Distributed Schur Complement Techniques for General Sparse Linear Systems *

Yousef Saad[†] Maria Sosonkina[‡]

March 5, 1998

Abstract Domain decomposition techniques constitute an excellent framework for describing and developing parallel algorithms for solving distributed linear systems. These systems are typically partitioned into blocks, which are then mapped into processors, most often one for each processor, and viewed as distributed objects (see, e.g, [10, 9, 11, 13, 7, 1, 2, 6]). The key issue is to develop preconditioners for the global linear system by exploiting its distributed data structure. The simplest of these techniques, Additive Schwarz, though easy to implement and subject to little parallel overhead, may fail to converge for difficult linear systems and for large numbers of processors. On the other hand, when the Schur complement related to the original global system is considered, iterative methods can be developed whose convergence properties do not deteriorate. Here we present Schur complement techniques that do not require an explicit construction of the global Schur complement matrix, meaning that each processor performs operations only on its local portion of the global Schur complement matrix.

Consider a linear system of the form

$$Ax = b, \tag{1}$$

where A is a large sparse nonsymmetric real matrix of size n. The system is partitioned into p processors such that pairs of equations-unknowns are assigned to the same processor. Thus, each processor holds a set of equations (rows of the linear system) and variables associated with these rows.

A good distributed data structure is crucial for the development of effective sparse iterative solvers. It is important, for example, to have a convenient representation of the local equations as well as the dependencies between the local and external variables. An important feature of the data structure used is the separation of the interface points from the interior points. In each processor, local points are ordered such that the interface points are listed last after the interior points.

With this local ordering, each local vector of unknowns x_i is split into two parts: the subvector u_i of internal vector components followed by the subvector y_i of local interface vector components. The right-hand side b_i is conformally split into the subvectors f_i and g_i. When block partitioned according to this splitting, the local matrix A_i residing in processor i has the form:

$$A_i = \left(\begin{array}{c|c} B_i & F_i \\ \hline E_i & C_i \end{array}\right), \tag{2}$$

so the local equations can be written as follows:

$$\begin{pmatrix} B_i & F_i \\ E_i & C_i \end{pmatrix} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \begin{pmatrix} 0 \\ \sum_{j\in N_i} E_{ij} y_j \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}. \tag{3}$$

Here, N_i is the set of indices for subdomains that are neighbors to the subdomain i. The term $E_{ij}y_j$ reflects the contribution to the local equation from the neighboring subdomain j.

*This work was supported in part by ARPA under grant number NIST 60NANB2D1272, in part by NSF under grant CCR-9618827, and in part by the Minnesota Supercomputer Institute.

[†]Department of Computer Science and Engineering, University of Minnesota, 200 Union Street S.E., Minneapolis, MN 55455, e-mail:saad@cs.umn.edu.

[‡]Department of Computer Science, University of Minnesota, 320 Heller Hall, 10 University Drive, Duluth, MN 55812, e-mail:masha@d.umn.edu.

Schur complement systems are derived by eliminating the variable u_i from the system (3). Extracting from the first equation $u_i = B_i^{-1}(f_i - F_i y_i)$ yields, upon substitution in the second equation,

$$S_i y_i + \sum_{j \in N_i} E_{ij} y_j = g_i - E_i B_i^{-1} f_i \equiv g_i', \tag{4}$$

where S_i is the "local" Schur complement $S_i = C_i - E_i B_i^{-1} F_i$.

The equations (4) for all subdomains i $(i = 1, \ldots, p)$ constitute a system of equations involving only the interface unknown vectors y_i. This reduced system has a natural block structure related to the interface points in each subdomain:

$$\begin{pmatrix} S_1 & E_{12} & \ldots & E_{1p} \\ E_{21} & S_2 & \ldots & E_{2p} \\ \vdots & & \ddots & \vdots \\ E_{p1} & E_{p-1,2} & \ldots & S_p \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{pmatrix} = \begin{pmatrix} g_1' \\ g_2' \\ \vdots \\ g_p' \end{pmatrix}. \tag{5}$$

The system (5) can be written as $Sy = g'$, where $y = (y_1, \ldots, y_p)^T$ is the vector of all the interface variables and $g' = (g_1', \ldots, g_p')^T$ is the right-hand side vector. The matrix S is the "global" Schur complement matrix

A key idea in domain decomposition methods is to develop preconditioners for the *global system* (1) by exploiting methods that *approximately solve the reduced system* (5). These techniques, termed "induced preconditioners" (see, e.g., [12]), can be best explained by considering a reordered version of the global system (1) in which all the internal vector components $u = (u_1, \ldots u_p)^T$ are labeled first followed by all the interface vector components y. Such reordering leads to a block system

$$\left(\begin{array}{cccc|c} B_1 & & & & F_1 \\ & B_2 & & & F_2 \\ & & \ddots & & \vdots \\ & & & B_p & F_p \\ \hline E_1 & E_2 & \cdots & E_p & C \end{array}\right) \left(\begin{array}{c} u_1 \\ u_2 \\ \vdots \\ u_p \\ \hline y \end{array}\right) = \left(\begin{array}{c} f_1 \\ f_2 \\ \vdots \\ f_p \\ \hline g \end{array}\right), \tag{6}$$

which also can be rewritten as

$$\begin{pmatrix} B & F \\ E & C \end{pmatrix} \begin{pmatrix} u \\ y \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}. \tag{7}$$

Induced preconditioners for the global system are obtained by exploiting a block LU factorization for A. Consider the factorization

$$\begin{pmatrix} B & F \\ E & C \end{pmatrix} = \begin{pmatrix} B & 0 \\ E & S \end{pmatrix} \begin{pmatrix} I & B^{-1}F \\ 0 & I \end{pmatrix}, \tag{8}$$

where S is the global Schur complement $S = C - EB^{-1}F$. This Schur complement matrix is identical to the coefficient matrix of system (5) (see, e.g., [12]). The global system (7) can be preconditioned by an approximate LU factorization constructed such that

$$L = \begin{pmatrix} B & 0 \\ E & M_S \end{pmatrix} \quad \text{and} \quad U = \begin{pmatrix} I & B^{-1}F \\ 0 & I \end{pmatrix} \tag{9}$$

with M_S being some approximation to S. The preconditioners presented next are based on the global system of equations for the Schur unknowns (system (4) or the equivalent form (5)) and on the global block LU factorization (8), or rather its approximated version (9).

For convenience, (4) is rewritten as a preconditioned system with the diagonal blocks:

$$y_i + S_i^{-1} \sum_{j \in N_i} E_{ij} y_j = S_i^{-1} \left[g_i - E_i B_i^{-1} f_i \right]. \tag{10}$$

Note that this is simply a block-Jacobi preconditioned Schur complement system. System (10) may be solved by a GMRES-like accelerator, requiring a solve with S_i at each step. This solve can be carried out using the following observation (see [12]). Let A_i have the form (2) and be factored as $A_i = L_i U_i$, where

$$L_i = \begin{pmatrix} L_{B_i} & 0 \\ E_i U_{B_i}^{-1} & L_{S_i} \end{pmatrix} \quad \text{and} \quad U_i = \begin{pmatrix} U_{B_i} & L_{B_i}^{-1} F_i \\ 0 & U_{S_i} \end{pmatrix}.$$

Then, a rather useful result is that $L_{S_i}U_{S_i}$ is equal to the Schur complement S_i associated with the partitioning (2). When an approximate factorization to A_i is available, an approximate LU factorization to S_i can be obtained canonically by extracting the related parts from the L_i and U_i matrices. In other words, *an ILU factorization for the Schur complement is the trace of the global ILU factorization on the unknowns associated with the Schur complement.* For a local Schur complement, the ILU factorization obtained in this manner leads to an approximation $\tilde{S}_i$ of the local Schur complement S_i. Instead of the exact Schur complement system (4), or equivalently (10), the following approximate (local) Schur complement system derived from (10) can be considered in each processor i:

$$y_i + \tilde{S}_i^{-1} \sum_{j \in N_i} E_{ij} y_j = \tilde{S}_i^{-1} \left[g_i - E_i B_i^{-1} f_i \right]. \tag{11}$$

The global system related to equations (11) can be solved by a Krylov subspace method, e.g., GMRES. The matrix-vector operation associated with this solve involves a certain matrix M_S (cf. equation (9)). The global preconditioner (9) can then be defined from M_S.

Given a local ILU factorization $A_i = L_iU_i + R_i$, with which the factorization $S_i = L_{S_i}U_{S_i} + R_{S_i}$ is associated, the following algorithm applies in each processor the global approximate Schur LU preconditioner to a block vector $(f_i, g_i)^T$ to obtain the solution $(u_i, y_i)^T$. The algorithm uses m iterations of GMRES without restarting to solve the local part of the Schur complement system (11). In the description of Algorithm 1, P represents the projector that maps the whole block vector $(f_i, g_i)^T$ into the subvector vector g_i associated with the interface variables.

ALGORITHM **1** *Approximate Schur-LU solution step with GMRES*

Given: local right-hand side $rhs = \binom{f_i}{g_i}$
    Define an $(m+1) \times m$ *matrix* $\bar{H}_m$ *and set* $\bar{H}_m := 0$.
Arnoldi process:
    $y_i := 0$
    $r := (L_iU_i)^{-1} rhs$
    $v_1 := Pr/\|Pr\|_2$
    For $j = 1, ..., m$ *do*
        Exchange interface vector components y_i
        $t := (L_{S_i}U_{S_i})^{-1} X_i y_{i,ext}$
        $w := v_j + t$
        For $l = 1, \ldots, j$ *Do:*
            $h_{l,j} := (w, v_l)$
            $w := w - h_{l,j} v_l$
        EndDo
        $h_{j+1,j} := \|w\|_2$ *and* $v_{j+1} := w/h_{j+1,j}$
    EndDo
    Define $V_m := [v_1, ..., v_m]$.
Form the approximate solution for interface variables:
    Compute $y_i := y_i + V_m z_m$, *where*
    $z_m = \mathrm{argmin}_z \|\beta e_1 - \bar{H}_m z\|_2$ *and* $e_1 = [1, 0, \ldots, 0]^T$.
Find other local unknowns:
    Exchange interface vector components y_i
    $t := X_i y_{i,ext}$
    $rhs := rhs - \binom{0}{t}$
    $\binom{u_i}{y_i} := (L_iU_i)^{-1} rhs$.

To use approximate-inverse techniques in the approximation of the local Schur complements, we consider such a technique proposed in [3] and [4]. In the block LU factorization (9), this sparse approximate-inverse technique can be applied to approximate $B_i^{-1}F_i$ with a certain matrix Y_i. The resulting matrix Y_i is sparse and therefore $M_{S_i} = C_i - E_iY_i$ is a sparse approximation to S_i. A further approximation can be constructed using an ILU factorization for the matrix M_{S_i}. Consider an approximated version of the exact Schur complement system (4)

$$\tilde{S}_i y_i + \sum_{j \in N_i} E_{ij} y_j = g_i - E_i B_i^{-1} f_i, \qquad (12)$$

the right-hand side of which can be also computed approximately. System (12) requires an approximation $\tilde{S}_i$ to the local Schur complement $S_i = C_i - E_i B_i^{-1} F_i$. The matrix M_{S_i} defined from the approximate-inverse technique outlined above can be used for $\tilde{S}_i$. The global preconditioner M is given by the block factorization (9) in which M_S is the approximation to S obtained by *iteratively solving system* (12).

ALGORITHM **2** *Approximate-inverse Schur complement solution step with GMRES*

1. *Given: local right-hand side* $rhs = \begin{pmatrix} f_i \\ g_i \end{pmatrix}$
2. *Solve* $B_i u_i = f_i$ *approximately*
3. *Calculate the local right-hand side* $\tilde{g}_i := g_i - E_i u_i$
4. *Use GMRES to solve the distributed system* $M_{S_i} y_i + X_i y_{i,ext} = \tilde{g}_i$
5. *Compute an approximation to* $t := B_i^{-1} F_i y_i$
6. *Compute* $u_i := u_i - t$.

In the experiments, we compared the performance of the described preconditioners and the distributed Additive Schwarz preconditioning (see, e.g., [12]) on 2-D elliptic PDE problems and on several problems with the matrices from the Harwell-Boeing and Davis collections [5]. A flexible variant of restarted GMRES (FGMRES) [12] has been used to solve the original system since this accelerator permits a change in the preconditioning operation at each step. Thus, it is possible to use ILUT-preconditioned GMRES with `lfil` fill-in elements. For convenience, the following abbreviations will denote preconditioners and solution techniques used in the numerical experiments:

SAPINV Distributed approximate block LU factorization: M_{S_i} and $B_i^{-1} F_i$ are approximated using the matrix Y_i, constructed using the approximate-inverse technique described in [3];

SAPINVS Distributed approximate block LU factorization: M_{S_i} is approximated using the approximate-inverse technique in [3], but $B_i^{-1} F_i$ is applied using one matrix-vector multiplication followed by a solve with B_i;

SLU Distributed global system preconditioning defined via an approximate solve with M_S, in which $S_i \approx L_{S_i} U_{S_i}$;

BJ Approximate Additive Schwarz, where ILUT-preconditioned GMRES(k) is used to precondition each submatrix assigned to a processor;

SI "pure" Schur complement iteration, which solves the original system by solving the Schur complement system.

A comparison of timing results and iteration numbers for a global 360×360 mesh mapped to (virtual) square processor grids of increasing size is given in Figure 1. (In Figure 1, we omit the solution time for the BJ preconditioning, which is 95.43 seconds). The residual norm reduction by 10^{-6} was achieved by FGMRES(10). In preconditioning, ILUT with $lfil = 15$ and the dropping tolerance 10^{-4} was used as a choice of an incomplete LU factorization. Five iterations of GMRES with the relative tolerance 10^{-2} were used in the application of BJ and SLU. For SAPINV, forward-backward solves with $B_i \approx L_{B_i} U_{B_i}$ were performed in Line 2 of Algorithm 2.

Timing results for fixed local subproblem sizes of 15×15 and 70×70 are presented in Figure 2. (Premature termination of the curves for SI indicates nonconvergence in 300 iterations).

Table 1 describes three test problems from the Harwell-Boeing and Davis collections. The column Pattern specifies whether a given problem has a structurally symmetric matrix. In all three test problems, the matrix rows followed by the columns were scaled by 2-norm. Also, in the partitioning of a problem one level of overlapping with data exchanging was used following [8]. Tables 2–4 show iteration numbers required by FGMRES(20) with SAPINV, SAPINVS, SLU, and BJ till convergence on different numbers of processors. An asterisk indicates nonconvergence. In the preconditioning phase, approximate solves in

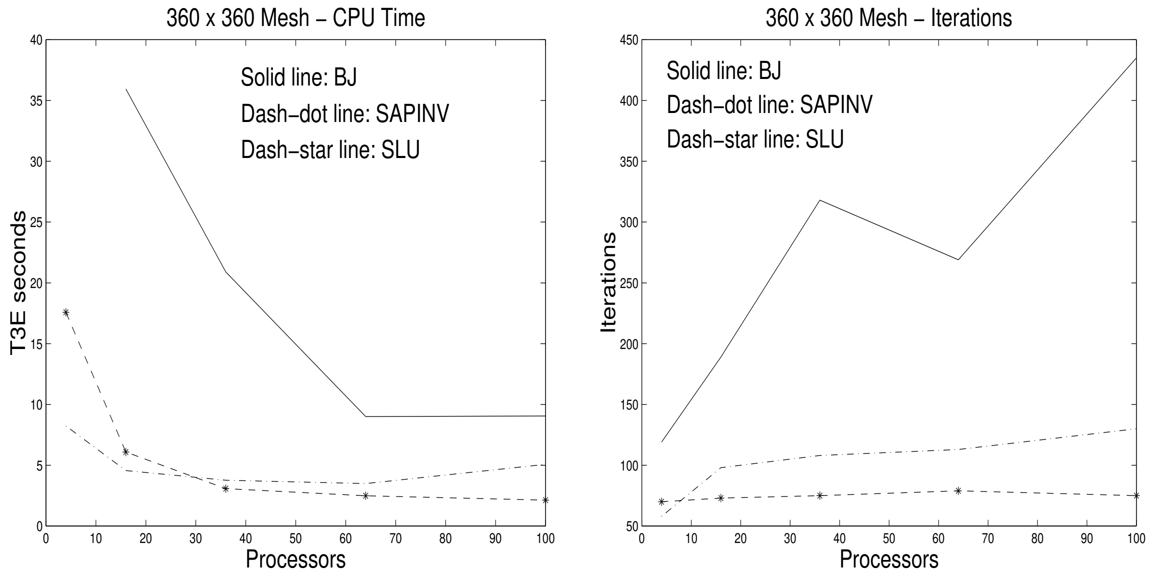

Figure 1: Times and iteration counts for solving a 360×360 discretized Laplacean problem with 3 different preconditioners using flexible GMRES(10).

Name	n	n_z	Pattern	Discipline
af23560	23560	484256	Symm	Airfoil, eigenvalue calculation
raefsky1	3242	22316	Unsymm	Incompressible flow in pressure driven pipe
sherman3	5005	20033	Symm	Oil reservoir modeling

Table 1: Description of test problems

each processor were carried out by GMRES to reduce the residual norm by 10^{-3} but no more than for five steps were allowed. As a choice of ILU factorization, ILUT with `lfil` fill-in elements (shown in the column `lfil`) was used in the experiments here. `lfil` corresponds also to the number of elements in a matrix column created by the approximate-inverse technique.

Several preconditioning techniques for distributed linear systems are derived from approximate solutions with the related Schur complement system. The preconditioners are built upon the already available distributed data structure for the original matrix, and an approximation to the global Schur complement is *never* formed explicitly. Thus, no communication overheard is incurred to construct a preconditioner, making the preprocessing phase simple and highly parallel. The preconditioning operations utilize the communication structure precomputed for the original matrix.

The preconditioning to the global matrix A is defined in terms of a block LU factorization which involves a solve with the global Schur complement system at each preconditioning step. This system is in turn solved approximately with a few steps of GMRES exploiting approximations to the local Schur complement for preconditioning. Two different techniques, incomplete LU factorization and approximate-inverse, are used to approximate these local Schur complements. Distributed preconditioners constructed and applied in this manner allow much flexibility in specifying approximations to the local Schur complements and local system solves and in defining the global induced Block-LU preconditioner to the original matrix.

With an increasing number of processors, a Krylov subspace method, such as FGMRES, preconditioned by the proposed techniques exhibits a very moderate growth in the execution time for scaled problem sizes. Experiments show that the proposed distributed preconditioners based on Schur complement techniques are superior to the commonly used Additive Schwarz preconditioning. In addition,

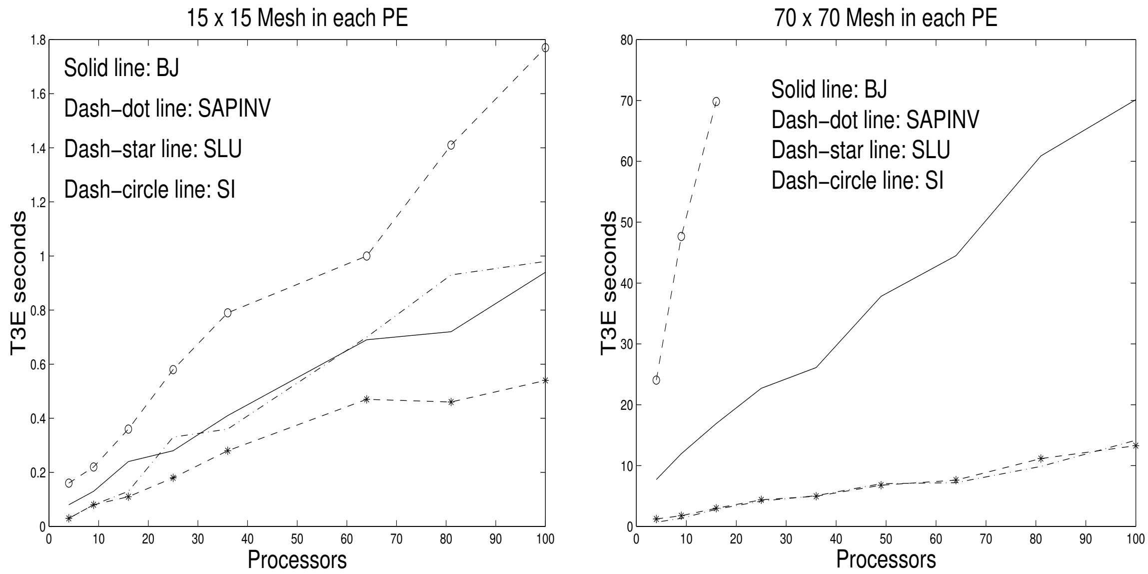

Figure 2: Solution times for a Laplacean problem with various local subproblem sizes using FGMRES(10) with 3 different preconditioners (BJ, SAPINV, SLU) and the Schur complement iteration (SI).

Name	Precon	`lfil`	4	8	16	24	36	40
raefsky1	SAPINV	10	14	13	10	11	8	8
		20	12	11	9	9	8	8
	SAPINVS	10	16	13	10	11	8	8
		20	13	11	9	9	8	8
	SLU	10	215	197	198	194	166	171
		20	48	50	40	42	41	41
	BJ	10	85	171	173	273	252	263
		20	82	170	173	271	259	259

Table 2: Number of FGMRES(20) iterations for the RAEFSKY1 problem.

Name	Precon	`lfil`	16	24	32	40	56	64	80	96
af23560	SAPINV	20	32	36	27	29	73	35	71	61
		30	32	35	23	29	46	60	33	52
	SAPINVS	20	32	35	24	29	55	35	37	59
		30	32	34	23	28	43	45	23	35
	SLU	20	81	105	94	88	90	76	85	71
		30	38	34	37	39	38	39	38	35
	BJ	20	37	153	53	60	77	80	95	*
		30	36	41	53	57	81	87	97	115

Table 3: Number of FGMRES(20) iterations for the AF23560 problem.

Name	Precon	`lfil`	16	24	32	40	48	56	60
sherman3	SAPINV	10	52	42	21	17	31	21	20
		20	54	40	18	17	29	20	19
	SAPINVS	10	54	48	21	18	33	21	21
		20	55	51	19	16	30	19	19
	SLU	10	34	32	16	20	20	23	17
		20	21	23	12	16	15	18	13
	BJ	10	158	155	72	208	110	122	84
		20	163	155	72	206	111	120	85

Table 4: Number of FGMRES(20) iterations for the SHERMAN3 problem.

this advantage comes at no additional cost in code-complexity or memory usage, since the same data structures as those for additive Schwarz preconditioners can be used.

References

[1] S. Balay, L. Curfman McInnes, W. D. Gropp, and B. F. Smith. PETSc 2.0 users manual. Technical Report ANL-95/11, Argonne National Lab, Argonne, IL, 1995.

[2] T. Barth, T. F. Chan, and W.-P. Tang. A parallel algebraic non-overlapping domain decomposition method for flow problems. Technical report, NASA Ames Research Center, 1997. In preparation.

[3] E. Chow and Y. Saad. Approximate inverse preconditioners for general sparse matrices. *SIAM Journal on Scientific Computing*, –, 1997. To appear.

[4] E. Chow and Y. Saad. Approximate inverse techniques for block-partitioned matrices. *SIAM Journal on Scientific Computing*, 1997. To appear.

[5] I. S. Duff, R. G. Grimes, and J. G. Lewis. Sparse matrix test problems. *ACM Transactions on Mathematical Software*, 15:1–14, 1989.

[6] V. Eijkhout and T. Chan. ParPre a parallel preconditioners package, reference manual for version 2.0.17. Technical Report CAM Report 97-24, UCLA, 1997.

[7] Scott A. Hutchinson, John N. Shadid, and R. S. Tuminaro. Aztec user's guide. version 1.0. Technical Report SAND95-1559, Sandia National Laboratories, Albuquerque, NM, 1995.

[8] G.-C. Lo and Y. Saad. Iterative solution of general sparse linear systems on clusters of workstations. Technical Report UMSI 96/117 & UM-IBM 96/24, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, 1996.

[9] S. Ma and Y. Saad. Distributed ILU(0) and SOR preconditioners for unstructured sparse linear systems. Technical Report 94–027, Army High Performance Computing Research Center, University of Minnesota, Minneapolis, MN, 1994.

[10] Y. Saad. Krylov subspace methods in distributed computing environments. Technical Report 92-126, Army High Performance Computing Research Center, Minneapolis, MN, 1992.

[11] Y. Saad. Krylov subspace methods in distributed computing environments. In M. Hafez, editor, *State of the Art in CFD*, pages 741–755, 1995.

[12] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS publishing, New York, 1996.

[13] Y. Saad and A. Malevsky. PSPARSLIB: A portable library of distributed memory sparse iterative solvers. In V. E. Malyshkin et al., editor, *Proceedings of Parallel Computing Technologies (PaCT-95), 3-rd international conference, St. Petersburg, Russia, Sept. 1995*, 1995.