Thirteenth International Conference on Domain Decomposition Methods
Editors: N. Debit, M.Garbey, R. Hoppe, J. Périaux, D. Keyes, Y. Kuznetsov ©2001 DDM.org

53 Schur Complement Based Preconditioners for Compressible Flow Computations

Marzio Sala[1]

Introduction

The solution of linear systems arising from compressible flow computations is a great challenge in the field of scientific computing. Modern high-performance computers are very often organised as a distributed environment, and every efficient solver must account for their multiprocessor nature. Domain decomposition (DD) techniques provide a natural possibility to combine classical and well-tested single-processor algorithms with parallel new ones. The basic idea is to decompose the original computational domain Ω into M smaller parts, called subdomains $\Omega^{(i)}$, $i = 1, \dots, M$, such that $\cup_{i=1}^{M} \bar{\Omega}^{(i)} = \bar{\Omega}$. Then we replace the global problem on Ω with M problems on $\Omega^{(i)}$. Of course, additional interface conditions must be provided.

The DD methods can roughly be classified into two groups [QV99, SBG96, CM94]. In the former, named after Schwarz, the computational domain is subdivided into overlapping subdomains, and local Dirichlet-type problems are then solved on each subdomain. The latter group, instead, uses non-overlapping subdomains. It is thus possible to decompose the unknowns into two sets: one formed by the unknowns on the interface between subdomains, and another formed by the unknowns associated to nodes internal to the subdomains. One may then compute a Schur complement (SC) matrix by "condensing" the unknowns in the second set. The system is then solved by first computing the interface unknowns and then solving the independent problems for the internal unknowns.

It can be shown [QV99] that the SC system is better conditioned than the global system. However, the solution of this system requires computing as many linear problems as the number of subdomains used. The dimension of these problems can be very large, unless the number of processors used is sufficiently high. A possible solution can be to solve the internal problems inexactly, using, for example, an incomplete factorisation [Saa96], or few steps of an iterative solver. The resulting approximate SC matrix can be seen as a preconditioner for the global system. Here we present some numerical results concerning the application of the SC matrix as a preconditioner for the global (unreduced) system. We have tested an elliptic problem, as well as a hyperbolic one. In the former the matrix arises from the Laplace operator, while in the latter from the compressible Euler equations.

This paper is organised as follows. Section 53 describes the SC system, showing two possible formulations, named element-oriented and vertex-oriented. Differences between the element-oriented and vertex-oriented SC matrix are here outlined. Section 53 describes the use of the SC system as a preconditioner for the global system. Numerical results for an elliptic test case and for the compressible Euler equations are reported in Section 53 . The tests have been conducted on a distributed memory parallel machine. Conclusions are drawn in Section 53.

[1]Département de Mathématiques, EPF-Lausanne

The Schur complement Method

Let us consider the solution of the following linear system:

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

where $A \in \mathbb{R}^{n\times n}$ is a (sparse) real matrix, $\mathbf{u}$ and $\mathbf{f} \in \mathbb{R}^n$ two column vectors. In general, we can think of (1) as begin the algebraic counterpart of a variational boundary value problem which reads

$$\begin{array}{l} \text{find } u_h \in V_h \text{ such that:} \\ a\left(u_h, v_h\right) = \left(f, v_h\right) \quad \text{for } \forall v_h \in V_h \, , \end{array}$$

where V_h is a finite dimensional space generated from finite element basis functions. For an elliptic problem $a(u_h, v_h)$ is bilinear form and u_h the discrete solution, while for the compressible Euler equations $a(u_h, v_h)$ should be regarded as the bilinear form expressing the Jacobian of the Euler system (after time and space discretisation), and u_h plays the role of the increment of the physical variables.

We now consider a partition of the domain $\Omega \subset \mathbb{R}^d, d = 2, 3$, made in the following way. We first triangulate Ω and we indicate by $\mathcal{T}_h^{(\Omega)}$ the corresponding mesh. For the sake of simplicity we assume that the boundary of Ω coincides with the boundary of the triangulation. We then partition $\mathcal{T}_h^{(\Omega)}$ into 3 parts, namely $\mathcal{T}_h^{(1)}$, $\mathcal{T}_h^{(2)}$ and $\mathcal{T}_h^{(\Gamma)}$ such that $\mathcal{T}_h^{(1)} \cup \mathcal{T}_h^{(2)} \cup \mathcal{T}_h^{(\Gamma)} = \mathcal{T}_h^{(\Omega)}$. We may associate to $\mathcal{T}_h^{(1)}$ and $\mathcal{T}_h^{(2)}$ two disjoint subdomains $\Omega^{(1)}$ and $\Omega^{(2)}$ formed by the interior of the union of the elements of $\mathcal{T}_h^{(1)}$ and $\mathcal{T}_h^{(2)}$ respectively, while $\Gamma^{(1,2)}$ is formed by the "elements" contained on $\mathcal{T}_h^{(\Gamma)}$.

We will consider two cases:

- $\Gamma^{(1,2)}$ reduces to a finite number of disjoint measurable $d-1$ manifolds. This situation represents the common case where $\bar{\Omega}^{(1)} \cap \bar{\Omega}^{(2)} = \Gamma^{(1,2)}$, i.e. $\Gamma^{(1,2)}$ is the discretisation of the common part Γ of the boundary of $\Omega^{(1)}$ and $\Omega^{(2)}$. This type of decomposition will be called *element oriented* (EO) decomposition, because each element of $\mathcal{T}_h^{(i)}, i = 1, 2$ belongs exclusively to one of the two subdomains $\bar{\Omega}^{(1)}$ and $\bar{\Omega}^{(2)}$.

- $\Gamma^{(1,2)} \subset \mathbb{R}^d$ and it is formed by only one layer of elements. That is, each node of $\Gamma^{(1,2)}$ coincides with a node of either $\mathcal{T}_h^{(1)}$ or $\mathcal{T}_h^{(2)}$. The portion of space Γ of which $\Gamma^{(1,2)}$ is a triangulation, is now formed by the union of a finite number of "strips" laying between $\Omega^{(1)}$ and $\Omega^{(2)}$. It will be called *vertex oriented* (VO) decomposition, because each vertex belongs exclusively to one of the two subdomains $\bar{\Omega}^{(i)}, i = 1, 2$.

A node is said to be *internal* if it is not connected to any node of other subdomains, while a node that lies on $\Gamma^{(1,2)}$ is said to be a *border* node. In the following, we will consistently use the subscripts I and B to indicate internal and border nodes, respectively, while the superscript (i) will denote the domain which we are referring to.

Vertex Oriented Schur complement Matrix

Let us consider again problem (1). The block representation reads

$$A\mathbf{u} = \begin{pmatrix} \multicolumn{2}{c}{A^{(1)}} & \begin{matrix} 0 & 0 \\ 0 & E^{(1,2)} \end{matrix} \\ \begin{matrix} 0 & 0 \\ 0 & E^{(2,1)} \end{matrix} & A^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^{(1)} \\ \mathbf{u}_B^{(1)} \\ \mathbf{u}_I^{(2)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I^{(1)} \\ \mathbf{f}_B^{(1)} \\ \mathbf{f}_I^{(2)} \\ \mathbf{f}_B^{(2)} \end{pmatrix}, \qquad (2)$$

where the submatrix $A^{(i)}$, relative to subdomain $\Omega^{(i)}$, can be written as

$$A^{(i)} = \begin{pmatrix} A_{II}^{(i)} & A_{IB}^{(i)} \\ A_{BI}^{(i)} & A_{BB}^{(i)} \end{pmatrix}.$$

In this partitioning the border nodes are subdivided into two sets: $B^{(1)}$ is the set of nodes of the triangulation of the strips $\Gamma^{(1,2)}$ which lay on the boundary of $\Omega^{(1)}$, while $B^{(2)}$ is that of nodes lying on the boundary of $\Omega^{(2)}$. Correspondingly, we have the blocks $\mathbf{u}_B^{(1)}$ and $\mathbf{u}_B^{(2)}$ in the vector of unknowns and $\mathbf{f}_B^{(1)}$ and $\mathbf{f}_B^{(2)}$ in the right hand side. $E^{(i,j)}$ represents the contribution to the equation associated to $B^{(i)}$ coming from the nodes in $B^{(i)}$. We call the nodes of $\Gamma^{(1,2)}$ contributing to $B^{(i)}$ *external nodes* of domain $\Omega^{(i)}$.
We can perform a LU elimination of internal nodes (which are coupled only to border nodes), obtaining the following Schur complement system:

$$S_{VO}\mathbf{u}_B = \begin{pmatrix} S^{(1)} & E^{(1,2)} \\ E^{(2,1)} & S^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_B^{(1)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \end{pmatrix}, \qquad (3)$$

where

$$S^{(i)} = A_{BB}^{(i)} - A_{BI}^{(i)} A_{II}^{(i)^{-1}} A_{IB}^{(i)} \text{ and } \mathbf{g}^{(i)} = \mathbf{f}_B^{(i)} - \sum_{i=1}^{M} R_i^T A_{BI}^{(i)} A_{II}^{(i)^{-1}} \mathbf{f}_I^{(i)}, \qquad (4)$$

with $i = 1, 2$. Note that S_{VO} is in general dense on the block diagonal, while the blocks $E^{(i,j)}$ are sparse. The technique just shown may be extended to any number of domains.
The Schur operators built on an element-oriented or a vertex-oriented DD are clearly different. The theoretical properties of the former are better known, since it has a more direct interpretation at differential level [QV99], while the latter is normally the result of a purely algebraic approach. Although the element-oriented decomposition has better theoretical foundations, the literature for complex CFD computations refers more frequently to vertex-oriented decomposition. In fact, the vertex-oriented approach is more simple derived by purely algebraic manipulations on the original system matrix.
Although the SC matrix is better conditioned that the unreduced matrix A, a suitable preconditioner has to be found. Many methods have been proposed in literature for the EO decomposition; see for example [QV99, SBG96] for an overview. Among them, we recall the balancing Neumann/Neumann, the wire-basket preconditioners FETI [FPL00] and others [CGT98]. These methods couple a local preconditioner with a coarse correction to avoid the degradation of the performance as the number of subdomains grows. Another possible way to derive a preconditioner for equation (3) is to use using the relation $S_{VO}^{-1} = R_B A^{-1} R_B^T$,

where R_B the restriction operator on the interface variables. It follows that form that from *any* preconditioner P_A for the matrix A one can obtain a preconditioner P_S for the matrix S_{VO}. The preconditioning operation for S_{VO} which is induced from P_A is defined by

$$P_S^{-1}\mathbf{v}_B = R_B P_A^{-1} \begin{pmatrix} 0 \\ \mathbf{v}_B \end{pmatrix} = R_B P_A^{-1} R_B^T \mathbf{v}_B \ .$$

For example, a Schwarz-type preconditioner can be used for the solution of the vertex-oriented SC matrix. In fact, we recall that the SC matrix obtained from a vertex-oriented decomposition is less dense than the one obtained from an element-oriented one. As one may note from equation (3), S_{VO} has dense diagonal blocks, while the non-diagonal blocks are sparse.

The Schur Complement System as a Preconditioner

The bottleneck of the SC system is the solution of the internal problems. This step can be done in parallel, however it can be very expensive for both memory requirement and time. Direct solvers can be used only with small problems, while iterative solvers need to be preconditioned.
Let us write the matrix A in the following block form, putting before all the internal nodes, followed by all border nodes:

$$A = \begin{pmatrix} A_{II} & 0 \\ A_{BI} & I \end{pmatrix} \begin{pmatrix} I & A_{II}^{-1} A_{IB} \\ 0 & S \end{pmatrix} \ . \qquad (5)$$

A possible preconditioner is

$$P_{ASC} = \begin{pmatrix} \tilde{A}_{II} & 0 \\ A_{BI} & I \end{pmatrix} \begin{pmatrix} I & \tilde{A}_{II}^{-1} A_{IB} \\ 0 & \tilde{S} \end{pmatrix} \ , \qquad (6)$$

where $\tilde{A}_{II}$ is, for example, an ILU decomposition of A_{II}, and $\tilde{S}$ is given, for instance, by few steps of an iterative method where in the *global* Schur Complement the internal Dirichlet problems are solved (approximately) using $\tilde{A}_{II}$. To apply P_{ASC} to a vector, we need to solve some local linear systems with the matrices A_{II} and a *global* linear system with $\tilde{S}$. In this case, the role of $\tilde{S}$ is to couple all the subdomains. In this way, we may avoid the definition of a coarse space. In fact, the definition of the coarse problem may be difficult when dealing with complex geometries or non-matching grids [CSZ96], especially for the choice of the boundary conditions. On the contrary, the definition of the ASC preconditioner is purely algebraic and it can be easily applied to any kind of matrices (provided that the incomplete factorization of A_{II} exists). This approach is similar to the one followed in [Zha00] and other papers with the same aim to construct the preconditioner without dealing with the geometrical data of the underline physical problem.

Numerical Implementation

In Section 53 we present some numerical results concerning a Laplace operator, while in Section 53 we apply the SC preconditioner to the solution of the compressible Euler equations. All the numerical results here presented have been obtained on a SGI-Cray machine located at

N_unks	np	sw1	sw2	ASC-2	ASC-5	ASC-10
4 processor						
64.000	116	38	33	35	28	24
125.000	154	46	39	43	36	32
216.000	227	54	45	50	42	39
512.000	253	66	58	66	56	51
1.000.000	454	91	66	81	67	64
8 processor						
64.000	116	43	35	32	22	14
125.000	154	51	40	38	29	20
216.000	227	56	48	45	37	27
512.000	253	68	59	60	49	40
1.000.000	454	105	67	73	64	54

Table 1: 3D Laplace problem. Number of iterations with 4 and 8 processors.

the EPFL, with 32 MIPS R14000 processors, each of them has 256Kbytes of local memory, 32 Kbytes of first level cache and 4 Mbytes of second level cache. For the solution of the linear system, we have used the AZTEC library, developed at the Sandia National Laboratories. The linear solver used is GMRESR, a variant of GMRES that allows the preconditioner to be different at each iteration. We have stopped the solver after a reduction of 10^{-5} of the initial residual. Each processor is given a single subdomain, and the MPI communicator has been used. About the Schwarz preconditioner, we have solved the local problem using an ILU decomposition. For the ASC preconditioner, in the solution of the linear system with $\tilde{S}$, we have used GMRES. The approximation is obtained replacing the exact LU of A_{II} decomposition the an incomplete factorisation $ILU(0)$.

An Elliptic Problem: the Laplace Operator

We have considered the following linear problem:

$$\begin{cases} -\Delta u &= f \text{ in } \Omega \\ u &= g \text{ on } \partial\Omega\,, \end{cases} \tag{7}$$

where $\Omega = (0,1) \times (0,1) \times (0,1)$ is discretized by piece-linear finite elements on tetrahedra regular grids. For this simple test case we have partitioned the domain into slices, using a vertex-oriented decomposition as indicated in Section 53. In Table 1 we have reported the iterations to converge using 4 and 8 subdomains for different values of the numbers of the unknowns. We indicate with np the non preconditioned case, sw1 the Schwarz preconditioner with an overlap of 1 element, sw2 with an overlap of 2 elements. ASC-L represents the ASC preconditioner, with L steps of the nested iterative solver.
One may note that the ASC preconditioner behaves better than the 1-level Schwarz preconditioner for suitable values of L. This value can be increased to improve the efficacy of the ASC preconditioner. Moreover, as the number of subdomains grows, the ASC preconditioner requires less iterations to converge.

A Hyperbolic Problem: The Euler equations

Let us consider the Euler equations for compressible flows, that can be written in the following form:

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

(plus suitable boundary conditions on $\partial\Omega$), where $\mathbf{U}$ and $\mathbf{F_j}$ are, respectively, the vector of conservative variables and the flux vector:

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

with $i = 1, \dots, d$ ($\Omega \subset \mathbb{R}^d$), and $\mathbf{u}$ is the velocity vector, ρ the density, p the pressure, H the specific enthalpy and δ_{ij} the Kronecker symbol.
The spatial discretisation applied to the Euler equations leads eventually to a system of ODE in time, which may be written as $dU/dt = R(U)$, where $U = (U_1, U_2, \dots, U_l, \dots)^T$ is the vector of unknown nodal states $U_i = U_i(t)$ and $R(U)$ the result of the spatial discretization of the Euler fluxes. Applying the backward Euler method to the semi-discrete equation yields

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

where Δt here represents a *diagonal matrix* of local time steps, since we are interested only in steady-state solutions. We adopt the so-called “local time stepping” technique, where the degrees of freedom associated to each node evolve with their own time step. This is a rather common technique to accelerate convergence to steady state. In order to solve the nonlinear problem (9), the Newton method is used. We refer to the literature for more detailed explanations (see, for example [BCT98, SLW98, KKS98]).
For its spatial discretization, we have used the code THOR, developed at the von Karman Institute, that makes use of the multidimensional upwind finite element discretization, while for the vertex-oriented decomposition of the computational domain we have used the software METIS. This decomposition is unstructured and the subdomains have no particular shape, as one may appreciate from the picture on the left of Figure 1.
The first test case is represented by a NACA0012 airfoil with one degree of angle of attack. The free-stream Mach number is 0.85. 41 time iterations were required to reach the convergence to the steady state. The CFL number goes from 10 to 10^5, and it is multiplied on each iteration by 2.
Tables 2 reports a comparison between the Schwarz and the ASC preconditioners using from 4 to 32 processors. As we can observe, the gain in terms of number of iterations using the ASC preconditioner can be very high especially as the number of subdomains grows. Although, the time is (slightly) bigger that the one needed by the Schwarz preconditioners.
The second test case correspond to the solution of the compressible Euler equations around an ONERA M6 wing. The 3D unstructured grid has 94493 nodes and 555514 elements. The free-stream Mach number is 0.84, and the angle of attack is 3.06. The CFL number goes from 10 to 10^6, multiplied by 2 at each time iteration. In Table 3 we have reported the CPU time required to reach the steady state. As we can observe, few iterations in the solution of $\tilde{S}$ seems to be appropriate to reach the prescribed accuracy.

N_procs	sw1	sw2	ASC-2	ASC-5	sw1	sw2	ASC-2	ASC-5
	Iterations				CPU-time			
4	487	454	370	369	124.8	162.7	312.7	276.0
8	507	458	357	351	56.6	65.4	125.8	165.0
16	544	488	329	317	36.1	40.5	60.2	66.8
32	587	502	311	278	21.3	24.8	29.2	42.0

Table 2: Iterations and total time (in seconds) required to reach the convergence. NACA0012 airfoil, 9239 nodes.

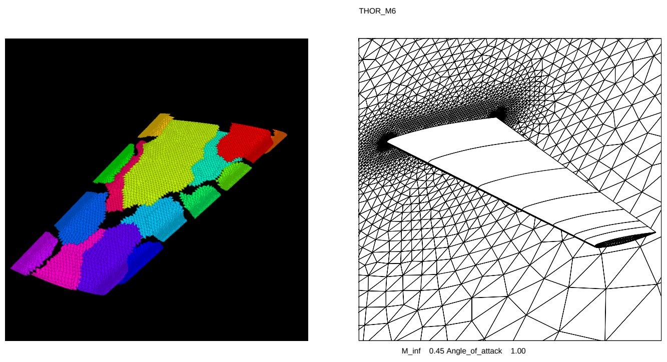

Figure 1: M6 Wing. Decomposition of the elements on the surface among subdomains (left) and particular of the unstructured 3D grid (right).

N_procs	ASC-2	ASC-4	ASC-8
8	**1538.4**	1600.4	1859.9
16	**544.8**	569.1	1330.5
32	**248.5**	286.0	358.9

Table 3: M6 wing, 94K nodes. CPU-time (in seconds) for ASC preconditioner, using different values of L.

Conclusion

In this paper, a preconditioner based on an approximation of the Schur complement system has been described. Numerical results have been presented for an elliptic problem and for the solution of the compressible Euler equations on 2D and 3D unstructured grids. The key idea is to use an approximate Schur complement matrix to precondition the unreduced matrix, exploiting the good parallel properties of the SC matrix. The approximate system is then solved by an iterative Krylov method, that couples all the subdomains.

The use of the SC system as a preconditioner for the global system seems promising, especially when the number of subdomains is large enough. The number of iterations to converge needed by the outer iterative solver is much lower than using a Schwarz preconditioner. Moreover, the effectiveness of the ASC preconditioner increases with the number of subdomains, even without a coarse operator, dislike the Schwarz method. Further numerical tests will be conducted to better investigate the parallel properties of this preconditioner.

References

[BCT98]T. J. Barth, T. F. Chan, and W. Tang. A parallel non-overlapping domain decomposition algorithm for compressible fluid flow problems on triangulated domains. In J. Mandel, C. Farhat, and X.-C. Cai, editors, *Tenth International Conference on Domain Decomposition Methods*, pages 23–41. AMS, Contemporary Mathematics 218, 1998.

[CGT98]Luiz Carvalho, Luc Giraud, and Patrick Le Tallec. Algebraic two-level preconditioners for the schur complement method. Technical report tr/pa/98/18, CERFACS, Toulouse, France, 1998. Preliminary version of the paper to appear in SIAM Journal on Scientific Computing.

[CM94]Tony F. Chan and Tarek P. Mathew. Domain decomposition algorithms. In *Acta Numerica 1994*, pages 61–143. Cambridge University Press, 1994.

[CSZ96]Tony F. Chan, Barry F. Smith, and Jun Zou. Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids. *Numer. Math.*, 73(2):149–167, 1996.

[FPL00]C. Farhat, K. H. Pierson, and M. Lesoinne. The second generation of feti methods and their application to the parallel solution of large-scale linear and geometrically nonlinear structural analysis problems. *Computer Methods in Applied Mechanics and Engineering*, 184:333–374, 2000.

[KKS98]D. K. Kaushik, D. E. Keyes, and B. F. Smith. NKS methods for compressible and incompressible flows on unstructured grids. *Proceedings of the 11th Intl. Conf. on Domain Decomposition Methods*, pages 513–520, 1998.

[QV99]Alfio Quarteroni and Alberto Valli. *Domain Decomposition Methods for Partial Differential Equations*. Oxford Science Publications, 1999.

[Saa96]Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, 1996.

[SBG96]Barry F. Smith, Petter E. Bjørstad, and William Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[SLW98]G. De Spiegeleer, A. Lerat, and Z. Wu. Implicit multidomain computation of compressible flows with a large number of subdomains. *Computational Fluid Dynamics Journal*, 7, 1998.

[Zha00]Jun Zhang. Preconditioned krylov subspace methods for solving nonsymmetric matrices. *Computer Methods in Applied Mechanics and Engineering*, 189(3):825–840, 2000.