Parallel Two-Level Methods for Three-Dimensional Transonic Compressible Flow Simulations on Unstructured Meshes

R. Aitbayev[a], X.-C. Cai[a], and M. Paraschivoiu[b]

[a] Department of Computer Science, University of Colorado,
Boulder, CO 80309, U.S.A., {rakhim, cai}@cs.colorado.edu

[b] Department of Mechanical and Industrial Engineering, University of Toronto,
Toronto, Canada, M5S 3G8, marius@mie.utoronto.ca

We discuss our preliminary experiences with several parallel two-level additive Schwarz type domain decomposition methods for the simulation of three-dimensional transonic compressible flows. The focus is on the implementation of the parallel coarse mesh solver which is used to reduce the computational cost and speed up the convergence of the linear algebraic solvers. Results of a local two-level and a global two-level algorithm on a multiprocessor computer will be presented for computing steady flows around a NACA0012 airfoil using the Euler equations discretized on unstructured meshes.

1. INTRODUCTION

We are interested in the numerical simulation of three-dimensional inviscid steady-state compressible flows using two-level Schwarz type domain decomposition algorithms. The class of overlapping Schwarz methods has been studied extensively in the literature [11], especially, the single level version of the method [6,9]. It is well-known, at least in theory, that the coarse space plays a very important role in the fast and scalable convergence of the algorithms. Direct methods are often used to solve the coarse mesh problem either redundantly on all processors or on a subset of processors [3]. This presents a major difficulty in a fully parallel implementation for 3D problems, especially when the number of processors is large. In this paper, we propose several techniques to solve the coarse mesh problem in parallel, together with the local fine mesh problems, using two nested layers of preconditioned iterative methods.

The construction of the coarse mesh is an interesting issue by itself. We take a different approach than what is commonly used in the algebraic multigrid methods in which the coarse mesh is obtained from the given fine mesh – not the given geometry. In our two-level methods to be presented in this paper, we construct both the coarse and the fine mesh from the given geometry. To better fit the boundary geometry, the fine mesh nodes may not be on the faces of the coarse mesh tetrahedrons. In other words, the coarse space and fine space are not nested. This does not present a problem as long as the proper interpolation is defined [2].

As a test case, we consider a symmetric nonlifting flow over a NACA0012 airfoil in

a three-dimensional setting. In the presented approach, we construct the fine mesh by refining an existing coarse mesh and updating the nodes of the fine mesh according to the boundary geometry of the given physical domain. Such approach is easy to implement since the same computer code can be used on both the fine and the coarse level, and only a minimal additional programming is required to construct the restriction and prolongation operators. Moreover, it gives a natural partition of the fine mesh from the partition of the coarse mesh. In the tests, the system of Euler equations is discretized using the backward difference approximation in the pseudo-temporal variable and a finite volume method in the spatial variables. The resulting system of nonlinear algebraic equations is linearized using the Defect Correction (DeC) scheme. At each pseudo-temporal level, the linear system is solved by a restricted additive Schwarz preconditioned FGMRES method [10], and the coarse mesh problem is solved with an inner level of restricted additive Schwarz preconditioned FGMRES method.

2. GOVERNING EQUATIONS AND BOUNDARY CONDITIONS

Let $\Omega \subset R^3$ be a bounded flow domain with the boundary consisting of two parts: a wall boundary Γ_w and an infinity boundary Γ_∞. Let ρ be the density, $\vec{u} = (u, v, w)^T$ the velocity vector, e the total energy per unit volume, and p the pressure. We consider ρ, $\vec{u}$, e and p as the unknowns at point (x, y, z), and the pseudo-temporal variable t. Set $U = (\rho, \rho u, \rho v, \rho w, e)^T$, $\vec{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)^T$. An inviscid compressible flow in Ω is described by the Euler equations

$$U_t + \vec{\nabla} \cdot \vec{F} = 0, \tag{1}$$

where $\vec{F} = (F, G, H)^T$ is the flux vector with the Cartesian components defined as on page 87 of [7]. The equation (1) is closed by the equation of state for a perfect gas $p = (\gamma - 1)(e - \rho\|\vec{u}\|_2^2/2)$, where γ is the ratio of specific heats and $\|\cdot\|_2$ is the 2-norm in R^3. We specify the initial condition $U|_{t=0} = U_0$, where U_0 is an initial approximation to a steady-state solution, and the following boundary conditions. On the wall boundary Γ_w, we impose a no-slip condition for the velocity $\vec{u} \cdot \vec{n} = 0$, where $\vec{n}$ is the outward normal vector to the wall boundary. On the infinity boundary Γ_∞, we impose uniform free stream conditions $\rho = \rho_\infty$, $\vec{u} = \vec{u}_\infty$, and $p_\infty = 1/(\gamma M_\infty^2)$, where M_∞ is the free stream Mach number. We seek a steady-state solution, that is, the limits of ρ, $\vec{u}$, e and p as $t \to \infty$.

3. DISCRETIZATION

In this section, we present an outline of the discretization of the Euler equations; for more details, see [5]. Let Ω_h be a tetrahedral mesh in Ω, and N be the number of mesh points. For the pseudo-temporal discretization, we use a first-order backward difference scheme. For the spatial discretization of (1), we use a finite volume scheme in which control volumes are centered on the vertices of the mesh. For upwinding, we use Roe's approximate Riemann solver which has the first order spatial accuracy. Second order accuracy is achieved by the MUSCL technique [13] which uses piecewise linear interpolation at the interface between control volumes.

For $i = 1, 2, \ldots, N$ and $n = 0, 1 \ldots$, let U_i^n denote the value of the discrete solution at point (x_i, y_i, z_i) and at the pseudo-temporal level n and set $U_h^n = (U_1^n, U_2^n, \ldots, U_N^n)^T$.

Let $U_i^0 = U_0(x_i, y_i, z_i)$ and $\Psi_h(U_h) = (\Psi_1(U_h), \ldots, \Psi_N(U_h))^T$, where $\Psi_i(U_h)$ denotes the described second order approximation of convective fluxes $\vec{\nabla} \cdot \vec{F}$ at point (x_i, y_i, z_i). We define the local time step size by

$$\Delta t_i^n = C_{\text{CFL}} \, h_i / (C_i + \|\vec{u}_i^n\|_2),$$

where $C_{\text{CFL}} > 0$ is a preselected number, τ_i is a control volume centered at node i, h_i is its characteristic size, C_i is the sound speed and $\vec{u}_i^n$ is the velocity vector at node i. Then, the proposed scheme has a general form

$$(U_i^{n+1} - U_i^n)/\Delta t_i^n + \Psi_i(U_h^{n+1}) = 0, \quad i = 1, 2, \ldots, N, \quad n = 0, 1, \ldots . \tag{2}$$

We note, the finite volume scheme (2) has the first order approximation in the pseudo-temporal variable and the second order approximation in the spatial variable. On Γ_w, no-slip boundary condition is enforced. On Γ_∞, a non-reflective version of the flux splitting of Steger and Warming [12] is used.

We apply a DeC-Krylov-Schwarz type method to solve (2); that is, we use the Defect Correction scheme as a nonlinear solver, the restarted FGMRES algorithm as a linear solver, and the restricted additive Schwarz algorithm as the preconditioner.

At each pseudo-temporal level n, the equation (2) represents a system of nonlinear equations for the unknown variable U_h^{n+1}. This nonlinear system is linearized by the Defect Correction (DeC) scheme [1] formulated as follows. Let $\tilde{\Psi}_h(U_h)$ be the first-order approximation of convective fluxes $\vec{\nabla} \cdot \vec{F}$ obtained in way similar to that of $\Psi_h(U_h)$, and let $\partial \tilde{\Psi}_h(U_h)$ denote its Jacobian. Suppose that, for fixed n, an initial guess $U_h^{n+1,0}$ is given (say $U_h^{n+1,0} = U_h^n$). For $s = 0, 1, \ldots$, solve for $U_h^{n+1,s+1}$ the following linear system

$$\left(D_h^n + \partial \tilde{\Psi}_h \left(U_h^{n+1,0}\right)\right) \left(U_h^{n+1,s+1} - U_h^{n+1,s}\right) = -D_h^n \left(U_h^{n+1,s} - U_h^n\right) - \Psi_h \left(U_h^{n+1,s}\right),$$

where $D_h^n = \mathrm{diag}\,(1/\Delta t_1^n, \ldots, 1/\Delta t_N^n)$ is a diagonal matrix. The DeC scheme (3) preserves the second-order approximation in the spatial variable of (2). In our implementation, we carry out only one DeC iteration at each pseudo-temporal iteration, that is, we use the scheme

$$\left(D_h^n + \partial \tilde{\Psi}_h \left(\tilde{U}_h^n\right)\right) \left(\tilde{U}_h^{n+1} - \tilde{U}_h^n\right) = -\Psi_h \left(\tilde{U}_h^n\right), \quad n = 0, 1, \ldots, \quad \tilde{U}_h^0 = U_h^0. \tag{3}$$

4. LINEAR SOLVER AND PRECONDITIONING

Let the nonlinear iteration n be fixed and denote

$$A = D_h^n + \partial \tilde{\Psi}_h \left(\tilde{U}_h^n\right). \tag{4}$$

Matrix A is nonsymmetric and indefinite in general. To solve (4), we use two nested levels of restarted FGMRES methods [10]; one at the fine mesh level and one at the coarse mesh level inside the additive Schwarz preconditioner (AS) to be discussed below.

4.1. One-level AS preconditioner

To accelerate the convergence of linear iterations in the FGMRES algorithm, we use an additive Schwarz preconditioner. The method splits the original linear system into a collection of independent smaller linear systems which could be solved in parallel.

Let Ω_h be subdivided into k non-overlapping subregions $\Omega_{h,1}$, $\Omega_{h,2}$, ..., $\Omega_{h,k}$. Let $\Omega'_{h,1}$, $\Omega'_{h,2}$, ..., $\Omega'_{h,k}$ be overlapping extensions of $\Omega_{h,1}$, $\Omega_{h,2}$, ..., $\Omega_{h,k}$, respectively, and be also subsets of Ω_h. The size of overlap is assumed to be small, usually one mesh layer. The node ordering in Ω_h determines the node orderings in the extended subregions. For $i = 1, 2, \ldots, k$, let R_i be a global-to-local restriction matrix that corresponds to the extended subregion $\Omega'_{h,i}$, and let A_i be a "part" of matrix A that corresponds to $\Omega'_{h,i}$. The AS preconditioner is defined by

$$M_{AS}^{-1} = \sum_{i=1}^{k} R_i^T A_i^{-1} R_i.$$

For certain matrices arising from the discretizations of elliptic partial differential operators, an AS preconditioner is spectrally equivalent to the matrix of a linear system with the equivalence constants independent of the mesh step size h, although, the lower spectral equivalence constant has a factor $1/H$, where H is the subdomain size. For some problems, adding a coarse space to the AS preconditioner removes the dependency on $1/H$, hence, the number of subdomains [11].

4.2. One-level RAS preconditioner

It is easy to see that, in a distributed memory implementation, multiplications by matrices R_i^T and R_i involve communication overheads between neighboring subregions. It was recently observed [4] that a slight modification of R_i^T allows to save half of such communications. Moreover, the resulting preconditioner, called the restricted AS (RAS) preconditioner, provides faster than the original AS preconditioner convergence for some problems. The RAS preconditioner has the form

$$M_{RAS}^{-1} = \sum_{i=1}^{k} R_i'^T A_i^{-1} R_i,$$

where $R_i'^T$ corresponds to the extrapolation from $\Omega_{h,i}$. Since it is too costly to solve linear systems with matrices A_i, we use the following modification of the RAS preconditioner:

$$M_1^{-1} = \sum_{i=1}^{k} R_i'^T B_i^{-1} R_i, \tag{5}$$

where B_i corresponds to the $ILU(0)$ decomposition of A_i. We call M_1 the *one-level RAS preconditioner* ($ILU(0)$ modified).

4.3. Two-level RAS preconditioners

Let Ω_H be a coarse mesh in Ω, and let R_0 be a fine-to-coarse restriction matrix. Let A_0 be a coarse mesh version of matrix A defined by (4). Adding a scaled coarse mesh component to (5), we obtain

$$M_2^{-1} = (1 - \alpha) \sum_{i=1}^{k} R_i'^T B_i^{-1} R_i + \alpha \, R_0^T A_0^{-1} R_0, \tag{6}$$

where $\alpha \in [0, 1]$ is a scaling parameter. We call M_2 the *global two-level RAS precondi-tioner* ($ILU(0)$ modified). Preconditioning by M_2 requires solving a linear system with matrix A_0, which is still computationally costly if the linear system is solved directly and redundantly. In fact, the approximation to the coarse mesh solution could be sufficient for a better preconditioning. Therefore, we solve the coarse mesh problem in parallel using again a restarted FGMRES algorithm, which we call the *coarse mesh FGMRES*, with a modified RAS preconditioner.

Let Ω_H be divided into k subregions $\Omega_{H,1}, \Omega_{H,2}, \ldots, \Omega_{H,k}$ with the extented counterparts $\Omega'_{H,1}, \Omega'_{H,2}, \ldots, \Omega'_{H,k}$. To solve the coarse mesh problem, we use FGMRES with the one-level $ILU(0)$ modified RAS preconditioner

$$M_{0,1}^{-1} = \sum_{i=1}^{k} (R'_{0,i})^T B_{0,i}^{-1} R_{0,i}, \tag{7}$$

where, for $i = 1, 2, \ldots, N$, $R_{0,i}$ is a global-to-local coarse mesh restriction matrix, $(R'_{0,i})^T$ is a matrix that corresponds to the extrapolation from $\Omega_{H,i}$, and $B_{0,i}$ is the $ILU(0)$ decomposition of matrix $A_{0,i}$, a part of A_0 that corresponds to the subregion $\Omega'_{H,i}$. After r coarse mesh FGMRES iterations, A_0^{-1} in (6) is approximated by $\tilde{A}_0^{-1} = \text{poly}_l(M_{0,1}^{-1} A_0)$ with some $l \leq r$, where $\text{poly}_l(x)$ is a polynomial of degree l, and its explicit form is often not known. We note, l maybe different at different fine mesh FGMRES iterations, and it depends on a stopping condition. Therefore, FGMRES is more appropriate than the regular GMRES. Thus, the actual preconditioner for A has the form

$$\tilde{M}_2^{-1} = (1 - \alpha) \sum_{i=1}^{k} R_i'^{T} B_i^{-1} R_i + \alpha \, R_0^{T} \tilde{A}_0^{-1} R_0. \tag{8}$$

For the fine mesh linear system, we also use a preconditioner obtained by replacing A_0^{-1} in (6) with $M_{0,1}^{-1}$ defined by (7):

$$M_3^{-1} = \sum_{i=1}^{k} \left((1 - \alpha) \, R_i'^{T} B_i^{-1} R_i + \alpha \, R_0^{T} (R'_{0,i})^T B_{0,i}^{-1} R_{0,i} R_0 \right). \tag{9}$$

We call M_3 a *local two-level RAS preconditioner* ($ILU(0)$ modified) since the coarse mesh problems are solved locally, and there is no global information exchange among the subre-gions. We expect that M_3 works better than M_1 and that $\tilde{M}_2$ does better than M_3. Since no theoretical results are available at the present, we test the described preconditioners M_1, $\tilde{M}_2$, and M_3 numerically.

5. NUMERICAL EXPERIMENTS

We computed a compressible flow over a NACA0012 airfoil on the computational do-main with the nonnested coarse and fine meshes. First, we constructed an unstructured coarse mesh Ω_H; then, the fine mesh Ω_h was obtained by refining the coarse mesh twice. At each refinement step, each coarse mesh tetrahedron was subdivided into 8 tetrahe-drons. After each refinement, the boundary nodes of the fine mesh were adjusted to the geometry of the domain. Sizes of the coarse and fine meshes are given in Table 1.

Table 1

Coarse and fine mesh sizes

	Coarse	Fine	Fine/coarse ratio
Nodes	2,976	117,525	39.5
Tetrahedrons	9,300	595,200	64
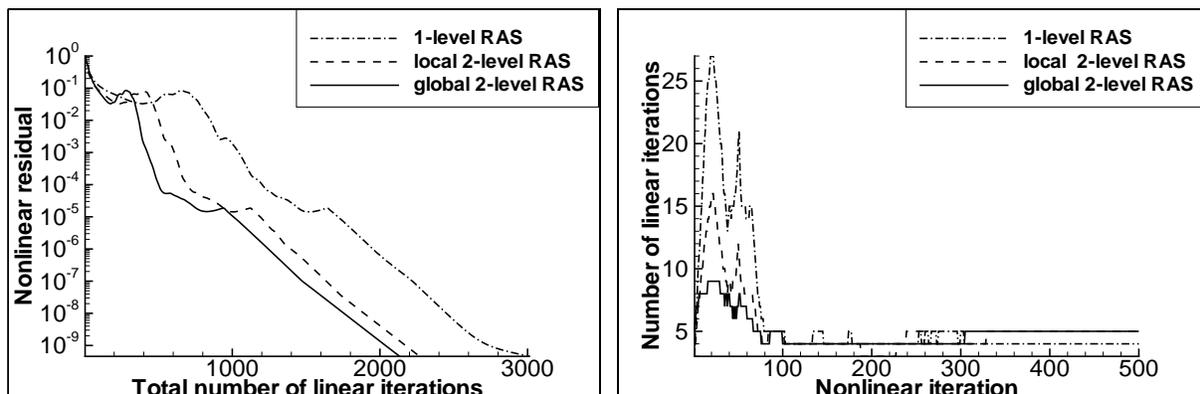
Figure 1. Comparison of the one-level, local two-level, and global two-level RAS preconditioners in terms of the total numbers of linear iterations(left picture) and nonlinear iterations (right picture). The mesh has 32 subregions.

For parallel processing, the coarse mesh was divided, using METIS [8], into 16 or 32 submeshes with nearly the same number of tetrahedrons. The fine mesh partition was obtained directly from the corresponding coarse mesh partition. The size of overlap both in the coarse and the fine mesh partition was set to one, that is, two neighboring extended subregions share a single layer of tetrahedrons. In (8) and (9), R_0^T was set to a matrix of a piecewise linear interpolation. Multiplications by R_0^T and R_0, solving linear systems with M_1, $\tilde{M}_2$, and M_3, and both the fine and the coarse FGMRES algorithm were implemented in parallel. The experiments were carried out on an IBM SP2.

We tested convergence properties of the preconditioners defined in (5), (8), and (9) with $\alpha = N_c/N_f$, where N_c and N_f are the numbers of nodes in the coarse and fine meshes, respectively. We studied a transonic case with M_∞ set to 0.8. Some of the computation results are presented in Figures 1 and 2.

The left picture in Figure 1 shows residual reduction in terms of total numbers of linear iterations. We see that the algorithms with two-level RAS preconditioners give significant improvements compared to the algorithm with the one-level RAS preconditioner. The improvement in using the global two-level RAS preconditioner compared to the local two-level RAS preconditioner is not very much. Recall, that in the former case the inner FGMRES is used which could increase the CPU time. In Table 2, we present a summary from the figure. We see that the reduction percentages in the numbers of linear iterations drop with the decrease of the nonlinear residual (or with the increase of the nonlinear iteration number). This is seen even more clear in the right picture in Figure 1. After

Table 2
Total numbers of linear iterations and the reduction percentages compared to the algorithm with the one-level RAS preconditioner (32 subregions).

Residual	One-level RAS Iterations	Local two-level RAS Iterations	Reduction	Global two-level RAS Iterations	Reduction
10^{-2}	859	513	40%	371	57%
10^{-4}	1,205	700	42%	503	58%
10^{-6}	1,953	1,397	28%	1,245	36%
10^{-8}	2,452	1,887	23%	1,758	28%
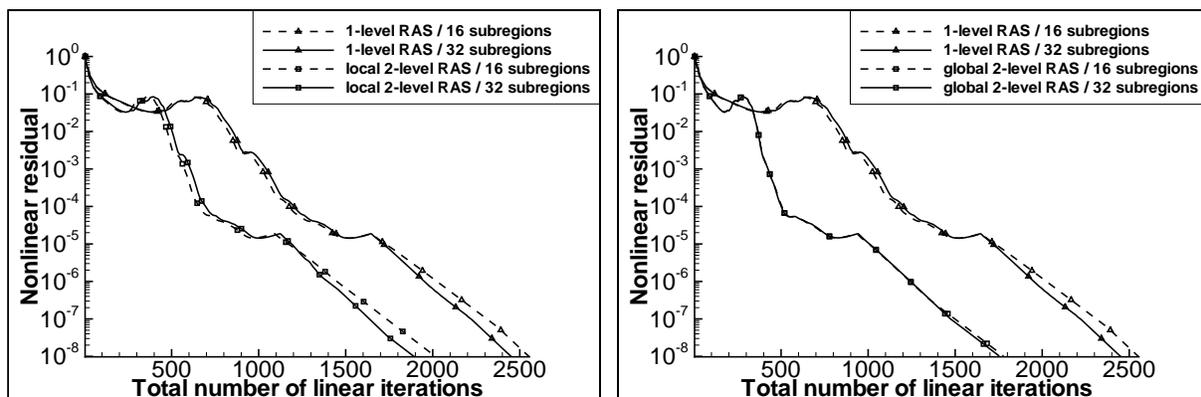
Figure 2. Comparison of the one-level RAS preconditioner with the local two-level RAS (left picture) and the global two-level RAS preconditioner (right picture) on the meshes with 16 and 32 subregions.

approximately 80 nonlinear iterations, the three algorithms give basically the same number of linear iterations at each nonlinear iteration. This suggests that the coarse mesh may not be needed after some number of initial nonlinear iterations.

In Figure 2, we compare the algorithms on the meshes with different numbers of subregions, 16 and 32. The left picture shows that the algorithms with the one-level and local two-level RAS preconditioners initially increase the total numbers of linear iterations as the number of subregions was increased from 16 to 32. On the other hand, we see in the right picture in Figure 2 that the the increase in the number of subregions almost did not affect the convergence of the algorithm with the global two-level RAS preconditioner. These results suggest that the algorithm with the global two-level RAS preconditioner is well scalable to the number of subregions (processors) while the other two are not. In both pictures we observe the decrease in the total number of linear iterations to the end of computations. This is due to the fact that only 4 or 5 linear iterations were carried out at each nonlinear iteration in both cases, with 16 and 32 subregions (see the right picture in Figure 1), with linear systems in the case of 32 subregions solved just one iteration faster than the linear systems in the case of 16 subregions.

6. CONCLUSIONS

When both the fine and the coarse mesh is constructed from the domain geometry, it is fairly easy to incorporate a coarse mesh component into a one-level RAS preconditioner. The applications of the two-level RAS preconditioners give a significant reduction in total numbers of linear iterations. For our test cases, the coarse mesh component seems not needed after some initial number of nonliner iterations. The algorithm with the global two-level RAS preconditioner is scalable to the number of subregions (processors). Sizes of fine and coarse meshes should be well balanced, that is, if a coarse mesh is not coarse enough, the application of a coarse mesh component could result in the CPU time increase.

REFERENCES

1. K. BOHMER, P. HEMKER, AND H. STETTER, *The defect correction approach*, Comput. Suppl, 5 (1985), pp. 1–32.
2. X.-C. CAI, *The use of pointwise interpolation in domain decomposition methods with non-nested meshes*, SIAM J. Sci. Comput., 16 (1995), pp. 250–256.
3. 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.
4. X.-C. CAI AND M. SARKIS, *A restricted additive schwarz preconditioner for general sparse linear systems*, SIAM J. Sci. Comput., (1999). To appear.
5. C. FARHAT AND S. LANTERI, *Simulation of compressible viscouse flows on a variety of MPPs: computational algorithms for unstructured dynamic meshes and performance results*, Comput. Methods Appl. Mech. Engrg., 119 (1994), pp. 35–60.
6. W. D. GROPP, D. E. KEYES, L. C. MCINNES, AND M. D. TIDRIRI, *Globalized Newton–Krylov–Schwarz algorithms and software for parallel implicit CFD*, Int. J. High Performance Computing Applications, (1999). Submitted.
7. C. HIRSCH, *Numerical Computation of Internal and External Flows*, John Wiley and Sons, New York, 1990.
8. G. KARYPIS AND V. KUMAR, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.
9. D. K. KAUSHIK, D. E. KEYES, AND B. F. SMITH, *Newton–Krylov–Schwarz methods for aerodynamics problems: Compressible and incompressible flows on unstructured grids*, in Proc. of the Eleventh Intl. Conference on Domain Decomposition Methods in Scientific and Engineering Computing, 1999. To appear.
10. Y. SAAD, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Stat. Comput., 14 (1993), pp. 461–469.
11. B. F. SMITH, P. E. BJØRSTAD, AND W. D. GROPP, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.
12. J. STEGER AND R. F. WARMING, *Flux vector splitting for the inviscid gas dynamic with applications to finite-difference methods*, J. Comp. Phys., 40 (1981), pp. 263–293.
13. B. VAN LEER, *Towards the ultimate conservative difference scheme V: a second order sequel to Godunov's method*, J. Comp. Phys., 32 (1979), pp. 361–370.