PARALLEL MULTILEVEL RESTRICTED SCHWARZ PRECONDITIONERS WITH POLLUTION REMOVING FOR PDE-CONSTRAINED OPTIMIZATION*

ERNESTO E. PRUDENCIO† AND XIAO-CHUAN CAI‡

Abstract. We develop a class of V-cycle type multilevel restricted additive Schwarz (RAS) methods and study the numerical and parallel performance of the new fully coupled methods for solving large sparse Jacobian systems arising from the discretization of some optimization problems constrained by nonlinear partial differential equations. Straightforward extensions of the one-level RAS to multilevel do not work due to the pollution effects of the coarse interpolation. We then introduce, in this paper, a pollution removing coarse-to-fine interpolation scheme for one of the components of the multi-component linear system, and show numerically that the combination of the new interpolation scheme with the RAS smoothed multigrid method provides an effective family of techniques for solving rather difficult PDE-constrained optimization problems. Numerical examples involving the boundary control of incompressible Navier-Stokes flows are presented in detail.

Key words. Schwarz preconditioners, domain decomposition, multilevel methods, parallel computing, partial differential equations constrained optimization, inexact Newton, flow control.

1. Introduction.

There are two major families of Newton techniques for solving nonlinear optimization problems: reduced space methods, characterized by the partition of the problem into smaller ones at each Newton step, and full space ones. As computers become more powerful in processing speed and memory capacity, full space methods become more attractive, as exemplified by Lagrange-Newton-Krylov-Schur [3, 4] and one-level *Lagrange-Newton-Krylov-Schwarz* [25].

A key element of any successful full space approach is the Jacobian preconditioner, which needs to be able to simultaneously reduce the condition number and provide good parallel scalability [20]. In this paper, we focus on fully coupled Schwarz type preconditioners in which all components of the linear system are treated equally, i.e., no variables are eliminated as in some Schur complement type approaches. Among Schwarz type preconditioners [29, 30], the recently introduced restricted versions [6, 9] seem to be more robust and computationally more efficient, especially for harder problems such as those indefinite, highly ill-conditioned, multi-components systems arising from PDE-constrained optimizations. The extension of the one-level restricted additive Schwarz method (RAS) to multilevel using the multigrid V-cycle idea and standard coarse to fine interpolations is straightforward, but may not work as expected due to the pollution effects of the interpolation. After many experiments with some flow control problems, we identified the source of the pollution at one of the three components of the Jacobian system, namely the Lagrange multiplier. Using a pollution removing interpolation scheme we have then been able to restore the robust and fast convergence of RAS. We only discuss linear versions of Schwarz methods even though nonlinear versions can also be obtained [8, 13]. We refer to [2, 19, 28] for the analysis of some preconditioning techniques for optimal control problems and general saddle point problems.

*The research is supported in part by the National Science Foundation, CCR-0219190, ACI-0072089 and ACI-0305666, and in part by the Department of Energy, DE-FC02-01ER25479.

†Advanced Computations Department, Stanford Linear Accelerator Center, Menlo Park, CA 94025 (prudenci@slac.stanford.edu).

‡Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309 (cai@cs.colorado.edu).

In this paper we only consider optimization problems with equality constraints:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{W}} & \mathcal{F}(\mathbf{x}) \\ \text{s.t.} & \mathbf{C}(\mathbf{x}) = \mathbf{0} \in \mathbf{Y}. \end{cases} \tag{1.1}$$

Here $\mathbf{W}$ and $\mathbf{Y}$ are normed spaces, $\mathbf{W}$ is the space of optimization variables, $\mathcal{F} : \mathbf{W} \to \mathbb{R}$ is the objective functional and $\mathbf{C} : \mathbf{W} \to \mathbf{Y}$. The associated Lagrangian functional $\mathcal{L} : \mathbf{W} \times \mathbf{Y}^* \to \mathbb{R}$ is defined as

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \equiv \mathcal{F}(\mathbf{x}) + \langle \boldsymbol{\lambda}, \mathbf{C}(\mathbf{x}) \rangle_{\mathbf{Y}}, \quad \forall\ (\mathbf{x}, \boldsymbol{\lambda}) \in \mathbf{W} \times \mathbf{Y}^*,$$

where $\mathbf{Y}^*$ is the adjoint space of $\mathbf{Y}$, $\langle \cdot, \cdot \rangle_{\mathbf{Y}}$ denotes the duality pairing and variables $\boldsymbol{\lambda}$ are called Lagrange multipliers or adjoint variables. In many cases it is possible to prove that, if $\hat{\mathbf{x}}$ is a (local) solution of (1.1) then there exist Lagrange multipliers $\hat{\boldsymbol{\lambda}}$ such that $(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}})$ is a critical point of $\mathcal{L}$. So, under sufficient smoothness assumptions, one proves that a solution has to necessarily solve a system of equations, called Karush-Kuhn-Tucker (KKT) or optimality system.

For the numerical solution of (1.1) we discretize it with a mesh Ω_h of characteristic size $h > 0$, obtaining a finite dimensional optimization problem with $\mathbf{W} = \mathbb{R}^{n_h}$ and $\mathbf{Y} = \mathbb{R}^{m_h} = \mathbf{Y}^*$. The KKT system becomes $\nabla \mathcal{L}_h(\hat{\mathbf{x}}, \hat{\boldsymbol{\lambda}}) = \mathbf{0} \in \mathbb{R}^{n_h + m_h}$. Omitting symbols "$h$" and " $\hat{}$ ", and denoting $N \equiv n + m$ and $\mathbf{X} \equiv (\mathbf{x}, \boldsymbol{\lambda}) \in \mathbb{R}^N$, one has $\mathcal{L} : \mathbb{R}^N \to \mathbb{R}$ and an optimality system $\nabla \mathcal{L}(\mathbf{X}) = \mathbf{0} \in \mathbb{R}^N$, which is solved in this paper by inexact Newton's method [15, 23] globalized by line search, according to the heuristic explained in [25].

Let ξ be some real value and $\hat{\mathbf{x}}(\xi)$ denote the solution, if it exists, of the finite dimensional problem

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbb{R}^n} & \mathcal{F}(\mathbf{x}) & & \\ \text{s.t.} & \mathbf{C}_1(\mathbf{x}) & = & 0, \\ & \vdots & & \\ & \mathbf{C}_{i-1}(\mathbf{x}) & = & 0, \\ & \mathbf{C}_i(\mathbf{x}) & = & \xi, \\ & \mathbf{C}_{i+1}(\mathbf{x}) & = & 0, \\ & \vdots & & \\ & \mathbf{C}_m(\mathbf{x}) & = & 0, \end{cases}$$

with arbitrarily fixed $1 \leqslant i \leqslant m$. Under appropriate assumptions, one can prove [16, 26]

$$\left. \frac{\partial \mathcal{F}(\hat{\mathbf{x}}(\xi))}{\partial \xi} \right|_{\xi=0} = \lambda_i, \tag{1.2}$$

that is, the Lagrange multiplier λ_i indicates how sensitive the optimal value is to changes on the i-th constraint. The module $|\lambda_i|$ gives a measure of how "effective" the corresponding constraint is. In the case of constraints corresponding to the discretized equations of boundary value problems, one might intuitively expect (1) different equations to have different effects on the objective function and (2) the "effectiveness" of a given original PDE constraint to have a "smooth behavior" throughout the domain where the PDE acts over. Such intuitive expectations match result (1.2). One less

obvious symptom, however, is the potential discontinuity between those Lagrange multipliers corresponding to PDEs governing the system behavior inside the problem domain and those Lagrange multipliers corresponding to boundary and initial boundary conditions. Indeed, we have observed such discontinuity in our test problems.

The rest of the paper is organized as follows. Section 2 explains multilevel restricted Schwarz preconditioners and Section 3 introduces the pollution removing interpolation. In Section 4 we test the combination of both approaches on some boundary flow control problems. Final conclusions are given in Section 5.

2. Multilevel restricted Schwarz preconditioners. Schwarz methods can be used in one-level or multilevel variants and, in each case, in combination with additive and/or multiplicative algorithms [30]. They can be also used as linear [14] and nonlinear preconditioners [8].

In this section we introduce a multilevel version of the one-level RAS preconditioner initially studied in [6, 9]. The multilevel preconditioner is applicable to general linear systems arising from the discretized PDEs on a mesh using finite element or finite difference methods. Let $\Omega \subset \mathbb{R}^2$ be a bounded open domain on which a PDE is defined and a discretization is performed with a mesh Ω_h of characteristic size $h > 0$. To obtain the overlapping partition, we first divide Ω_h into non-overlapping subdomains Ω_j, $j = 1, \ldots, N_S$. We then expand each Ω_j to $\Omega_j^{'}$, i.e., $\Omega_j \subset \Omega_j^{'}$. The overlap $\delta > 0$ is defined as the minimum distance between $\partial\Omega_j^{'}$ and $\partial\Omega_j$, in the interior of Ω. For boundary subdomains we simply cut off the part outside Ω. Let $H > 0$ denote the characteristic diameter of $\{\Omega_j\}$.

Let N and N_j denote the number of degrees of freedom associated to Ω and $\Omega_j^{'}$, respectively. Let $\mathbf{K}$ be a $N \times N$ sparse matrix of a linear system

$$\mathbf{K}\mathbf{p} = \mathbf{b} \tag{2.1}$$

that needs to be solved during the application of an algorithm for the numerical solution of the discretized differential equations. Let d indicate the number of degrees of freedom per mesh point. For simplicity let us assume that d is the same throughout the entire mesh. We define the $N_j \times N$ matrix $\mathbf{R}_j^{\delta}$ as follows: its $d \times d$ block element $(\mathbf{R}_j^{\delta})_{l_1,l_2}$ is either (a) an identity block if the integer indices $1 \leqslant l_1 \leqslant N_j/d$ and $1 \leqslant l_2 \leqslant N/d$ are related to the same mesh point and this mesh point belongs to $\Omega_j^{'}$ or (b) a zero block otherwise. The multiplication of $\mathbf{R}_j^{\delta}$ with a $N \times 1$ vector generates a smaller $N_j \times 1$ vector by discarding all components corresponding to mesh points outside $\Omega_j^{'}$. The $N_j \times N$ matrix $\mathbf{R}_j^{0}$ is similarly defined, with the difference that its application to a $N \times 1$ vector also zeros out all those components corresponding to mesh points on $\Omega_j^{'} \setminus \Omega_j$. Let $\tilde{\mathbf{K}}_j$ be defined as

$$\tilde{\mathbf{K}}_j \;\equiv\; \mathbf{R}_j^{\delta}\,\mathbf{K}\,\left(\mathbf{R}_j^{\delta}\right)^T,$$

that is, as the $N_j \times N_j$ matrix related to a subdomain problem having zero Dirichlet boundary conditions at regions of $\partial\Omega_j^{'}$ not coinciding with $\partial\Omega$. We assume $\tilde{\mathbf{K}}_j$ to be nonsingular and denote by $\tilde{\mathbf{B}}_j^{-1}$ either the inverse of or a preconditioner for $\tilde{\mathbf{K}}_j$. The one-level classical, right restricted (r-RAS) and left restricted (l-RAS) additive Schwarz preconditioners for $\mathbf{K}$ respectively are defined as [6, 9, 14]

$$\mathbf{B}_{\delta\delta}^{-1} = \sum_{j=1}^{N_s} \left(\mathbf{R}_j^{\delta}\right)^T \tilde{\mathbf{B}}_j^{-1} \mathbf{R}_j^{\delta}, \quad \mathbf{B}_{\delta 0}^{-1} = \sum_{j=1}^{N_s} \left(\mathbf{R}_j^{\delta}\right)^T \tilde{\mathbf{B}}_j^{-1} \mathbf{R}_j^{0}, \quad \mathbf{B}_{0\delta}^{-1} = \sum_{j=1}^{N_s} \left(\mathbf{R}_j^{0}\right)^T \tilde{\mathbf{B}}_j^{-1} \mathbf{R}_j^{\delta}.$$

When the distinction is not important, we will denote a Schwarz preconditioner simply by $\mathbf{B}^{-1}$. We refer to [17, 21, 22] for further analysis and examples of one-level restricted preconditioning techniques.

For the description of multilevel Schwarz preconditioners [31], let us use index $i = 0, 1, \ldots, L-1$ to designate any of the $L \geqslant 2$ levels. All previously defined entities using the subindex "j" will now use double subindexes "i, j": $\mathbf{\Omega}_{i,j}$, $\mathbf{\Omega}'_{i,j}$, $N_{i,j}$, $\mathbf{R}^{\delta}_{i,j}$, $\mathbf{R}^{0}_{i,j}$, $\tilde{\mathbf{K}}_{i,j}$ and $\tilde{\mathbf{B}}^{-1}_{i,j}$. All previously defined entities using no subindex will now use the subindex "i": h_i, N_i, $N_{S,i}$, H_i, δ_i, $\mathbf{B}^{-1}_{i,\delta\delta}$, $\mathbf{B}^{-1}_{i,\delta 0}$, $\mathbf{B}^{-1}_{i,0\delta}$, $\mathbf{B}^{-1}_{i}$ and $\mathbf{K}_i$, with the eventual notation $\mathbf{K}_{L-1} = \mathbf{K}$. The L meshes are not assumed to be either nested or structured. Let $\mathbf{I}_i$ denote the identity operator and, for $i > 0$, let

$$\mathcal{I}_i^{i-1} : \mathbb{R}^{N_i} \to \mathbb{R}^{N_{i-1}}$$

denote a linear restriction operator from level i to level $i-1$ and let

$$\mathcal{I}_{i-1}^{i} : \mathbb{R}^{N_{i-1}} \to \mathbb{R}^{N_i}$$

denote a linear interpolation operator from level $i-1$ to level i. Given the iterate used for the computation of $\mathbf{K}_{L-1}$, the computation of coarse matrices $\mathbf{K}_i$ (i.e., $0 \leqslant i \leqslant L-2$) proceeds recursively from the finest coarse level $i = L-2$ until the coarsest level $i = 0$ by simply first restricting or injecting the finer iterate and then computing the Jacobian. Multilevel Schwarz preconditioners are obtained through the combination of one-level Schwarz preconditioners $\mathbf{B}_i^{-1}$ assigned to each level. Here we focus on multilevel preconditioners that can be seen as multigrid (MG) V-cycle algorithms [5] having Schwarz preconditioned Richardson working as the pre and the post smoother at each level $i > 0$, with $\mathbf{B}^{-1}_{i,\text{pre}}$ preconditioning the $\mu_i \geqslant 0$ pre smoother iterations and $\mathbf{B}^{-1}_{i,\text{post}}$ preconditioning the $\nu_i \geqslant 0$ post smoother iterations. In a general MG V-cycle algorithm with $L \geqslant 2$ levels, given the current iterate $\mathbf{p}^{(\ell)}$ for the solution of (2.1), the next iterate is computed as $\mathbf{p}^{(\ell+1)} = AlgV(\mathbf{b}, L, \mathbf{p}^{(\ell)})$, where the procedure $\mathbf{v}_i = AlgV(\mathbf{b}_i, i, \mathbf{v}_i)$ consists of the following steps:

```
if i = 0
    Solve K_0 v_0 = b_0;                                  /* Coarsest correction */
else
    Smooth μ_i times K_i v_i = b_i :                      /* Pre-smoothing */
    (b_i − K_i v_i) = (I_i − K_i B^{-1}_{i,pre})^{μ_i} (b_i − K_i v_i);
    b_{i−1} = I_i^{i−1} (b_i − K_i v_i);                  /* Residual restriction */
    v_{i−1} = AlgV(b_{i−1}, i − 1, 0);                    /* Recursivity */
    v_i = v_i + I_{i−1}^{i} v_{i−1};                      /* Correction interpolation */
    Smooth ν_i times K_i v_i = b_i :                      /* Post-smoothing */
    (b_i − K_i v_i) = (I_i − K_i B^{-1}_{i,post})^{ν_i} (b_i − K_i v_i);
end
```

In this paper we consider multilevel Schwarz preconditioners with coarsest correction computed as

$$\mathbf{v}_0 = \mathbf{B}_0^{-1}\mathbf{b}_0,$$

where $\mathbf{B}_0^{-1}$ might denote either a Schwarz preconditioner or an exact solver.

Then, as iterative methods for (2.1), with $\mathbf{r}^{(\ell)}$ denoting the residual at iteration $\ell = 0, 1, 2, \ldots$, such multilevel Schwarz procedures can be described in the case $L = 2$ as

$$\mathbf{r}^{(\ell+1)} = (\mathbf{I}_1 - \mathbf{K}_1\mathbf{B}_{1,\text{post}}^{-1})^{\nu_1}(\mathbf{I}_1 - \mathbf{K}_1\mathcal{I}_0^1\mathbf{B}_0^{-1}\mathcal{I}_1^0)(\mathbf{I}_1 - \mathbf{K}_1\mathbf{B}_{1,\text{pre}}^{-1})^{\mu_1}\mathbf{r}^{(\ell)}. \tag{2.2}$$

In the case of $L = 3$ and the usual choices of $\mathcal{I}_i^{i-1} = \mathbf{R}_i$ and $\mathcal{I}_{i-1}^i = \mathbf{R}_i^T$, for some $N_{i-1} \times N_i$ matrix $\mathbf{R}_i$, a graphical representation is given in Figure 2.1.

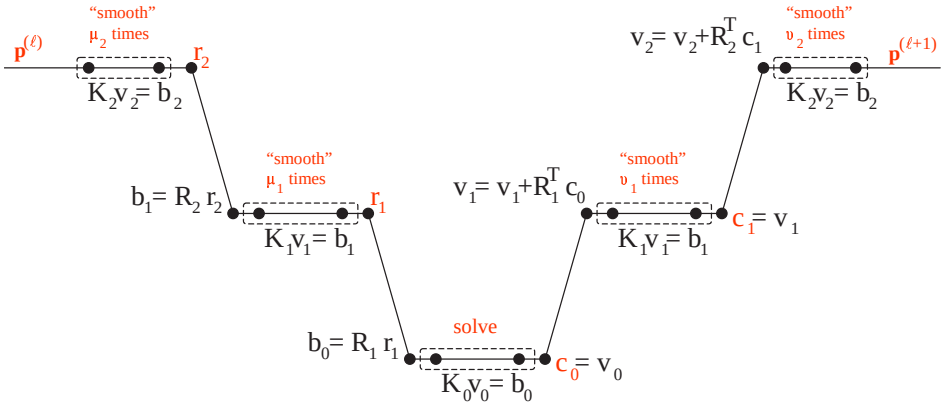


FIG. 2.1. *Graphical representation of a three-level MG V-cycle method:* $\mathbf{r}_1$ *and* $\mathbf{r}_2$ *denote residuals,* $\mathbf{c}_0$ *and* $\mathbf{c}_1$ *denote corrections.*

When classic Schwarz preconditioners are applied to symmetric positive definite systems arising from the discretization of elliptical problems defined in $H_0^1(\Omega)$, the condition number κ of the preconditioned system satisfies $\kappa \leqslant C\,(1 + H/\delta)\,/H^2$ for one-level methods and $\kappa \leqslant C\,(1 + H/\delta)$ for two-level methods, where C is independent of h, H and δ. The factor $1/H^2$, associated to the number of subdomains on the fine level, relates itself to the increase on the number of iterations (needed for the exchange of information among distant subdomains) with the increase in the total number of subdomains. The use of a coarse level helps the exchange of information. The necessity of information exchange among distant domain regions can be understood through the expression of the solutions of elliptic problems in terms of Green's functions: although the solution value at a point strongly depends on surrounding values, there is weaker dependence w.r.t. the entire domain [29]. Regarding the application of two-level methods to indefinite model problems, the study in [10] suggests that the coarse mesh needs to be sufficiently fine for the two-level Schwarz preconditioner to perform well.

Theoretically, however, these results may not be directly applied to the case of symmetric indefinite KKT Jacobians. Let $\bar{\ell}$ be the average number of linear iterations per Newton step. We then look for the following more general properties when applying a two-level preconditioner:

$$\text{For fixed } h \text{ and } H/\delta,\ \bar{\ell} \text{ is not very sensitive to } H \text{ decreasing,} \tag{2.3}$$

$$\text{For fixed } H \text{ and } \delta,\ \bar{\ell} \text{ is not very sensitive to the mesh refinement.} \tag{2.4}$$

Also, it might happen that a presumed "enhancement" of the preconditioner, aiming to cluster the eigenvalues around 1, eventually results in negative eigenvalues getting

too close to 0. Although there is no necessary relationship between the eigenvalue distribution and ill-conditioning for general matrices [24], this process might qualitatively explain an eventual degradation of the linear convergence before it finally gets better [12, page 199]. So,

> presumed "enhancements" of the preconditioner, e.g. increase of H or δ, eventually result in a worse preconditioned Krylov convergence. (2.5)

We report such temporary degradation in some of our numerical experiments in Section 4.

3. Pollution removing interpolation. The motivation of including, additively or multiplicatively, coarse preconditioners to an existing fine mesh preconditioner is to make the overall preconditioner scalable w.r.t. the number of processors, or the number of subdomains. In most cases, the addition of coarse preconditioners reduces the total number of iterations, as expected. However, for some classes of important problems, the unexpected happens: the number of iterations increases when a coarse preconditioner is involved. We refer to this phenomenon as coarse mesh pollution. To figure out the source of the pollution for a general problem is a highly nontrivial task. For the constrained optimization problems being considered we have discovered that the pollution is due to the coarse-to-fine mesh interpolation for one of the three types of variables (state variables, control variables and Lagrange multipliers) of the problem, namely the Lagrange multipliers [26]. Due to the sharp jumps often encountered on the multiplier values over those regions of Ω where constraints are greatly affecting the behavior of the optimized system, we introduce a new interpolation for these particular components. Such modified interpolation constitutes, indeed, a key procedure in our proposed multilevel preconditioner. Although the evidence of this discontinuity property of Lagrange multipliers is just empirical in this paper, it is consistent with their interpretation (1.2): the value of a Lagrange multiplier at a mesh point gives the rate of change of the optimal objective function value w.r.t. to the respective constraint at that point.

In the case of the problem corresponding to Figure 3.1, for instance, an external force causes the fluid to move clockwise and the boundary consists of rigid slip walls (see Figure 4.1). The vertical walls greatly affect the overall vorticity throughout the domain, i.e., the value of the objective function, because they completely oppose the horizontal velocity component v_1. The values of λ_1 at the walls then reflect this situation. Similarly, λ_2 develops sharp jumps at the other two walls opposing v_2. In all our experiments the discontinuities are located only across the boundary and not around it, even for very fine meshes. Common coarse-to-fine interpolation techniques will then smooth the sharp jumps present in coarse solutions, with a more gradual change, from interior mesh points towards boundary mesh points, appearing in those fine cells (elements, volumes) located inside coarse boundary ones, as represented in Figure 3.2-*b*. That is, the good correction information provided by the coarse solution is lost with a common interpolation. We refer to the smoothed jump as "pollution", in contrast to the "clean" sharp jump that is expected at the fine level as well.

We therefore propose a *modified* coarse-to-fine interpolation procedure that is based on a general and simple "removal of the pollution". Let $\mathcal{I}_{i-1}^{i}$ denote, as before,

any unmodified interpolation operator, while

$$\boldsymbol{\mathcal{Z}}_i \text{ is the operator that zeros out, from a vector at level } i, \text{ the Lagrange multipliers at all those mesh points with equations that have a greater influence on the objective function.} \tag{3.1}$$

For the case of PDEs describing physical systems, the number of such points can be expected to be relatively small. Our modified interpolation is then expressed by

$$\mathcal{I}^i_{i-1,\text{modif}} = \mathcal{I}^i_{i-1} - \boldsymbol{\mathcal{Z}}_i \mathcal{I}^i_{i-1}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}). \tag{3.2}$$

This procedure removes the smoothed contributions due to the coarse discontinuities, maintaining, at the fine level, the sharp jumps originally present at the coarse level. It should be noted that the operator $\mathcal{I}^i_{i-1}$ is usually available for the coding of a multilevel procedure and, once the locations where the operator $\boldsymbol{\mathcal{Z}}_i$ needs to act over are known, the implementation of (3.2) is extremely easy and can be performed on *any* mesh in *any* dimension, with *any* number of components.

The Lagrange multipliers reflect an eventual "discontinuity" of the type of equations (or their physical dimensions) between equations in different regions of $\overline{\Omega}$. ¿From this point of view, it seems "natural" to apply different interpolations to the Lagrange multipliers depending on their location on $\overline{\Omega}$. In the case of the problems in Section 4, for instance, a clear difference exists between those equations in Ω and those on $\partial\Omega$, that is, for those problems the operator $\boldsymbol{\mathcal{Z}}_i$ zeros the Lagrange multiplier components located at the boundary. A schematic representation of (3.2) for such situation is then given in Figure 3.2 for the simpler case of a piecewise linear interpolation. However, the modified interpolation (3.1)-(3.2) is *not* at all restricted to situations where the discontinuities are located at the boundary. Figure 3.3-a gives an schematic example where Lagrange multiplier discontinuities are spread throughout different regions of the domain. With the specification of the region where the operator $\boldsymbol{\mathcal{Z}}_i$ acts over, given by Figure 3.3-b, the pollution (smoothed jumps) created by usual interpolation operators $\mathcal{I}^i_{i-1}$ will be removed in a way similar to the way exemplified in the simpler situation of Figure 3.2.

Whenever the shape of the Lagrange multipliers in the final solution of a PDE-constrained optimization problem, the eventual sharp jumps will tend to appear more on the first Newton steps and, at each Newton iteration, on the coarse corrections related to the first Krylov iterations. As both the Newton and Krylov iterations progress towards the respective solutions, the steps and corrections will naturally shrink in magnitude. Nonetheless, even when sharp jumps are not involved anymore, the modified interpolation can still be applied with no harm: the interpolation of an eventual smooth coarse correction with small values on the Lagrange multiplier components will still present small values at the fine level. This fact facilitates the programming of (3.1)-(3.2), since the application of the modified interpolation is not conditional.

In our tests we apply the modified interpolation only for the Lagrange multiplier components of coarse corrections, while the optimization variables continue to be interpolated with the unmodified process $\mathcal{I}^i_{i-1}$. Also, the restriction process remains $\mathcal{I}^{i-1}_i$ for all variables, i.e., (2.2) becomes

$$\mathbf{r}^{(\ell+1)} = (\mathbf{I}_1 - \mathbf{K}_1\mathbf{B}^{-1}_{1,\text{post}})^{\nu_1}(\mathbf{I}_1 - \mathbf{K}_1\mathcal{I}^1_{0,\text{modif}}\mathbf{B}^{-1}_0\mathcal{I}^0_1)(\mathbf{I}_1 - \mathbf{K}_1\mathbf{B}^{-1}_{1,\text{pre}})^{\mu_1}\mathbf{r}^{(\ell)}.$$

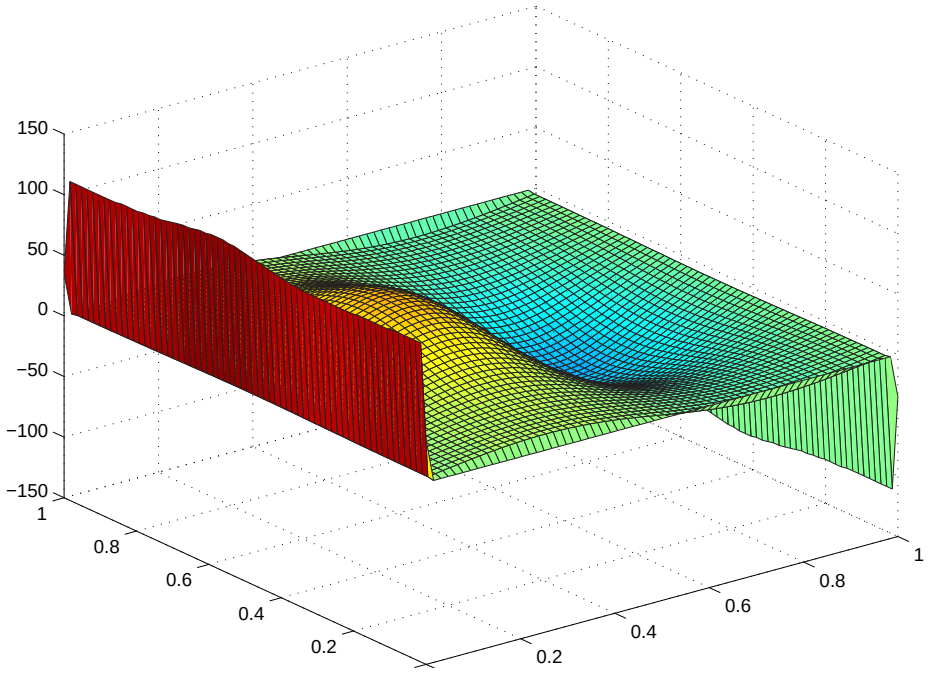


FIG. 3.1. *Lagrange multiplier* $\boldsymbol{\lambda}_1$ *related to the horizontal velocity* $\mathbf{v}_1$ *of a steady-state flow inside a unit square box with rigid slip walls.*

Two important aspects of our proposed modified interpolation operator should be highlighted before we proceed to the presentation of numerical experiments and results in the next section. First, the operator (3.2) is nonlinear, even for linear operators $\mathcal{I}_{i-1}^{i}$. Consequently, the multilevel preconditioner will become nonlinear as well. The linearity of preconditioners is usually theoretically desired because one can then guarantee the same solution when going from the original system (2.1) to the preconditioned one. In our tests, however, the same solution was obtained with both linear and nonlinear preconditioners. Second, the application of the modification process to an already modified interpolation operator gives the same modified operator: denoting

$$\mathcal{I}_{i-1,\text{modif},\text{modif}}^{i} = \mathcal{I}_{i-1,\text{modif}}^{i} - \boldsymbol{\mathcal{Z}}_i \mathcal{I}_{i-1,\text{modif}}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}),$$

we have

$$\begin{aligned}
\boldsymbol{\mathcal{Z}}_i \mathcal{I}_{i-1,\text{modif}}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) &= \boldsymbol{\mathcal{Z}}_i \left[\mathcal{I}_{i-1}^{i} - \boldsymbol{\mathcal{Z}}_i \mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1})\right](\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) \\
&= \boldsymbol{\mathcal{Z}}_i \left[\mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) - \boldsymbol{\mathcal{Z}}_i \mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1})^2\right] \\
&= \boldsymbol{\mathcal{Z}}_i \left[\mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) - \boldsymbol{\mathcal{Z}}_i \mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1})\right] \\
&= \boldsymbol{\mathcal{Z}}_i(\mathbf{I}_i - \boldsymbol{\mathcal{Z}}_i)\mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) \\
&= (\boldsymbol{\mathcal{Z}}_i - \boldsymbol{\mathcal{Z}}_i^2)\mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}) \\
&= (\boldsymbol{\mathcal{Z}}_i - \boldsymbol{\mathcal{Z}}_i)\mathcal{I}_{i-1}^{i}(\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1}),
\end{aligned}$$

the null operator. This result might be important for eventual theoretical demonstrations of preconditioning properties.

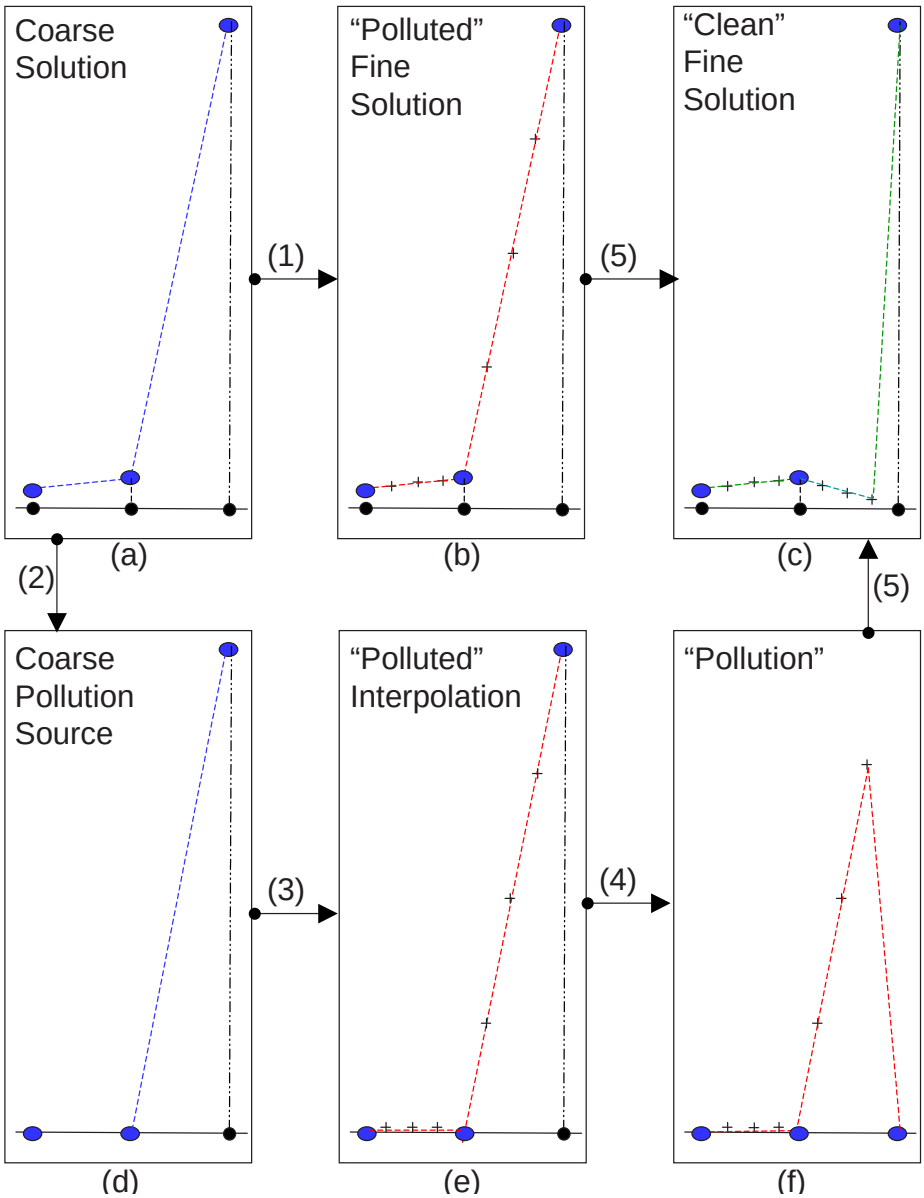


FIG. 3.2. *Representation of the modified coarse-to-fine interpolation* (3.2)*, with (a) input* $\boldsymbol{\varphi}_{i-1}$ *and (c) output* $\boldsymbol{\varphi}_i$*. The five steps are: (1) interpolation* $\mathcal{I}_{i-1}^{i}\boldsymbol{\varphi}_{i-1}$*, (2) coarse jump values* $\tilde{\boldsymbol{\varphi}}_{i-1} = (\mathbf{I}_{i-1} - \boldsymbol{\mathcal{Z}}_{i-1})\boldsymbol{\varphi}_{i-1}$*, (3) polluted* $\tilde{\boldsymbol{\varphi}}_i = \mathcal{I}_{i-1}^{i}\tilde{\boldsymbol{\varphi}}_{i-1}$*, (4) pollution isolation* $\boldsymbol{\mathcal{Z}}_i\tilde{\boldsymbol{\varphi}}_i$*, (5) pollution removal* $\boldsymbol{\varphi}_i = \mathcal{I}_{i-1}^{i}\boldsymbol{\varphi}_{i-1} - \boldsymbol{\mathcal{Z}}_i\tilde{\boldsymbol{\varphi}}_i$*.*

4. Numerical experiments. Our numerical experiments in this paper focus on optimal control problems, where the optimization space in (1.1) is generally given by $\mathbf{W}=\mathbf{S}\times\mathbf{U}$, with $\mathbf{S}$ being the state space and $\mathbf{U}$ the control space. Upon discretization, one has $n=n_s+n_u$, where n_s (n_u) is the number of discrete state (control) variables. More specifically, we treat the boundary control of two-dimensional

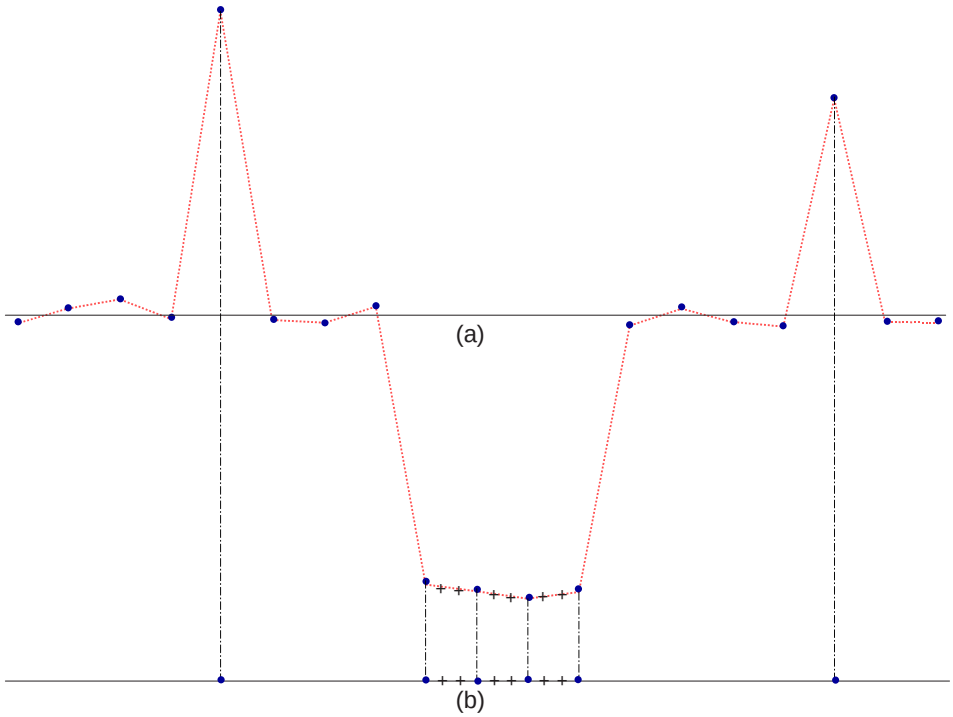


FIG. 3.3. *(a) A schematic example of a possible distribution of Lagrange multiplier components throughout a coarse mesh at level $i-1$. (b) The corresponding coarse (and fine) mesh points where the operator $\boldsymbol{\mathcal{Z}}_{i-1}$ (and $\boldsymbol{\mathcal{Z}}_i$) would act over. The plus sign refers to the fine mesh points where $\boldsymbol{\mathcal{Z}}_i$ would act over as well.*

steady-state incompressible Navier-Stokes equations in the velocity-vorticity formulation: $\mathbf{v} = (v_1, v_2)$ is the velocity and ω is the vorticity. Let $\Omega \subset \mathbb{R}^2$ be an open and bounded smooth domain, $\Gamma = \partial\Omega$ its boundary, $\boldsymbol{\nu}$ the unit outward normal vector along Γ and $\mathbf{f}$ a given external force defined in Ω. Let $L^2(\Omega)$ and $L^2(\Gamma_u)$ be the spaces of square Lebesgue integrable real functions in Ω and $\Gamma_u \subset \Gamma$ respectively. The control problems consist on finding $(\mathbf{s}, \mathbf{u}) = (v_1, v_2, \omega, u_1, u_2) \in \mathbf{S} \times \mathbf{U} = L^2(\Omega)^3 \times L^2(\Gamma_u)^2$ such that the minimization

$$\min_{(\mathbf{s},\mathbf{u})\in\mathbf{S}\times\mathbf{U}} \mathcal{F}(\mathbf{s}, \mathbf{u}) = \frac{1}{2}\int_\Omega \omega^2 \, d\Omega + \frac{c}{2}\int_{\Gamma_u} \|\mathbf{u}\|_2^2 \, d\Gamma_u \tag{4.1}$$

is achieved subject to the constraints

$$\begin{cases} -\Delta v_1 - \dfrac{\partial \omega}{\partial x_2} & = & 0 & \text{in } \Omega, \\ -\Delta v_2 + \dfrac{\partial \omega}{\partial x_1} & = & 0 & \text{in } \Omega, \\ -\Delta \omega + Re\ v_1 \dfrac{\partial \omega}{\partial x_1} + Re\ v_2 \dfrac{\partial \omega}{\partial x_2} - Re\ \text{curl } \mathbf{f} & = & 0 & \text{in } \Omega, \\ v_i - v_{D,i} & = & 0 & \text{on } \Gamma_{D,i}, \quad i = 1,2, \\ \dfrac{\partial v_i}{\partial \boldsymbol{\nu}} - v_{N,i} & = & 0 & \text{on } \Gamma_{N,i}, \quad i = 1,2, \\ \mathbf{v} - \mathbf{u} & = & \mathbf{0} & \text{on } \Gamma_u, \\ \omega + \dfrac{\partial v_1}{\partial x_2} - \dfrac{\partial v_2}{\partial x_1} & = & 0 & \text{on } \Gamma, \\ \int_\Gamma \mathbf{v} \cdot \boldsymbol{\nu}\, d\Gamma & = & 0, \end{cases} \tag{4.2}$$

where curl $\mathbf{f} = -\partial f_1/\partial x_2 + \partial f_2/\partial x_1$ and, for $i = 1, 2$, $\Gamma = \Gamma_{D,i} \cup \Gamma_{N,i} \cup \Gamma_u$ and $\Gamma_{D,i}$ ($\Gamma_{N,i}$) is the part of the boundary where the velocity component v_i is specified through a Dirichlet (Neumann) condition with a prescribed velocity $v_{D,i}$ ($v_{N,i}$). The parameter $c > 0$ is used to adjust the relative importance of the control norms in achieving the minimization, so indirectly constraining their sizes. The physical objective in (4.1)-(4.2) is the minimization of turbulence [18] The last constraint is necessary for the consistency with the physical law of mass conservation, making $m \neq n_s$ and causing the complexity of the parallel finite difference approximation of Jacobians to increase, since non-adjacent mesh points become coupled by the integral. An alternative formulation, that compromises between the physical law of mass conservation and the complex computation of Jacobians, eliminates the integral constraint but adds to the objective function the term $\tilde{c}\left[\int_\Gamma \mathbf{v} \cdot \boldsymbol{\nu}\, d\Gamma\right]^2/2$, with $\tilde{c} \gg 1$ [4]. We restrict our numerical experiments to *tangential* boundary control problems, i.e., $\mathbf{u} \cdot \boldsymbol{\nu} = 0$ on Γ_u and the velocity $\mathbf{v}$ is assumed to guarantee mass conservation along $\Gamma \setminus \Gamma_u$, so that $m = n_s$.

We consider rectangular domains $\Omega = (0, L_1) \times (0, L_2)$, $L_2 \leqslant L_1$. We define $E_{1,a} = \{(x_1, x_2) \in E_1 : 0 < x_1 \leqslant L_2\}$, $E_{1,b} = E_1 \setminus E_{1,a}$, $E_{4,a} = \{(x_1, x_2) \in E_4 : \frac{L_2}{2} \leqslant x_2 < L_2\}$ and $E_{4,b} = E_4 \setminus E_{4,a}$. Two flow problems are considered: cavity and backward-facing step. In each case we consider both simulation problems and tangential boundary control problems. In the descriptions below we only define Γ_u and the velocity boundary conditions on $\Gamma \setminus \Gamma_u$, since all control problems seek the minimization of the objective function (4.1) with $c = 10^{-2}$ and have in common the three PDEs in Ω, the ω boundary condition on Γ and the tangential boundary control on Γ_u. In the case of *cavity* flows, $L_1 = L_2 = 1$, $\Gamma_u = \Gamma$ and $\mathbf{f} = (f_1, f_2) = \big(-\sin^2(\pi x_1)\cos(\pi x_2)\sin^2(\pi x_2), \sin^2(\pi x_2)\cos(\pi x_1)\sin^2(\pi x_1)\big)$. In the case of *backward-facing step* flows, $L_1 = 6$, $L_2 = 1$, $\Gamma_u = E_{4,b} \cup \{C_1\} \cup E_{1,a}$ and $\mathbf{f} = \mathbf{0}$. The velocity boundary conditions on $\Gamma \setminus \Gamma_u$ are $v_1 = 0$ on $E_{1,b} \cup \overline{E_3}$, $v_1 = 8(1 - x_2)(x_2 - 1/2)$ on $E_{4,a}$, $v_1 = x_2(1 - x_2)$ on $\overline{E_2}$ and $v_2 = 0$. All corresponding simulation problems, used for comparison with control problems, impose $\mathbf{v} \cdot \boldsymbol{\nu} = 0$ and $\partial \mathbf{v}/\partial \boldsymbol{\nu} = 0$ on Γ_u, i.e., Γ_u becomes a (set of) slip rigid wall(s).

4.1. Details of numerical approaches. For discretization we use a five-point second order finite difference method on a uniform nonstaggered mesh. We divide the horizontal edges on $N_{1,\text{div}}$ equally spacing subintervals and the vertical edges on $N_{2,\text{div}}$ also equally spacing subintervals, generating a rectangular grid with a total of $N_g = (1 + N_{1,\text{div}})(1 + N_{2,\text{div}})$ points. We denote $h_1 = L_1/N_{1,\text{div}}$ and $h_2 = L_2/N_{2,\text{div}}$.

Each grid point is assigned an integer index k and denoted $\mathbf{x}_k = (x_{1,k}, x_{2,k}),\ k \in K = \{k \in \mathbb{N},\ 0 \leqslant k < N_g\}$. K_b is the set of indexes related to the $N_b = 2(N_{1,\text{div}} + N_{2,\text{div}})$ grid points located at the boundary, l_k is the elementary boundary length surrounding a point $\mathbf{x}_k,\ k \in K_b$, and a_k is the elementary area surrounding a point $\mathbf{x}_k,\ k \in K$, that is:

$$l_k = h_1,\ a_k = \frac{h_1 h_2}{2} \text{ for points on horizontal edges,}$$

$$l_k = h_2,\ a_k = \frac{h_1 h_2}{2} \text{ for points on vertical edges,}$$

$$l_k = \frac{h_1 + h_2}{2},\ a_k = \frac{h_1 h_2}{4} \text{ for points on corners,}$$

$$a_k = h_1 h_2 \text{ for } k \in K \setminus K_b.$$

The total number of discrete state variables is $n_s = 3N_g$. For code implementation convenience, both discrete control components are defined everywhere in the domain, i.e., the total number of discrete control variables is $n_u = 2N_g$. There is no problem with such approach since all control components not used as Dirichlet controls (i.e., not appearing in the constraints) are automatically forced to zero in the optimality system. The discrete state space is $\mathbf{S}_h = \mathbb{R}^{n_s}$ and the discrete control space is $\mathbf{U}_h = \mathbb{R}^{n_u}$. The discretized version of (4.1) then reads

$$\min_{(\mathbf{s},\mathbf{u}) \in \mathbf{S}_h \times \mathbf{U}_h} \mathcal{F}_h(\mathbf{s}, \mathbf{u}) = \frac{1}{2} \sum_{k \in K} \omega_k^2 a_k + \frac{c}{2} \sum_{k \in K_b} \|\mathbf{u}_k\|_2^2 l_k + \frac{c}{2} \sum_{k \in K \setminus K_b} \|\mathbf{u}_k\|_2^2 a_k.$$

All derivative terms in (4.2) are discretized with a second order scheme, including the ω boundary condition. For parallel performance reasons, only mesh points adjacent to the boundary are used by applying the definition $\omega = -\partial v_1/\partial x_2 + \partial v_2/\partial x_1$ at these points as well [25].

In order to form the algebraic system of nonlinear discretized equations, we need to order the unknowns and the corresponding functions. The unknowns are ordered mesh point by mesh point, in contrast to physical variable by physical variable as usually required by other methods. At each mesh point the unknowns are ordered as $v_1, v_2, \omega, u_1, u_2, \lambda_1, \lambda_2$ and λ_3. The mesh points are ordered subdomain by subdomain, for the purpose of parallel processing. The ordering of the subdomains is not important since we use, at each level other than the coarsest, additive methods whose performance has nothing to do with the subdomain ordering. In order to avoid pivoting during the sparse LU method (used in our experiments), the corresponding functions are ordered as $\nabla_{\lambda_1}\mathcal{L}, \nabla_{\lambda_2}\mathcal{L}, \nabla_{\lambda_3}\mathcal{L}$, $\nabla_{u_1}\mathcal{L}, \nabla_{u_2}\mathcal{L}$, $\nabla_{v_1}\mathcal{L}, \nabla_{v_2}\mathcal{L}$ and $\nabla_{\omega}\mathcal{L}$. Because the orderings for the unknowns and for the function components are different, the Jacobian matrix becomes nonsymmetric and so we use GMRES [27].

The Jacobian matrix is constructed approximately using a multi-colored central finite difference method [11] with step size 10^{-5}. In problem (4.1)-(4.2), since there is no variable with power greater than two, central finite difference approximations of the Jacobian are exact up to roundoff errors. To solve the Jacobian systems we use restarted GMRES with an absolute (relative) tolerance equal to 10^{-8} (10^{-4}), a restart parameter equal to 90 and a maximum number of iterations equal to 5,000. When using left preconditioning, the GMRES tolerances are checked over preconditioned

residuals. Regarding the one-level additive Schwarz preconditioner, the number of subdomains is equal to the number of processors, and the extended subdomain problems have zero Dirichlet interior boundary conditions and are solved with sparse LU. All subdomains Ω_j and $\Omega_j^{'}$ are rectangular and made up of integral number of mesh cells. The computation of coarse matrices at each Newton iteration proceeds recursively from the finest coarse level until the coarsest level by simply first injecting the finer iterate and then computing the Jacobian. We use nested meshes, the redundant LU solver at the coarsest level (all processors construct and solve a full coarsest linear system by sending their local portions of the Jacobian and of the right-hand side to all other processors) and the interpolation is piecewise linear. Richardson smoothers are all used with damping factor equal to 1. Line search is performed with augmented Lagrangian merit function and cubic backtracking. For Newton iterations, the maximum allowed number is 100 and the absolute (relative) stopping tolerance is 10^{-6} (10^{-10}). Simulation problems are solved with Newton-Krylov-Schwarz [7]. We do not use Reynolds continuation in any of the algorithms.

If not stated otherwise, the modified interpolation process explained in Section 3 is always used in multilevel tests.

4.2. Results of numerical experiments. We have performed tests on a cluster of Linux PCs and developed our parallel object-oriented software using the C++ programming language and the Portable, Extensible Toolkit for Scientific Computing (PETSc) from Argonne National Laboratory [1]. Since the cluster network is relatively slow and is shared with other processes, and since the redundant LU solver used at the coarsest level is not scalable in time w.r.t. the number of processors, our analysis focuses on the number of Newton and Krylov iterations, not on computing times.

4.2.1. Cavity Flows. Figure 4.1-*b* shows the controlled velocity field for the cavity flow with $Re = 200$: although the fluid in the interior follows the clockwise direction imposed by the external force, the movement near the boundary is much less intense than in Figure 4.1-*a*. In fact, a zoom on the boundary velocity of Figure 4.1-*b* shows the control acting in the counter-clockwise way, making the integral of ω^2 on Ω to decrease from the value of 55.4 for the simulation with slip walls to the controlled value of 32.5. Figure 3.1 shows the sharp jump of the Lagrangian multiplier function λ_1 at the boundary. The other two Lagrangian multipliers behave similarly, no matter how fine the mesh is.

Tables 4.1-4.4 show results for one-level preconditioners. In Tables 4.1-4.2 we fix the mesh size in 280×280 and the preconditioning side to *left*. Then, for Re=200 and 300, respectively, we perform tests with 25 and 49 processors, varying the combinations of Schwarz preconditioner type (ASM, l-RAS or r-RAS) and relative overlap (1/8, 1/4 or 1/2). In Table 4.3, we do the same for $Re = 300$, only fixing the preconditioning side to *right*. Most of the results in these first three Tables are consistent with the behavior of one-level Schwarz preconditioners for positive-definite systems: the average number $\overline{\ell}$ of Krylov iterations per Newton iteration grows with the number N_p of processors and decreases as the overlap gets larger. Whenever some Newton step demanded $5,000$ Krylov iterations to be computed, the overall test was considered to fail. This was the case in the six tests not reported in Tables 4.2 and 4.3.

The results in these first three Tables suggest that the more robust combination is the *left l-RAS with* δ/H=1/2, which is then fixed for the next set of tests, shown in Table 4.4, with varying Re, mesh size and number of processors, with the number of processors going up to 100 now. However, even with the pretty large relative overlap $\delta/H = 1/2$ the method fails for $Re = 250$ with 100 processors.

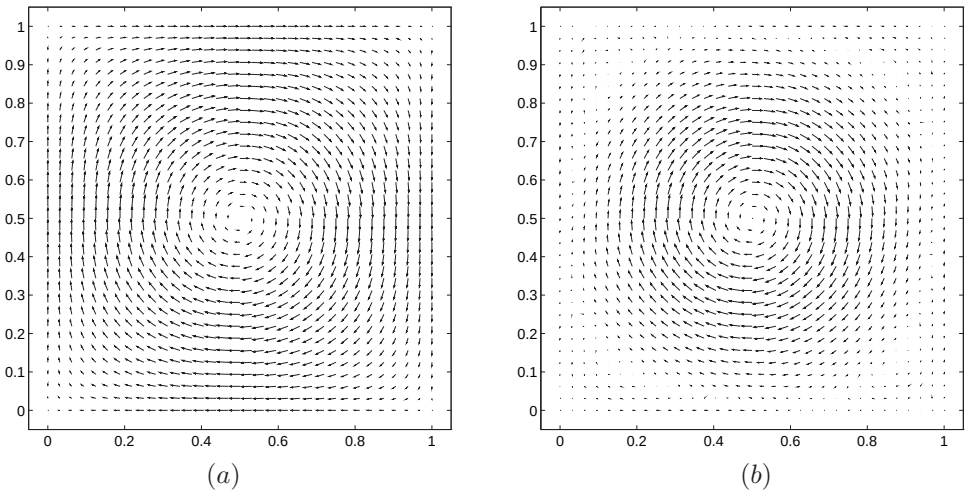


FIG. 4.1. *Computed velocity fields for the cavity flow problems with* $Re = 200$*: (a) simulation problem and (b) tangential boundary control problem. The Lagrange multiplier* λ_1 *corresponding to* v_1 *is shown in Figure 3.1.*

TABLE 4.1

One-level left *Schwarz preconditioner results for the cavity flow control problem with* Re**=200** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$*.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H		
		1/8	1/4	1/2
ASM	25	$k = 6$; $\overline{\ell} \approx \mathbf{110}$	$k = 7$; $\overline{\ell} \approx \mathbf{64}$	$k = 7$; $\overline{\ell} \approx \mathbf{60}$
	49	$k = 6$; $\overline{\ell} \approx \mathbf{196}$	$k = 6$; $\overline{\ell} \approx \mathbf{148}$	$k = 7$; $\overline{\ell} \approx \mathbf{142}$
l-RAS	25	$k = 7$; $\overline{\ell} \approx \mathbf{114}$	$k = 7$; $\overline{\ell} \approx \mathbf{67}$	$k = 8$; $\overline{\ell} \approx \mathbf{52}$
	49	$k = 6$; $\overline{\ell} \approx \mathbf{205}$	$k = 7$; $\overline{\ell} \approx \mathbf{158}$	$k = 7$; $\overline{\ell} \approx \mathbf{83}$
r-RAS	25	$k = 7$; $\overline{\ell} \approx \mathbf{113}$	$k = 7$; $\overline{\ell} \approx \mathbf{67}$	$k = 7$; $\overline{\ell} \approx \mathbf{50}$
	49	$k = 6$; $\overline{\ell} \approx \mathbf{238}$	$k = 7$; $\overline{\ell} \approx \mathbf{356}$	$k = 7$; $\overline{\ell} \approx \mathbf{540}$

TABLE 4.2

One-level left *Schwarz preconditioner results for the cavity flow control problem with* Re**=300** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$*.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H		
		1/8	1/4	1/2
ASM	25	$k = 11$; $\overline{\ell} \approx \mathbf{123}$	$k = 10$; $\overline{\ell} \approx \mathbf{84}$	$k = 11$; $\overline{\ell} \approx \mathbf{111}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{324}$	$k = -$; $\overline{\ell} \approx -$	$k = 11$; $\overline{\ell} \approx \mathbf{333}$
l-RAS	25	$k = 11$; $\overline{\ell} \approx \mathbf{165}$	$k = 10$; $\overline{\ell} \approx \mathbf{96}$	$k = 11$; $\overline{\ell} \approx \mathbf{74}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{454}$	$k = -$; $\overline{\ell} \approx -$	$k = 11$; $\overline{\ell} \approx \mathbf{216}$
r-RAS	25	$k = 11$; $\overline{\ell} \approx \mathbf{172}$	$k = 10$; $\overline{\ell} \approx \mathbf{86}$	$k = 11$; $\overline{\ell} \approx \mathbf{72}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{498}$	$k = -$; $\overline{\ell} \approx -$	$k = 10$; $\overline{\ell} \approx \mathbf{180}$

TABLE 4.3

One-level right *Schwarz preconditioner results for the cavity flow control problem with* Re**=300** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$*.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H		
		1/8	1/4	1/2
ASM	25	$k = 10$; $\overline{\ell} \approx \mathbf{184}$	$k = 10$; $\overline{\ell} \approx \mathbf{109}$	$k = 10$; $\overline{\ell} \approx \mathbf{108}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{413}$	$k = -$; $\overline{\ell} \approx -$	$k = 10$; $\overline{\ell} \approx \mathbf{313}$
l-RAS	25	$k = 10$; $\overline{\ell} \approx \mathbf{231}$	$k = 10$; $\overline{\ell} \approx \mathbf{136}$	$k = 10$; $\overline{\ell} \approx \mathbf{82}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{494}$	$k = -$; $\overline{\ell} \approx -$	$k = 10$; $\overline{\ell} \approx \mathbf{266}$
r-RAS	25	$k = 10$; $\overline{\ell} \approx \mathbf{201}$	$k = 10$; $\overline{\ell} \approx \mathbf{122}$	$k = 10$; $\overline{\ell} \approx \mathbf{67}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{646}$	$k = -$; $\overline{\ell} \approx -$	$k = 10$; $\overline{\ell} \approx \mathbf{202}$

TABLE 4.4

One-level left *l-RAS preconditioner results for the cavity flow control problem with relative overlap* $\delta/H = 1/2$*, for different combinations of Re, number* N_p *of processors and mesh size.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration. The number of variables is* $2,517,768$ *on the case of finest mesh.*

Re	N_p	Mesh		
		140×140	280×280	560×560
200	25	$k = 8$; $\overline{\ell} \approx \mathbf{46}$	$k = 8$; $\overline{\ell} \approx \mathbf{52}$	out of memory
	49	$k = 8$; $\overline{\ell} \approx \mathbf{73}$	$k = 7$; $\overline{\ell} \approx \mathbf{83}$	out of memory
	100	$k = 8$; $\overline{\ell} \approx \mathbf{247}$	$k = 7$; $\overline{\ell} \approx \mathbf{277}$	$k = 6$; $\overline{\ell} \approx \mathbf{205}$
250	25	$k = 12$; $\overline{\ell} \approx \mathbf{53}$	$k = 9$; $\overline{\ell} \approx \mathbf{57}$	not tested
	49	$k = 13$; $\overline{\ell} \approx \mathbf{112}$	$k = 9$; $\overline{\ell} \approx \mathbf{117}$	not tested
	100	$k = -$; $\ell^{(10)}$=5,000	$k = -$; $\ell^{(5)}$=5,000	not tested

Clearly, then, some stabilization is needed. This stabilization is achieved with the use of a coarse mesh and the modified interpolation process explained in Section 3. The results so far suggest that one can focus on RAS preconditioners on the Richardson smoothers and not use ASM. We however made some experiments with ASM preconditioned smoother and in all cases the preconditioned GMRES diverged, with either left or right preconditioning. We also tried to avoid tests with $\delta = H/2$ as well, in order to check if the two-level method can be used with smaller and larger overlaps.

In Table 4.5 we fix the mesh size in 280×280, the preconditioning side to *left* and, for Re=200 and 250, we test two-level Schwarz with 25 and 49 processors, varying the combinations of Schwarz preconditioner type (l-RAS or r-RAS) and relative overlap (1/8 or 1/4). The method with left preconditioning diverges in many cases. The divergences were caused by the fact that the true linear residuals at the Newton steps did not decrease as much as the preconditioned linear residuals used for the Krylov stopping criteria. The overall algorithm then failed after some Newton iterations, either by computing steps that did not provide descent directions or by computing step lengths too small.

In Tables 4.6-4.8 we do basically the same as in Table 4.5, but now we fix the preconditioning side to *right* and test up to $Re = 300$ and 100 processors. In the case of $Re = 300$ (Table 4.8) the method demanded too many iterations for convergence

with $\delta = H/8$ and so we report experiments with $\delta = H/2$.

It is interesting to observe in Tables 4.7 and 4.8 that, fixing the overlap, one of the RAS smoothers (l-RAS or r-RAS) causes the preconditioner to converge with a smaller average number of Krylov iterations for 25 subdomains, while the other RAS smoother contributes for a better average Krylov convergence in the case of 100 subdomains. That is, there is no concept such as a unique RAS version that is always the best smoother for all possible number of processors.

TABLE 4.5

Two-level left *Schwarz preconditioner results for the cavity flow control problem with a* 280×280 *mesh (*$631,688$ *variables), for different combinations of Re, number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

Re	$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H	
			1/8	1/4
200	l-RAS	25	$k = 8$; $\overline{\ell} \approx \mathbf{16}$	$k = -$; $\overline{\ell} \approx -$
		49	$k = 7$; $\overline{\ell} \approx \mathbf{16}$	$k = 8$; $\overline{\ell} \approx \mathbf{13}$
	r-RAS	25	$k = -$; $\overline{\ell} \approx -$	$k = 8$; $\overline{\ell} \approx \mathbf{19}$
		49	$k = 10$; $\overline{\ell} \approx \mathbf{32}$	$k = 8$; $\overline{\ell} \approx \mathbf{22}$
250	l-RAS	25	$k = -$; $\overline{\ell} \approx -$	$k = -$; $\overline{\ell} \approx -$
		49	$k = -$; $\overline{\ell} \approx -$	$k = 10$; $\overline{\ell} \approx \mathbf{18}$
	r-RAS	25	$k = 12$; $\overline{\ell} \approx \mathbf{24}$	$k = 12$; $\overline{\ell} \approx \mathbf{17}$
		49	$k = 20$; $\overline{\ell} \approx \mathbf{36}$	$k = 11$; $\overline{\ell} \approx \mathbf{24}$

TABLE 4.6

Two-level right *Schwarz preconditioner results for the cavity flow control problem with* Re**=200** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H	
		1/8	1/4
l-RAS	25	$k = 6$; $\overline{\ell} \approx \mathbf{20}$	$k = 6$; $\overline{\ell} \approx \mathbf{17}$
	49	$k = 6$; $\overline{\ell} \approx \mathbf{23}$	$k = 6$; $\overline{\ell} \approx \mathbf{17}$
	100	$k = 6$; $\overline{\ell} \approx \mathbf{28}$	$k = 6$; $\overline{\ell} \approx \mathbf{18}$
r-RAS	25	$k = 6$; $\overline{\ell} \approx \mathbf{102}$	$k = 6$; $\overline{\ell} \approx \mathbf{21}$
	49	$k = 6$; $\overline{\ell} \approx \mathbf{59}$	$k = 6$; $\overline{\ell} \approx \mathbf{25}$
	100	$k = 6$; $\overline{\ell} \approx \mathbf{32}$	$k = 6$; $\overline{\ell} \approx \mathbf{26}$

Based on the results of Tables 4.6-4.8, we perform the tests shown in Tables 4.9-4.11, where we fix the preconditioning side to *right* and vary Re, mesh size and number of processors. Clearly, the two-level method performed in a very robust way, solving problems for which the one-level version has previously failed. In all three Tables 4.9-4.11, the average number $\overline{\ell}$ of Krylov iterations per Newton iteration is kept stable in the tested range of mesh size and number of processors. The results are consistent with predictions (2.3)-(2.4): in each column of the tables, $\overline{\ell}$ is not very sensitive to the increase in the number N_p of processors (prediction (2.3)), and in each line of the tables, $\overline{\ell}$ is not too sensitive to mesh refinement (prediction (2.4)).

TABLE 4.7

Two-level right *Schwarz preconditioner results for the cavity flow control problem with* Re**=250** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H	
		1/8	1/4
l-RAS	25	$k = -$; $\ell^{(6)} > 2,518$	$k = 8$; $\overline{\ell} \approx \mathbf{87}$
	49	$k = -$; $\ell^{(7)} = 3,150$	$k = 8$; $\overline{\ell} \approx \mathbf{21}$
	100	$k = 8$; $\overline{\ell} \approx \mathbf{33}$	$k = 8$; $\overline{\ell} \approx \mathbf{23}$
r-RAS	25	$k = 8$; $\overline{\ell} \approx \mathbf{29}$	$k = 8$; $\overline{\ell} \approx \mathbf{22}$
	49	$k = 8$; $\overline{\ell} \approx \mathbf{106}$	$k = 8$; $\overline{\ell} \approx \mathbf{28}$
	100	$k = 8$; $\overline{\ell} \approx \mathbf{251}$	$k = 8$; $\overline{\ell} \approx \mathbf{86}$

TABLE 4.8

Two-level right *Schwarz preconditioner results for the cavity flow control problem with* Re**=300** *and* 280×280 *mesh (*$631,688$ *variables), for different combinations of number* N_p *of processors, relative overlap* δ/H *and Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

$\mathbf{B}^{-1}$	N_p	Relative overlap δ/H	
		1/4	1/2
l-RAS	25	$k = -$; $\ell^{(6)} > \mathbf{4,290}$	$k = 10$; $\overline{\ell} \approx \mathbf{43}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{73}$	$k = -$; $\ell^{(6)} = \mathbf{1,794}$
	100	$k = 10$; $\overline{\ell} \approx \mathbf{26}$	$k = 10$; $\overline{\ell} \approx \mathbf{25}$
r-RAS	25	$k = 10$; $\overline{\ell} \approx \mathbf{23}$	$k = 10$; $\overline{\ell} \approx \mathbf{21}$
	49	$k = 10$; $\overline{\ell} \approx \mathbf{48}$	$k = 10$; $\overline{\ell} \approx \mathbf{21}$
	100	$k = 10$; $\overline{\ell} \approx \mathbf{52}$	$k = 10$; $\overline{\ell} \approx \mathbf{29}$

As intuitively expected, the Jacobian ill-conditioning gets worse as we increase Re and refine the mesh, demanding Schwarz preconditioning with bigger overlap and/or more smoothing iterations.

Many of the results reported so far, for both one-level and two-level Schwarz preconditioners, might be related to comment (2.5). Namely, the fact that the average number of Krylov iterations per Newton step gets eventually worse with an increase in the size H of subdomains or in the overlap δ. Such behavior is "strange" if we consider the literature results on the application of Schwarz preconditioners to the solution of positive definite systems. In the case of one-level tests, examples of strange results exist in Table 4.2, with 49 processors, and in Table 4.3, also with 49 processors. In the case of two-level tests, examples of strange results exist in Table 4.5, with $Re = 200$ and with $Re = 250$, l-RAS preconditioner and relative overlap $\delta/H = 1/4$, in Table 4.7, with l-RAS preconditioner and relative overlap $\delta/H = 1/8$, and in Table 4.8, with l-RAS preconditioner.

Figures 4.2-a and 4.2-b clearly show the stabilization on the average number of Krylov iterations provided by the two-level preconditioner with modified interpolation. The one-level preconditioner fails with 100 processors for $Re = 250$ and $Re = 300$.

Table 4.12 shows the efficacy of the modified interpolation process, which performs much better than the unmodified one, causing the two-level preconditioner to

TABLE 4.9

Two-level right *Schwarz preconditioner results for the cavity flow control problem with Re=**200**, Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother and relative overlap* δ/H*, for different combinations of number* N_p *of processors and mesh size.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration. The number of variables is* $2,517,768$ *on the case of finest mesh.*

N_p $\left[\mathbf{B}^{-1}\right]$ $\left(\frac{\delta}{H}\right)$	Mesh		
	140×140	280×280	560×560
25 [l-RAS] $\left(\frac{1}{4}\right)$	$k = 7$; $\overline{\ell} \approx \mathbf{10}$	$k = 6$; $\overline{\ell} \approx \mathbf{17}$	not tested
49 [l-RAS] $\left(\frac{1}{4}\right)$	$k = 7$; $\overline{\ell} \approx \mathbf{10}$	$k = 6$; $\overline{\ell} \approx \mathbf{17}$	not tested
100 [l-RAS] $\left(\frac{1}{4}\right)$	$k = 7$; $\overline{\ell} \approx \mathbf{11}$	$k = 6$; $\overline{\ell} \approx \mathbf{18}$	$k = 6$; $\overline{\ell} \approx \mathbf{22}$

TABLE 4.10

Two-level right *Schwarz preconditioner results for the cavity flow control problem with Re=**250**, Schwarz preconditioner* $\mathbf{B}^{-1}$ *for the Richardson smoother and relative overlap* δ/H*, for different combinations of number* N_p *of processors and mesh size.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration. The number of variables is* $2,517,768$ *on the case of finest mesh.*

N_p $\left[\mathbf{B}^{-1}\right]$ $\left(\frac{\delta}{H}\right)$	Mesh		
	140×140	280×280	560×560
25 [r-RAS] $\left(\frac{1}{4}\right)$	$k = 12$; $\overline{\ell} \approx \mathbf{17}$	$k = 8$; $\overline{\ell} \approx \mathbf{22}$	not tested
49 [l-RAS] $\left(\frac{1}{4}\right)$	$k = 12$; $\overline{\ell} \approx \mathbf{14}$	$k = 8$; $\overline{\ell} \approx \mathbf{21}$	not tested
100 [l-RAS] $\left(\frac{1}{4}\right)$	$k = 12$; $\overline{\ell} \approx \mathbf{14}$	$k = 8$; $\overline{\ell} \approx \mathbf{23}$	$k = 7$; $\overline{\ell} \approx \mathbf{31}$

TABLE 4.11

Two-level right *Schwarz preconditioner results, for the cavity flow control problem with Re=300 and a* 70×70 *coarse mesh, for different situations of number* N_p *of processors and mesh size.* k *is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration. To each situation corresponds a combination of the number* σ *of Richardson iterations, the RAS preconditioner and the relative overlap* δ/H *used in the pre and post smoothers. The number of variables is* $2,517,768$ *in the case of finest mesh.*

N_p $\left(\frac{\delta}{H}\right)$	Mesh		
	140×140	280×280	560×560
25 $\left(\frac{1}{4}\right)$	$\sigma = 1$; r-RAS $k = 13$; $\overline{\boldsymbol{\ell}} = \mathbf{20}$	$\sigma = 1$; r-RAS $k = 10$; $\overline{\boldsymbol{\ell}} = \mathbf{23}$	– –
49 $\left(\frac{1}{2}\right)$	$\sigma = 1$; r-RAS $k = 13$; $\overline{\boldsymbol{\ell}} = \mathbf{18}$	$\sigma = 1$; r-RAS $k = 10$; $\overline{\boldsymbol{\ell}} = \mathbf{21}$	– –
100 $\left(\frac{1}{2}\right)$	$\sigma = 1$; l-RAS $k = 13$; $\overline{\boldsymbol{\ell}} = \mathbf{18}$	$\sigma = 1$; l-RAS $k = 10$; $\overline{\boldsymbol{\ell}} = \mathbf{25}$	$\sigma = 2$; r-RAS $k = 8$; $\overline{\boldsymbol{\ell}} = \mathbf{27}$

outperform the one-level preconditioner.

4.2.2. Backward-Facing Step Flows. The numerical behavior observed in these problems is very different than the one observed in the cavity flows of the previous section. The backward-facing step problems demand more Newton iterations, but the computation of each Newton step is easier. In fact, the one-level Schwarz preconditioner now works for problems with 100 processors for all three $Re = 200$, 250 and 300, the same happening for the two-level method with the polluted (unmodified) interpolation procedure. Also, similarly to the case of positive-definite systems, left

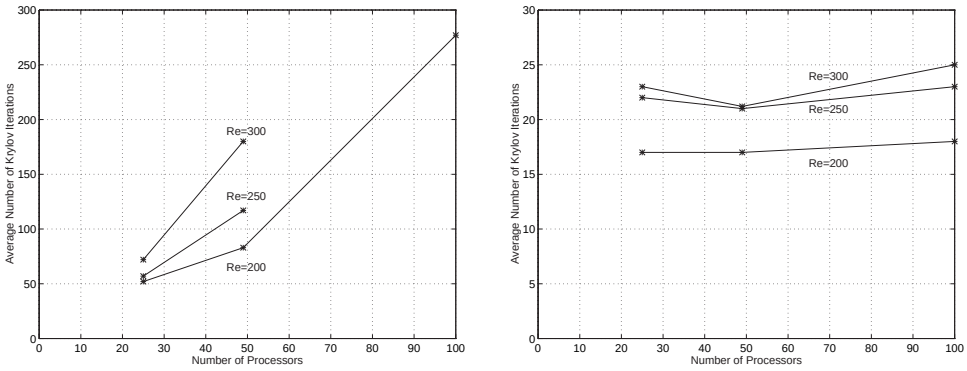


FIG. 4.2. *Cavity tangential boundary control problem results for (a) one-level and (b) two-level right Schwarz preconditioned with 49 processors, a* 280×280 *mesh (*$631,688$ *variables) and a* 70×70 *coarse mesh.*

TABLE 4.12

Right Schwarz preconditioner results for the cavity flow control problem with Re=250, a 280×280 *mesh (*$631,688$ *variables), 49 processors, relative overlap* $\delta/H = 1/4$ *and a* 70×70 *coarse mesh, for different combinations of number L of levels, piecewise linear interpolation type, number* σ *of pre and post smoother iterations, and RAS preconditioner. k is the total number of Newton iterations and* $\overline{\ell}$ *is the average number of Krylov iterations per Newton iteration.*

L	Linear Interpolation Type	σ	RAS preconditioner	
			l-RAS	r-RAS
1	–	–	$k = 8;\ \overline{\ell} = 336$	$k = 9;\ \overline{\ell} = 973$
2	Unmodified	1	$k = 8;\ \overline{\ell} = 1,110$	$k = 8;\ \overline{\ell} = 1,150$
2	Unmodified	2	$k = 8;\ \overline{\ell} = 356$	$k = 8;\ \overline{\ell} = 222$
2	Modified	1	$k = 8;\ \boldsymbol{\overline{\ell} = 21}$	$k = 8;\ \boldsymbol{\overline{\ell} = 28}$

preconditioner now works better than right preconditioner, as shown by Tables 4.13 and 4.14.

Nonetheless, the pollution free two-level RAS preconditioner was the best choice, as exemplified by Table 4.15, where the use of the modified interpolation decreases the average number of Krylov iterations by a further factor of two w.r.t. the unmodified interpolation. Figures 4.3-*a* and 4.3-*b* also show the stabilization on the average number of Krylov iterations provided by the preconditioner. Finally, as in the case of cavity flow control problems, Figure 4.3-b is consistent with prediction (2.3).

5. Conclusions. We have developed a parallel multilevel Schwarz preconditioner that has shown a robust performance when tested on some tangential boundary flow control problems. With such preconditioner we were able to extend the one-level Lagrange-Newton-Krylov-Schwarz presented in [25, 26] to a multilevel method.

In general, the success of Lagrange-Newton methods with line search rests on four tasks: the computation of the Jacobian, the computation of a Newton step with descent direction, the choice of a penalty parameter in the merit function, and the computation of the step length. Although we addressed all these issues in our work, our main contribution in this paper consists in the combination of a general multigrid V-cycle preconditioner with (1) RAS preconditioned Richardson smoothers

TABLE 4.13

Two-level left *Schwarz preconditioner results for the backward-facing step flow control problem with Re=**300**, r-RAS preconditioned Richardson smoother and relative overlap $\delta/H = 1/4$, for different combinations of number N_p of processors and mesh size. k is the total number of Newton iterations and $\overline{\ell}$ is the average number of Krylov iterations per Newton iteration. The number of variables is $1,780,232$ on the case of finest mesh.*

N_p	Mesh		
	288×48	576×96	$1,152 \times 192$
24	$k = 46$; $\overline{\ell} \approx \mathbf{4}$	$k = 36$; $\overline{\ell} \approx \mathbf{4}$	not tested
54	$k = 46$; $\overline{\ell} \approx \mathbf{4}$	$k = 36$; $\overline{\ell} \approx \mathbf{5}$	not tested
96	$k = 46$; $\overline{\ell} \approx \mathbf{5}$	$k = 36$; $\overline{\ell} \approx \mathbf{6}$	$k = 25$; $\overline{\ell} \approx \mathbf{7}$

TABLE 4.14

Two-level right *Schwarz preconditioner results for the backward-facing step flow control problem with Re=**300**, r-RAS preconditioned Richardson smoother and relative overlap $\delta/H = 1/4$, for different combinations of number N_p of processors and mesh size. k is the total number of Newton iterations and $\overline{\ell}$ is the average number of Krylov iterations per Newton iteration. The number of variables is $1,780,232$ on the case of finest mesh.*

N_p	Mesh		
	288×48	576×96	$1,152 \times 192$
24	$k = 46$; $\overline{\ell} \approx \mathbf{6}$	$k = 36$; $\overline{\ell} \approx \mathbf{7}$	not tested
54	$k = 46$; $\overline{\ell} \approx \mathbf{7}$	$k = 36$; $\overline{\ell} \approx \mathbf{8}$	not tested
96	$k = 46$; $\overline{\ell} \approx \mathbf{8}$	$k = 36$; $\overline{\ell} \approx \mathbf{10}$	$k = 26$; $\overline{\ell} \approx \mathbf{11}$

TABLE 4.15

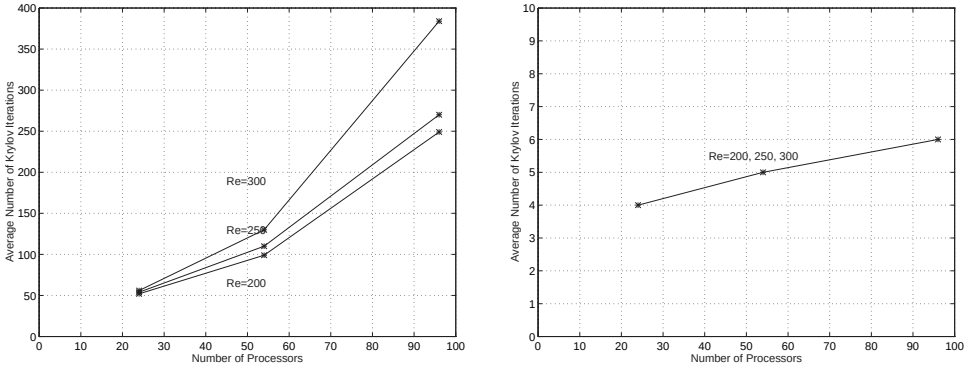
Left Schwarz preconditioner results for the backward-facing step flow control problem with Re=300, a 1152×192 mesh ($1,780,232$ variables), 96 processors, r-RAS with relative overlap $\delta/H = 1/4$ and a 144×24 coarse mesh, for different combinations of number L of levels, piecewise linear interpolation type and number σ of pre and post smoother iterations. k is the total number of Newton iterations and $\overline{\ell}$ is the average number of Krylov iterations per Newton iteration.

L	Linear interpolation type	σ	r-RAS preconditioner
1	–	–	$k = 26$; $\overline{\ell} = 404$
2	Unmodified	1	$k = 27$; $\overline{\ell} = 14$
2	Modified	1	$k = 25$; $\overline{\boldsymbol{\ell}} = \mathbf{7}$

and (2) a modified interpolation procedure that removes the pollution often generated by the application of common interpolation techniques to the Lagrange multipliers. Such combination is the key for the successful application of the two-level Schwarz preconditioner in our experiments and the consequent improvement over the one-level method, handling flow control problems with higher Reynolds number, finer meshes and more processors. Surprisingly, RAS preconditioners performed much better than the classical ones.

We expect multilevel pollution removing preconditioners to have a wide application in many scientific and engineering applications.

6. Acknowledgements. We would like to thank the PETSc team for helping us using the PETSc library, the Hemisphere cluster team at the University of Colorado at Boulder for their support and Professor Richard Byrd and two anonymous referees



(a) (b)

FIG. 4.3. *Backward facing-step tangential boundary control problems results for (a) one-level and (b) two-level left Schwarz preconditioner with 54 processors, a* 576×96 *mesh* $(447,752$ *variables) and a* 144×24 *coarse mesh.*

for their useful comments.

REFERENCES

[1] S. BALAY, K. BUSCHELMAN, W. D. GROPP, D. KAUSHIK, M. KNEPLEY, L. C. MCINNES, B. F. SMITH, AND H. ZHANG, *Portable, Extensible Toolkit for Scientific Computation (PETSc).* Homepage http://www.mcs.anl.gov/petsc, 2004.

[2] M. BENZI, G. H. GOLUB, AND J. LIESEN, *Numerical solution of saddle point problems*, Acta Numerica, 14 (2005), pp. 1–137.

[3] G. BIROS AND O. GHATTAS, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization, part I: The Krylov-Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.

[4] ———, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization, part II: The Lagrange-Newton solver and its application to optimal control of steady viscous flows*, SIAM J. Sci. Comput., 27 (2005), pp. 714–739.

[5] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, *A Multigrid Tutorial*, SIAM, Second ed., 2000.

[6] X.-C. CAI, M. DRYJA, AND M. SARKIS, *Restricted additive Schwarz preconditioners with harmonic overlap for symmetric positive definite linear systems*, SIAM J. Numer. Anal., 41 (2003), pp. 1209–1231.

[7] X.-C. CAI, W. D. GROPP, D. E. KEYES, R. G. MELVIN, AND D. P. YOUNG, *Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation*, SIAM J. Sci. Comput., 19 (1998), pp. 246–265.

[8] X.-C. CAI AND D. E. KEYES, *Nonlinearly preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 24 (2002), pp. 183–200.

[9] X.-C. CAI AND M. SARKIS, *A restricted additive Schwarz preconditioner for general sparse linear systems*, SIAM J. Sci. Comput., 21 (1999), pp. 792–797.

[10] X.-C. CAI AND O. B. WIDLUND, *Domain decomposition algorithms for indefinite elliptic problems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 243–258.

[11] T. F. COLEMAN AND J. J. MORÉ, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

[12] J. J. DONGARRA, I. S. DUFF, D. C. SORENSEN, AND H. A. VAN DER VORST, *Numerical Linear Algebra for High-Performance Computers*, SIAM, 1998.

[13] M. DRYJA AND W. HACKBUSCH, *On the nonlinear domain decomposition method*, BIT, (1997), pp. 296–311.

[14] M. DRYJA AND O. WIDLUND, *Domain decomposition algorithms with small overlap*, SIAM J. Sci. Comp., 15 (1994), pp. 604–620.

[15] S. C. EISENSTAT AND H. F. WALKER, *Globally convergent inexact Newton method*, SIAM J. Optim., 4 (1994), pp. 393–422.

[16] A. V. Fiacco, *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, vol. 68 of Mathematics in Science and Engineering, Academic Press, 1983.
[17] A. Frommer and D. B. Szyld, *An algebraic convergence theory for restricted additive Schwarz methods using weighted max norms*, SIAM J. Numer. Anal., 39 (2001), pp. 463–479.
[18] M. D. Gunzburger, *Perspectives in Flow Control and Optimization*, SIAM, First ed., 2003.
[19] M. Heinkenschloss and H. Nguyen, *Domain decomposition preconditioners for linear quadratic elliptic optimal control problems*, Tech. Report TR04-20, Department of Computational and Applied Mathematics. Rice University, 2004.
[20] D. E. Keyes, *How scalable is domain decomposition in practice?*, in Proceedings of the 11th International Conference on Domain Decomposition Methods, C.-H. Lai et al., ed., 1998, pp. 286–297.
[21] Z. Li and Y. Saad, *SchurRAS: A restricted version of the overlapping Schur complement preconditioner*, Tech. Report UMSI-2004-76, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, 2004.
[22] R. Nabben and D. B. Szyld, *Convergence theory of restricted multiplicative Schwarz methods*, SIAM J. Numer. Anal., 40 (2003), pp. 2318–2336.
[23] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, First ed., 2000.
[24] S. C. R. Noël M. Nachtigal and L. N. Trefethen, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 778–795.
[25] E. Prudencio, R. Byrd, and X.-C. Cai, *Parallel full space SQP Lagrange-Newton-Krylov-Schwarz algorithms for PDE-constrained optimization problems*, SIAM J. Sci. Comp., 27 (2006), pp. 1305–1328.
[26] E. E. Prudencio, *Parallel Fully Coupled Lagrange-Newton-Krylov-Schwarz Algorithms and Software for Optimization Problems Constrained by Partial Differential Equations*, PhD thesis, Department of Computer Science, University of Colorado at Boulder, 2005.
[27] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Second ed., 2003.
[28] J. Schöberl and W. Zulehner, *On Schwarz-type smoothers for saddle point problems*, Numerische Mathematik, 95 (2003), pp. 377–399.
[29] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, First ed., 1996.
[30] A. Toselli and O. Widlund, *Domain Decomposition Methods - Algorithms and Theory*, Springer-Verlag, 2005.
[31] X. Zhang, *Multilevel Schwarz methods*, Numer. Math., 63 (1992), pp. 521–539.