar to RGMRES, in terms of number of iterations, execution time and scalability. However, the number of BiCGSTAB iterations is smaller than for RGMRES and BiCGSTAB is always able to reach the required accuracy within the maximum number of iterations. As a consequence, the BiCGSTAB solution time is often smaller, especially for $np \leq 8$. The smallest BiCGSTAB times are obtained with 3LDI4, which outperforms both 2LDI4 and RAS, except in for $np = 32, 64$, where RAS is superior. In particular, for $np = 64$ and $ov = 0$ RAS achieves the smallest among all the execution times (about 0.44 sec.). The corresponding speedup, with respect to RAS on 1 processor, is of 25.2. The best multilevel time (0.75 sec.) is obtained with 3LDI4 for $np = 16$ and $ov = 0$; the speedup for this case is 7.3.

6 Conclusions and Future Work

We have analyzed the behaviour of Krylov solvers preconditioned with parallel algebraic multilevel DD methods, implemented in the MLD2P4 package, in the solution of singular elliptic linear systems arising in the numerical simulation of uncompressible wall-bounded turbulent flows. Such preconditioners appear to be effective in terms of efficiency and scalability and we expect that they can be efficiently used in the development of scalable parallel codes for accurate simulations of turbulent flows in real-world applications.

Future work will be devoted to evaluate the impact of these multilevel preconditioners in a complete LES of flows, in both simple and complex geometries, especially those for which reference experimental solutions are available.

	Iterations			Tprec			Tsolve			Residual 2-norm		
np	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4
						$ov = 0$						
1	107	32	22	0.439	4.106	5.156	19.130	9.134	6.290	0.00012	9.15e-05	0.000125
2	108	31	24	0.223	2.613	3.264	9.906	5.626	3.801	0.000138	0.000125	9.85e-05
4	108	30	25	0.104	1.397	1.687	4.150	2.541	1.923	0.000131	0.000138	0.000107
8	114	31	30	0.051	0.757	1.020	2.120	1.530	1.259	0.000135	0.000127	9.72e-05
16	197	42	33	0.025	0.481	0.686	1.973	1.023	0.859	0.00014	0.000116	0.000141
32	245	61	64	0.014	0.297	0.566	1.599	1.124	2.021	0.00014	0.00012	0.000134
64	304	98	72	0.038	0.374	0.559	1.734	1.527	1.648	0.00014	0.00012	0.000137
						$ov = 1$						
1	107	32	22	0.440	4.166	5.144	19.990	9.247	6.407	0.00012	9.15e-05	0.000125
2	107	30	23	0.376	2.754	3.434	10.420	5.511	3.743	0.000133	0.000139	0.000125
4	107	30	25	0.305	1.488	1.847	4.648	2.609	2.034	0.000134	0.000114	7.61e-05
8	108	30	29	0.207	0.916	1.159	2.399	1.416	1.494	0.000126	0.000128	9.7e-05
16	110	35	32	0.176	0.629	1.007	1.395	1.071	0.908	0.000131	9.63e-05	0.000136
32	163	239	200	0.179	0.459	0.865	1.454	5.818	5.233	0.00014	0.000136	0.00012
64	213	112	147	0.286	0.329	0.822	2.282	2.640	3.973	0.000138	0.00014	0.000134
						$ov = 2$						
1	107	32	22	0.434	4.115	5.157	20.010	9.242	6.390	0.00012	9.15e-05	0.000125
2	107	30	23	0.528	2.924	3.601	10.850	5.747	3.813	0.000124	0.000139	0.000125
4	107	30	25	0.417	1.655	1.994	4.794	2.982	2.061	0.000128	0.000114	7.58e-05
8	108	30	29	0.355	1.066	1.367	2.696	1.701	1.394	0.000121	0.000126	9.59e-05
16	109	35	32	0.504	0.949	0.988	1.819	1.110	1.018	0.00014	9e-05	0.000131
32	142	253	191	0.329	0.958	1.065	1.823	6.421	5.641	0.000133	0.00014	0.000139
64	149	3000	3000	0.433	0.451	0.913	1.437	77.790	107.200	0.000141	83.2	31.6

Table 1
Statistics for RGMRES with RAS, 2LDI4 and 3LDI4.

	Iterations			Tprec			Tsolve			Residual 2-norm		
np	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4	RAS	2LDI4	3LDI4
						$ov = 0$						
1	59	20	13	0.434	4.109	5.179	10.500	7.501	5.487	3.61e-05	0.000101	8.04e-05
2	55	19	13	0.223	2.615	3.302	5.405	5.007	3.121	0.000127	0.000119	0.000119
4	57	19	17	0.103	1.317	1.705	2.835	2.544	2.091	0.000104	6.19e-05	2.79e-05
8	59	19	17	0.051	0.747	1.024	1.454	1.387	1.173	9.78e-05	4.67e-05	0.000109
16	78	21	19	0.025	0.475	0.678	0.987	0.839	0.753	0.000125	0.000138	0.000132
32	101	36	37	0.022	0.342	0.724	0.688	1.156	1.151	0.000125	0.000112	0.0001
64	105	47	43	0.007	0.220	0.556	0.389	1.480	2.371	0.000114	7.6e-05	0.000111
						$ov = 1$						
1	59	20	13	0.434	4.122	5.191	11.310	7.571	5.544	3.61e-05	0.000101	8.04e-05
2	62	21	13	0.376	2.768	3.427	6.867	5.747	3.243	7.96e-05	1.9e-05	7.14e-05
4	59	20	16	0.253	1.473	1.862	3.417	2.830	2.157	6.44e-05	3.49e-05	2.36e-05
8	57	18	18	0.200	0.898	1.172	1.783	1.414	1.358	0.000132	6.77e-05	7.86e-05
16	99	24	19	0.175	0.626	0.823	1.862	1.487	1.082	0.000116	9.79e-05	7.37e-05
32	73	164	146	0.166	0.448	1.022	0.905	6.927	6.279	0.000141	9.64e-05	8.16e-05
64	85	58	86	0.140	0.334	0.694	0.929	2.140	5.407	0.000128	0.00014	0.000139
						$ov = 0$						
1	59	20	13	0.434	4.130	5.169	11.320	7.535	5.511	3.61e-05	0.000101	8.04e-05
2	56	21	13	0.536	2.914	3.571	6.559	6.022	3.311	8.62e-05	1.83e-05	7.2e-05
4	59	19	14	0.405	1.631	1.986	3.762	2.913	1.930	0.000101	5.89e-05	0.000126
8	58	18	18	0.366	1.057	1.353	2.149	1.530	1.485	0.000117	4.6e-05	6.92e-05
16	92	24	18	0.329	0.794	0.989	2.456	1.300	1.139	9.5e-05	3.88e-05	0.000109
32	94	172	182	0.321	0.790	0.880	1.991	8.407	10.350	7.39e-05	6.9e-05	7.17e-05
64	103	2558	973	0.436	1.222	1.211	2.092	115.200	59.590	0.000107	9.43e-05	0.000111

Table 2
Statistics for BiGCSTAB with RAS, 2LDI4 and 3LDI4.

References

[1] A.S. Almgren, J.B. Bell, W.G. Szymczak, *A numerical method for the incompressible Navier-Stokes equations based on an approximate projection*, (SIAM J. Sci. Comput., 17, 1996).

[2] A. Aprovitola, F.M. Denaro, *On the application of congruent upwind discretizations for large eddy simulations* (J. Comp. Phys., 194/1, 2004).

[3] A. Aprovitola, F.M. Denaro, *A non-diffusive, divergence-free, Finite Volume-based double projection method on non-staggered grids* (Int. J. Num. Methods in Fluids, 53, 2007).

[4] M. Brezina, P. Vanek, *A Black-Box Iterative Solver Based on a Two-Level Schwarz Method* (Computing, 63, 1999).

[5] P.N. Brown, H.F. Walker *GMRES on (Nearly) Singular Systems* (SIAM J. Matrix Anal. Appl., 18, 1997).

[6] A. Buttari, P. D'Ambra, D. di Serafino, S. Filippone, *Extending PSBLAS to Build Parallel Schwarz Preconditioners*, in J. Dongarra, K. Madsen, J. Wasniewski, eds., Proceedings of PARA 04 Workshop on State of the Art in Scientific Computing (Lecture Notes in Computer Science, Springer, 3732, 2005).

[7] A. Buttari, P. D'Ambra, D. di Serafino, S. Filippone, *2LEV-D2P4: a package of high-performance preconditioners for scientific and engineering applications*, (Applicable Algebra for Engineering, Communication and Computing, 18, 2007).

[8] X.C. Cai, M. Sarkis, *A Restricted Additive Schwarz Preconditioner for General Sparse Linear Systems* (SIAM J. Sci. Comput., 21 1999).

[9] A. J. Chorin, *Numerical Solution of the Navier-Stokes Equations*, (Math. Comp., 22, 1968).

[10] P. D'Ambra, D. di Serafino, S. Filippone, *MLD2P4, Users' Guide* (in preparation).

[11] P. D'Ambra, D. di Serafino, S. Filippone *On the development of PSBLAS - based parallel two-level Schwarz preconditioners* (Applied Numerical Mathematics, DOI 10.1016/j.apnum.2007.01.006, Elsevier).

[12] T.A. Davis, *Algorithm 832: UMFPACK - an Unsymmetric-pattern Multifrontal Method with a Column Pre-ordering Strategy* (ACM Transactions on Mathematical Software, 30, 2004. See also `http://www.cise.ufl.edu/~davis/`).

[13] F.M. Denaro, *On the application of the Helmholtz-Hodge decomposition in projection methods for the numerical solution of the incompressible Navier-Stokes equations with general boundary conditions*, (Int. J. Num. Methods in Fluids, 43, 2003).

[14] G. De Stefano, F.M. Denaro, G. Riccardi, *High Order Filtering Formulation for Control Volumes Simulation of Turbulent Flows* (Int. J. Num. Methods in Fluids, 37, 2001).

[15] E. Efstathiou, J. G. Gander, *Why Restricted Additive Schwarz Converges Faster than Additive Schwarz* (BIT Numerical Mathematics, 43, 2003).

[16] J.H. Ferziger, M. Peric, *Computational Methods for Fluid Dynamics* (Springer, 1996).

[17] S. Filippone and M. Colajanni, *PSBLAS: A Library for Parallel Linear Algebra Computation on Sparse Matrices* (ACM Transactions on Mathematical Software, 26, 2000).

[18] P.M. Gresho *Incompressible fluid dynamics: some fundamental formulation issues* (*Ann. Rev. Fluid Mech.*, 23, 1991).

[19] W. Gropp, D. Kaushik, D. Keyes, B. Smith, *High-Performance Parallel Implicit CFD*, (Parallel Computing, 27, 2001).

[20] W.V.D. Hodge, *The Theory and Application of Harmonic Integrals*, (Cambridge Univ. Press, 1952).

[21] P. Iannelli, F.M. Denaro, G. De Stefano, *A Deconvolution-based Fourth Order Finite Volume Method for Incompressibile Flows on Non-Uniform Grids* (Int. J. Num. Methods in Fluids, 00, 2002)

[22] J. Jimenez, *Computing High-reynolds-number Turbulence: Will Simulations Ever Replace Experiments?* (Journal of Turbulence, 4, 2003).

[23] J. Kim, P. Moin, *Application of a Fractional-step Method to Incompressible Navier-Stokes Equations* (Journal of Computational Physics, 59, 1985).

[24] P.K. Kundu, Ira M. Cohen, *Fluid Mechanics*, 2nd Edition (Elsevier, 2002).

[25] M. Lesieur, *Turbulence in fluids*, 3rd edition (Kluwer, 1997).

[26] X.S. Li, J.W. Demmel, *SuperLU-DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems* (ACM Transactions on Mathematical Software, 29, 2003).

[27] P. Lin, M. Sala, J. Shadid, R. Tuminaro, *Performance of Fully-Coupled Algebraic Multilevel Domain Decomposition Preconditioners for Incompressible Flow and Transport* (Int. J. for Num. Meth. in Eng., 19, 2004).

[28] J. Mathieu, J. Scott, *An Introduction to Turbulent Flow* (Cambridge University Press, 2000).

[29] P. Moin, J. Kim, *Tackling Turbulence with Supercomputers* (Scientific American, January 1997).

[30] U. Piomelli, A. Scotti, E. Balaras, *Large-Eddy Simulations of Turbulent Flows, from Desktop to Supercomputer*, Lecture Notes in Computer Science, Vol. 1981 (Springer, 2001).

[31] W.J. Rider, *Approximate Projection Methods for Incompressible Flow: implementation, variants and robustness* (Technical Report LA-UR-200, Los Alamos National Laboratory, 1995).

[32] J.W. Ruge and K. Stüben, Algebraic Multigrid (AMG), in S.F. Mc Cormick, editor, Multigrid Methods, Frontiers in Applied Mathematics, vol. 3 (SIAM, Philadelphia, 1987).

[33] Stevens DE, Chan ST, Gresho P. *An Approximate Projection Methods for Incompressible Flow* (International Journal for Numerical Method in Fluids, 40, 2002).

[34] P. Sagaut, *Large Eddy Simulation for Incompressible Flows. An Introduction* (Springer, 2001).

[35] B. Smith, P. Bjørstad and W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, (Cambridge University Press, 1996).

[36] C. Stone, S. Menon, *Large-eddy Simulations on Distributed Shared Memory Clusters* (Journal of Parallel and Distributed Computing, 64, 2004).

[37] K. Stüben, Algebraic Multigrid (AMG): an Introduction with Applications, in A. Schüller, U. Trottenberg, C. Oosterlee, eds., Multigrid (Academic Press, 2000).

[38] R.S. Tuminaro and C. Tong, *Parallel Smoothed Aggregation Multigrid: Aggregation Strategies on Massively Parallel Machines*, in J. Donnelley, editor, Proceedings of SuperComputing 2000, Dallas, 2000.

[39] P. Vanek, J. Mandel, M. Brezina, *Algebraic Multigrid by Smoothed Aggregation for Second and Fourth Order Elliptic Problems*, (Computing, 56, 1996).

[40] F. Sotiropoulos, S. Abdallah, *The discrete continuity equation in primitive variable solutions of incompressible flow*, (*J. Comp. Phys.*, 95, 1991).