

Coarse Grid Acceleration of Parallel Incomplete Factorization Preconditioners

Y. Notay * Antoine Van de Velde †

Service de Métrologie Nucléaire

Université Libre de Bruxelles (C.P. 165)

50, Av. F.D. Roosevelt, B-1050 Brussels, Belgium.

email : ynotay@ulb.ac.be

Abstract

In this paper, we address the problem of finding good parallel preconditioners for the iterative solution of large sparse linear systems arising from discrete second order elliptic PDEs.

In this view, we consider approximate factorization preconditioners, and propose an ordering strategy which enables parallelism in the solution of the triangular systems, and for which only a limited deterioration in the conditioning properties is observed.

Next, we show that this deterioration can be prevented by adding a corrector term based on the exact solution of the system projected on a very coarse finite element basis. By “very”, we mean that the size of the basis need not increase with the number of processors.

keywords: iterative methods for linear systems, parallel computing, preconditioning.

AMS CLASSIFICATION : 65F10, 65B99, 65N20.

1 Introduction

We consider the iterative solution of large sparse symmetric positive definite linear systems

$$Ax = b. \quad (1.1)$$

For such systems, the conjugate gradient acceleration is known as a powerful technique, provided that efficient preconditioning can be used (see e.g. [2]).

*Supported by the “Fonds National de la Recherche Scientifique”, Chercheur qualifié.

†Present address : SIEMENS AG/KWU - BT 12, Bunsenstr. 43, Raum 3145, D-91058 Erlangen, Germany.



The preconditioner B is a symmetric positive definite matrix such that, on the one hand, solving a system with B , to be done at each iteration, is much easier than solving (1.1), while, on the other hand, the eigenvalue distribution of the preconditioned system matrix $B^{-1}A$, which determines the convergence rate, is better than that of the original system matrix A .

A given eigenvalue distribution is favorable when, a.o., the spectral condition number

$$\kappa(B^{-1}A) = \frac{\nu_{\max}(B^{-1}A)}{\nu_{\min}(B^{-1}A)} \quad (1.2)$$

(i.e. the ratio of the largest to the smallest eigenvalue), is small. Indeed, the number of iteration k_ϵ is bounded above by

$$\kappa_\epsilon \leq \frac{1}{2} \sqrt{\kappa(B^{-1}A)} \ln \frac{2}{\epsilon} + 1, \quad (1.3)$$

where ϵ is the relative error in norm $(\cdot, A)^{1/2}$. However, this bound is much too pessimistic when the eigenvalue distribution between ν_{\min} and ν_{\max} is not dense and interesting convergence rates may still be derived when a few eigenvalues are isolated at either end of the spectrum, provided that its dense part is well clustered (see e.g. [1, 4, 17]).

For systems arising from the discretization of second order discrete elliptic PDEs, it is common to use preconditioners obtained by computing an approximate factorization of the system matrix, see e.g. [2, 1]. The preconditioner then writes

$$B = (P - F^t)P^{-1}(P - F), \quad (1.4)$$

where P is diagonal, with positive diagonal entries, and F strictly upper triangular. The triangular factors are to be kept sparse, and a widespread choice consists in allowing F to be nonzero only in positions where A itself is nonzero (i.e. no fill in). For five point finite difference matrices, the setting of the offdiagonal part reduces then to

$$F + F^t = -\text{offdiag}(A). \quad (1.5)$$

The diagonal part P is generally computed so that either $\text{diag}(B - A) = 0$ (i.e. standard Incomplete Cholesky (IC) factorization) or $Ae = Be$ where $e = (1, \dots, 1)$ (i.e. Modified Incomplete Cholesky (MIC) factorization), or according some compromise between the preceding choices (see e.g. [1, 3, 7, 18]). The latter techniques, which are generally found more efficient than IC and more robust than MIC, are often referred to as "perturbed" modified incomplete factorizations.

On a sequential computer, (1.5) implies that solving a system with B requires approximately the same computing time as a multiplication by the system matrix A . On parallel computers however, one has to face the problem that the associated computations are intrinsically recursive. Indeed, the basic algorithm for solving $(P - F^t)v = r$ is (forward substitution):

$$v_i = p_{ii}^{-1} \left(r_i - \sum_{j < i} f_{ji} v_j \right), \quad i = 1, \dots, n \quad (1.6)$$



(where $P = (p_{ii} \delta_{ij})$ and $F = (f_{ij})$). Hence, computing v_i requires to have previously computed v_j for all $j < i$ such that $f_{ji} \neq 0$; and, for a given nonzero pattern, the level of parallelism in this algorithm depends on the ordering.

Incomplete factorization preconditioners have well established conditioning properties with respect to natural orderings, but these are unfortunately not very suitable for parallelism. For instance, it turns out from the analysis in [14] that a satisfactory parallelization may be obtained only for relatively small numbers of processors, and provided that the communication cost for small messages is not prohibitive.

Besides, several ordering schemes have been developed with the seek of improving the conditioning, and some of them exhibit in addition a high level of intrinsic parallelism, see e.g. [8, 12, 21]. However, to exploit this parallelism, one has to distribute the unknowns on the different processors in a very inefficient way, and even the multiplication by the system matrix requires then much communications on distributed memory computers.

On such computers, communications are kept minimal if one assigns the unknowns to the different processors on the basis of a partitioning of the domain in which the PDE is solved. The parallelization of the forward and backward substitutions requires then essentially that one leaves uncoupled the unknowns belonging to different subdomains.

This may be obtained by zeroing the entries in F connecting unknowns assigned to different processors. The effect on the convergence rate appears however quite dramatic, even for moderate number of processors [26].

In this paper, we propose to obtain the needed parallelism in the solution of the triangular systems by using a dedicated ordering strategy which generalizes van der Vorst four processor ordering (see [10]) to arbitrary numbers of processors.

We analyze the effect of this ordering strategy on the eigenvalue distribution associated to perturbed modified incomplete factorization preconditioners, and show that their convergence properties deteriorate only slightly as the number of processors increases.

Next, we consider adding to the preconditioner a corrector term based on the exact solution of the system projected on a coarse finite element basis. That is, if q is the size of this finite element basis, we consider as preconditioner the matrix \hat{B} , such that

$$\hat{B}^{-1} = B^{-1} + V(V^t A V)^{-1} V^t, \quad (1.7)$$

where B is our incomplete factorization preconditioner (1.4) and $V = [v_1, \dots, v_q]$ the $n \times q$ matrix whose columns v_i are the vectors which interpolate on the grid the coarse finite element basis functions.

We show that this allows to compensate for the deterioration of the convergence associated with the parallel orderings. This conclusion is obtained without increasing the size q of the finite element basis together with the number of processors.

Since the existence of incomplete factorization preconditioners is only guaranteed for Stieltjes matrices, we shall assume in the remaining of the paper that the system matrix is a Stieltjes matrix, i.e. that it is positive definite with nonpositive offdiagonal entries. Note however that only the matrix to factorize has to be such, and that it



may be any relevant (i.e. "spectrally equivalent") approximation of the system matrix. Techniques to derive such approximations are given in [1, 2, 9].

The remainder of this paper is organized as follows: in Section 2, we present our parallel ordering strategy, and analyze its influence on the eigenvalue distribution; some algebraic results for preconditioners of the form (1.7) are derived in Section 3, and their use in the context of parallel incomplete factorizations analyzed in Section 4.

General terminology and notation. Unless otherwise stated, all vectors belong to C^n ; all matrices are $n \times n$ real matrices. The symbols A^t and $\mathcal{R}(A)$ denote respectively the transposes and the range of the matrix A . The order relation between real matrices and vectors is the usual componentwise order: if $A = (a_{ij})$ and $B = (b_{ij})$ then $A \leq B$ ($A < B$) if $a_{ij} \leq b_{ij}$ ($a_{ij} < b_{ij}$) for all i, j ; A is called nonnegative (positive) if $A \geq 0$ ($A > 0$). If $A = (a_{ij})$, we denote by $diag(A)$ the (diagonal) matrix whose entries are $a_{ii}\delta_{ij}$ and we let $offdiag(A) = A - diag(A)$. $e = (1 \dots 1)^t$ is the vector with all components equal to unity.

$span\{v_1, \dots, v_q\} = \mathcal{R}(V)$ where $V = [v_1, \dots, v_q]$ denotes the set of vectors spanned by the vectors v_i , $i = 1, \dots, q$.

For any $n \times n$ diagonalizable matrix C with real eigenvalues, $\nu_i(C)$ denotes its i^{th} eigenvalue in increasing order: $\nu_1(C) \leq \nu_2(C) \leq \dots \leq \nu_n(C)$; sometimes, $\nu_{\min}(C)$ is used for the smallest eigenvalue $\nu_1(C)$, and $\nu_{\max}(C)$ for the largest $\nu_n(C)$.

2 A parallel ordering strategy

The ordering strategy we propose is a generalization of van der Vorst ordering for a 2×2 processor grid [10]. This ordering and its generalization to 3×2 , 4×2 and 4×4 processor grids are illustrated on Fig. 1 for a square five point mesh. On this figure, we have depicted the graph associated in each case to the system matrix A , representing each node i by a circle or a square, and each nonzero entry, a_{ij} $i \neq j$, by an arrow pointing from the smallest of the indices i, j to the largest.

We give this representation rather than an explicit numbering because it is perhaps more clear while, as far as E and F have same nonzero pattern as A (factorization without fill-in), this dependency structure completely determines the preconditioner independently of the particular numbering which is used.

With Fig. 1, one may see how parallelism can take place during the forward and backward substitutions. For instance, in the 2×2 processor ordering, the nodes represented by a filled circle delimit four regions in which a forward solve (1.6) may be started independently; then, communication is needed to perform the substitutions related to these "separator" nodes. In the other illustrated orderings, there is another type of "separator" line, whose nodes are marked by a square, and along which one has to start any forward substitution; next, the computation may be proceeded independently in the boxes delimited by these lines and the nodes represented by a square; the final update for these latter nodes is similar as in the 2×2 processor ordering.

The generalization of this principle for more and more processors presents no



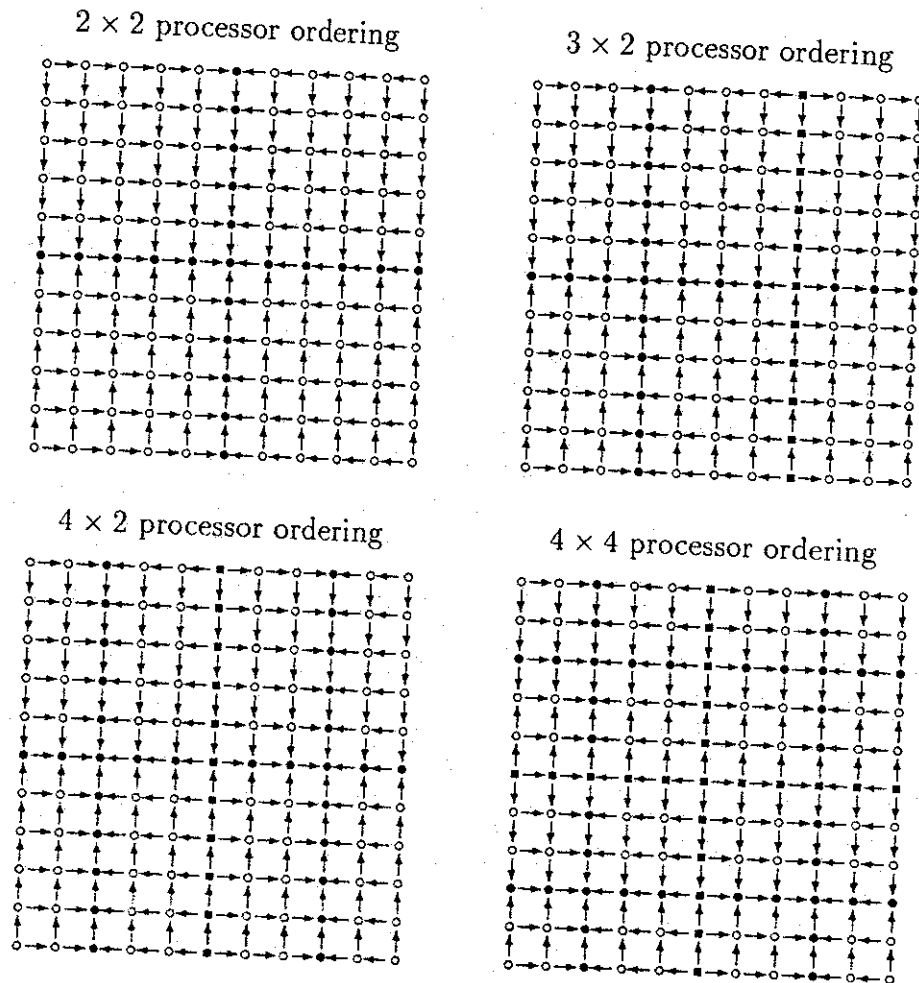


Figure 1: Illustration of the ordering strategy for a 11×11 five point grid

difficulties provided that the grid has enough nodes.

One may object that implementing such an algorithm in an effective parallel program is not an obvious task, and requires at least a much harder technical work than just zeroing entries connecting nodes on different processors. For instance, following Ortega [22] the separator nodes are either assigned to one of the neighbor processors, and one loses then some of the natural parallelism in the algorithm, or treated separately, and one may get then a sequential bottleneck.

Being aware of this difficulty, we refer to [20, 19] where the problem is analysed in detail and where an elegant easy-to-implement parallel algorithm without bottleneck is obtained by assigning the separator nodes to each processor to which they are connected, and repeating in parallel the concerned calculation.

In this paper, we shall therefore concentrate on the analysis of the convergence properties associated to this kind of ordering, assuming that the implementation problem has been solved either by using the algorithm in [20, 19] or by a dedicated technical work. We refer also to [19] for timing results on parallel computers.

Let us point out that the above mentioned algorithm has been successfully adapted



to SIMD computers by Ould Amar [23]. This will motivate considering very large processor grids in the present study.

Note also that the ordering strategy presented above can be easily extended to 3D grids. Some results for 3D problems are given in [19], from which one may conclude that the situation is more favorable than in 2D, the use of very large processor grids entailing only a marginal increase of the number of iterations.

In the remaining of this paper, we shall concentrate on the discussion of 2D problems.

Influence on the eigenvalue distribution

As said in the introduction, we deal essentially with "perturbed" modified incomplete factorizations. Indeed, the standard IC method cannot compete with these techniques for moderate to large problems (see e.g. [18]), while "unperturbed" MIC (i.e. factorizations for which $Be = Ae$) may yield a singular preconditioner for the kind of ordering considered here [15].

In the earliest works on these "perturbed" methods (see [7] for a survey), the diagonal P of the triangular factors was computed so that $Be = Ae + \Lambda e$, where $\Lambda = (\lambda_i, \delta_{ij})$ was a diagonal matrix whose diagonal entries were small nonnegative "perturbation" parameters. Much research focussed then on the design of factorization algorithms which automatically provide appropriate perturbations (see [2, 7, 18] and the references therein). This may be illustrated by the algorithm associated to the DRIC method, which we chose for the numerical tests because of its particular robustness for solving discrete elliptic PDEs [18, 24]:

$$\text{for } i = 1, \dots, n$$

$$f_{ij} = \begin{cases} -a_{ij} - \sum_{k < i} \frac{f_{ki} f_{kj}}{p_{kk}} & \text{if } (i, j) \in J \\ -a_{ij} & \text{otherwise} \end{cases}$$

$$j = i + 1, \dots, n$$

$$p_{ii} = a_{ii} - \sum_{k < i} \frac{f_{ki}^2}{p_{kk}} - \sum_{k < i} \sum_{\substack{j > k, j \neq i \\ (i, j), (j, i) \notin J}} \omega_k \frac{f_{ki} f_{kj}}{p_{kk}}$$

$$\text{with } \omega_k = \min \left(\frac{2(1 - \alpha) p_{kk}}{\sum_{j > k} f_{kj}} - 1, 1 \right)$$

In this algorithm, J is the set of position (i, j) , $j > i$, in which fill-in is permitted in the upper triangular factor, and α is an input parameter.

We observe that letting $\omega_k \equiv 0$ in this algorithm would lead to standard IC, whereas $\omega_k \equiv 1$ would lead to unperturbed MIC. The DRIC algorithm may be consequently seen as dynamically modulating the "modification" to IC in function of α (the corresponding perturbations to MIC are not explicitly computed). This is but internal to the algorithm and the user has only to specify the α parameter. If



$A = (a_{ij})$ is a Stieltjes matrix with $Ae \geq 0$, it can be shown [18] that all computed p_{ii} are positive (i.e. B (1.4) is symmetric positive definite) while

$$\nu_{\max}(B^{-1}A) \leq \alpha^{-1}. \quad (2.1)$$

Hence, the resulting eigenvalue distribution depends mainly on the lowest eigenvalues behaviour in function of α .

Several works [2, 6, 13, 16] have been dedicated to the analysis of the lower end of the spectrum associated to these "perturbed" modified incomplete factorization methods. Actually these works are more particularly concerned with the method referred to as DMIC in [18], or some of its earlier version, and none of them directly address the case of the DRIC method which is more recent.

However, it is shown in [18] that both these methods behave very similarly when applied to discrete second order elliptic PDEs with isotropic coefficients, whereas the behaviour of DRIC is better in presence of anisotropies.

Hence, we may consider that the above mentioned results apply a fortiori to DRIC too. These works do not result in a general algebraic lower bound, but each of them proposes a procedure which, carefully applied, leads to a relatively sharp lower bound. For 2D discrete second order elliptic PDEs, the latter writes

$$\nu_{\min}(B^{-1}A) \geq \frac{1}{1 + c_1 \alpha^2 n + c_2 \alpha \sqrt{n}}, \quad (2.2)$$

where c_1, c_2 are independent of the mesh refinement. c_1, c_2 are in addition not much larger than 1 provided that one uses a natural ordering of the unknowns and that the unpreconditioned system presents no "degenerate" eigenvalues, i.e. provided that $n \nu_{\min}(D^{-1}A)$ where $D = \text{diag}(A)$ is not much smaller than 1.

Taking the ratio of (2.1) and (2.2), one get

$$\kappa(B^{-1}A) \leq \alpha^{-1} + c_1 \alpha \sqrt{n} + c_2 \sqrt{n}, \quad (2.3)$$

which is minimal for

$$\alpha^{-1} \simeq \sqrt{n}. \quad (2.4)$$

$\kappa(B^{-1}A)$ is then one order of magnitude better than the original conditioning $\kappa(D^{-1}A) \simeq cn$.

The case of "degenerate" eigenvalues in the original system matrix is analysed in detail in [16]. There, it is shown that, if there are $q - 1$ such nodes ($q \geq 1$), one may still derive a lower bound of the type (2.2) with c_1, c_2 not much larger than 1, but which bounds below the q^{th} eigenvalue in increasing order $\nu_q(B^{-1}A)$ instead of the first one $\nu_1(B^{-1}A)$.

Hence, the eigenvalue distribution is similar as in other cases, apart from $q - 1$ small isolated eigenvalues, and one may rely on the superlinear convergence properties of the conjugate gradients to still get acceptable convergence.

It is interesting to recall here how this superlinear convergence takes place in such cases. A thorough explanation is given in van der Sluis and van der Vorst [25]. Briefly stated, the convergence is initially slow, as one would expect from the bound (1.3).



However, after some delay, the $q - 1$ smallest Ritz values of the companion Lanczos process have converged to the $q - 1$ smallest eigenvalues, and one observes that rapid convergence takes place, all happening as if the corresponding modes were removed from the spectrum.

A generalization of the bound (1.3) to such cases is obtained in [17] (see also [1, 3]):

$$k_\varepsilon \leq \frac{1}{2} \sqrt{\kappa^{(q)}(B^{-1}A)} \left(\ln \frac{2}{\varepsilon} + \sum_{i=1}^{q-1} \left(\ln \frac{\nu_q}{\nu_i} + 2 \right) \right) + q \quad (2.5)$$

where

$$\kappa^{(q)}(B^{-1}A) = \frac{\nu_n(B^{-1}A)}{\nu_q(B^{-1}A)} \quad (2.6)$$

is the reduced spectral condition number. Hence,

$$\frac{1}{2} \sqrt{\kappa^{(q)}} \left(\ln \frac{\nu_q}{\nu_i} + 2 \right) + 1$$

is an upper bound for the necessary delay to remove the i^{th} mode. It worths note that it is proportional to the square root of the reduced condition number, and depends only logarithmically on the gap ratio ν_q/ν_i . Therefore, perturbed modified incomplete factorization preconditioners, which improve this reduced spectral condition number by an order of magnitude, reduce dramatically these delays even though the gap ratios are nearly unchanged.

Note also that (2.5) is valid in presence of rounding errors, in spite of the well known losses of orthogonality that occur during the conjugate gradient iterations [11, 17].

We have recalled in detail these results because, as discussed in [16], they also apply in the case of non natural orderings. However, in such cases, c_2 in (2.2) is an increasing function of the number of interior nodes which have more neighbours with larger indices than neighbours with smaller indices.

There are no such nodes for both the natural and the 2×2 processor ordering. With the 2×3 , 2×4 and 4×4 processor orderings depicted on Fig. 1, respectively one, one and two lines of such nodes appear, which correspond to the nodes represented by a square on the figure. More generally, for an ordering corresponding to a $p_x \times p_y$ processor grid one will have $\ell_f = \text{int} \left(\frac{p_x - 1}{2} \right) + \text{int} \left(\frac{p_y - 1}{2} \right)$ lines of such nodes. Hence, one has to expect some decrease of the lowest eigenvalues as the number of processor increases.

This decrease is however not that dramatic. Following thoroughly the procedure in [16], one would deduce that c_2 increases as $\mathcal{O}(\ell_f)$, i.e. that the lowest eigenvalues decreases at most as $\mathcal{O}((\ell_f)^{-1}) = \mathcal{O}(p^{-\frac{1}{2}})$ in the case of a square processor grid with p processors.

Incidentally this result shows that (approximately) square processor grids, which on the other hand minimize the number of interface nodes, are also the most advantageous ones from the conditioning point of view, at least within the framework of the methods considered here.



Another interesting remark is that the α -independent term in (2.3) tends to dominate as the number of processors becomes larger. Hence, although the bound is still minimal for approximately the same value (2.4), there is now a wide interval around this value for which the bound (2.3) is practically equal to its minimal value, and inside which there may be some room for an empirical optimization. This was done in [23] for very large square processor grids, with the conclusion that up to 20-25% iterations could be saved by exchanging the rule (2.4) for

$$\alpha^{-1} \simeq \sqrt{\frac{4n}{p}}. \quad (2.7)$$

We find now interesting to corroborate these conclusions with some numerical experiments. In this view, we consider the PDE

$$-\nabla a \nabla u = f \quad \text{in } \Omega = (0, 1) \times (0, 1) \quad (2.8)$$

with

Problem A :

$$a = \begin{cases} 100 & \text{in } (\frac{1}{3}, \frac{2}{3}) \times (\frac{1}{3}, \frac{2}{3}) \\ 1 & \text{elsewhere} \end{cases}$$

$$f = \begin{cases} 100 & \text{in } (\frac{1}{3}, \frac{2}{3}) \times (\frac{1}{3}, \frac{2}{3}) \\ 0 & \text{elsewhere} \end{cases}$$

$$\begin{cases} u = 0 & \text{for } 0 \leq x \leq 1, y = 0 \\ \partial_n u = 0 & \text{on the remaining part of the boundary} \end{cases}$$

Problem B :

$$a = \begin{cases} 10^{-3} & \text{in } (\frac{1}{12}, \frac{7}{12}) \times (\frac{1}{12}, \frac{7}{12}) \\ 1 & \text{elsewhere} \end{cases}$$

$$f = \begin{cases} 1 & \text{in } (\frac{1}{12}, \frac{7}{12}) \times (\frac{1}{12}, \frac{7}{12}) \\ 0 & \text{elsewhere} \end{cases}$$

$$\begin{cases} u = 0 & \text{for } 0 \leq x \leq 1, y = 1 \text{ and } 0 \leq y \leq 1, x = 0 \\ \partial_n u = 0 & \text{on the remaining part of the boundary} \end{cases}$$

In each case, we use five point finite differences with uniform mesh size h in both directions.

In these experiments, we ran the conjugate gradient algorithm until $\| r_k \| / \| r_0 \| < 10^{-7}$, where r_k denotes the residual at iteration k . Then, we computed the Ritz values of the companion Lanczos process (that is the roots of the polynomial generated by the conjugate gradient algorithm, see [25] or [11, Section 2], for instance). At that stage, the extreme Ritz values have converged to the isolated eigenvalues at either end



of the spectrum, while the dense part of the latter is well fitted by the interval(s) in which many Ritz values are concentrated. Hence, although the set of Ritz values is far from representing exactly the whole spectrum, their distribution provides sufficient information about the eigenvalue distribution to discuss the associated convergence properties.

These Ritz value distributions are plotted on Fig. 2 and 3 for respectively problems A and B with $h^{-1} = 96, 192$. Several orderings are considered for the DRIC preconditioner, a " $p_x \times p_y$ processor ordering" meaning an ordering as illustrated on Fig. 1, for which the "separator" nodes are the nodes corresponding to gridpoints on the lines $x = \frac{j}{p_x}, j = 1, \dots, p_x - 1$ and the lines $y = \frac{j}{p_y}, j = 1, \dots, p_y - 1, j$ even giving the separator nodes ordered first (represented by a filled circle on the Figure) and j odd the separator nodes ordered last (represented by a square).

We used in each case $\alpha = h$ in agreement with the standard rule (2.4). The Jacobi preconditioning is included in order to give an idea of the eigenvalues of $D^{-1}A$. To facilitate the discussion, the latter Ritz values have been multiplied by n .

One may check that for both the natural and the 2×2 processor ordering, the number of eigenvalues associated to DRIC that are much smaller than 1 is equal to the number of eigenvalues of $D^{-1}A$ much smaller than n^{-1} , while the lower end of the spectrum shifts to the left as the number of processors increases. The estimate $\nu_{\min} = \mathcal{O}(p^{-\frac{1}{2}})$ is however by far much too pessimistic. Nevertheless, this benefit is partially compensated by the increase of the largest eigenvalues, which is however perfectly mastered thanks to the upper bound (2.1). Finally, the comparison of the 8×8 processor ordering with the 2×32 and the 32×2 ones confirms our preference for square processor grids.

Besides these tests, we found interesting to fix the ordering and let α vary. The resulting Ritz values distribution are given on Fig. 4 for the two same problems with the 16×16 processor ordering. For easier comparison, all distributions have been scaled such that the maximal value is 1. $\alpha^{-1} = h^{-1}$ corresponds to the rule (2.4) and $\alpha^{-1} = h^{-1}/8$ to the rule (2.7).

At first sight, the spectra are very similar, in agreement with the analysis above. However, the reduced spectral condition numbers, in particular $\kappa^{(4)}$ for Problem A and $\kappa^{(3)}$ for Problem B, are significantly smaller for $\alpha^{-1} = h^{-1}/8$ than for other values of the parameter. Hence, the "superlinear" convergence should take place more rapidly and result in a faster convergence for $\alpha^{-1} = h^{-1}/8$. On this basis, we may follow Ould Amar [23], and advice the rule (2.7) in case of many processor orderings (with square processor grids).

Note that $h^{-1} = 192$ does not correspond to a very fine grid for which massive parallelism is viable. However, the figures derived for for $h^{-1} = 96$ and $h^{-1} = 192$ are very similar, apart from the increase of the largest eigenvalues as predicted by the theory. Hence, we believe that the conclusion above are essentially independent of the mesh size. They are in addition corroborated by the iteration count given in [19, 23] for much finer grids.

This remark also holds for the numerical results in Section 4.



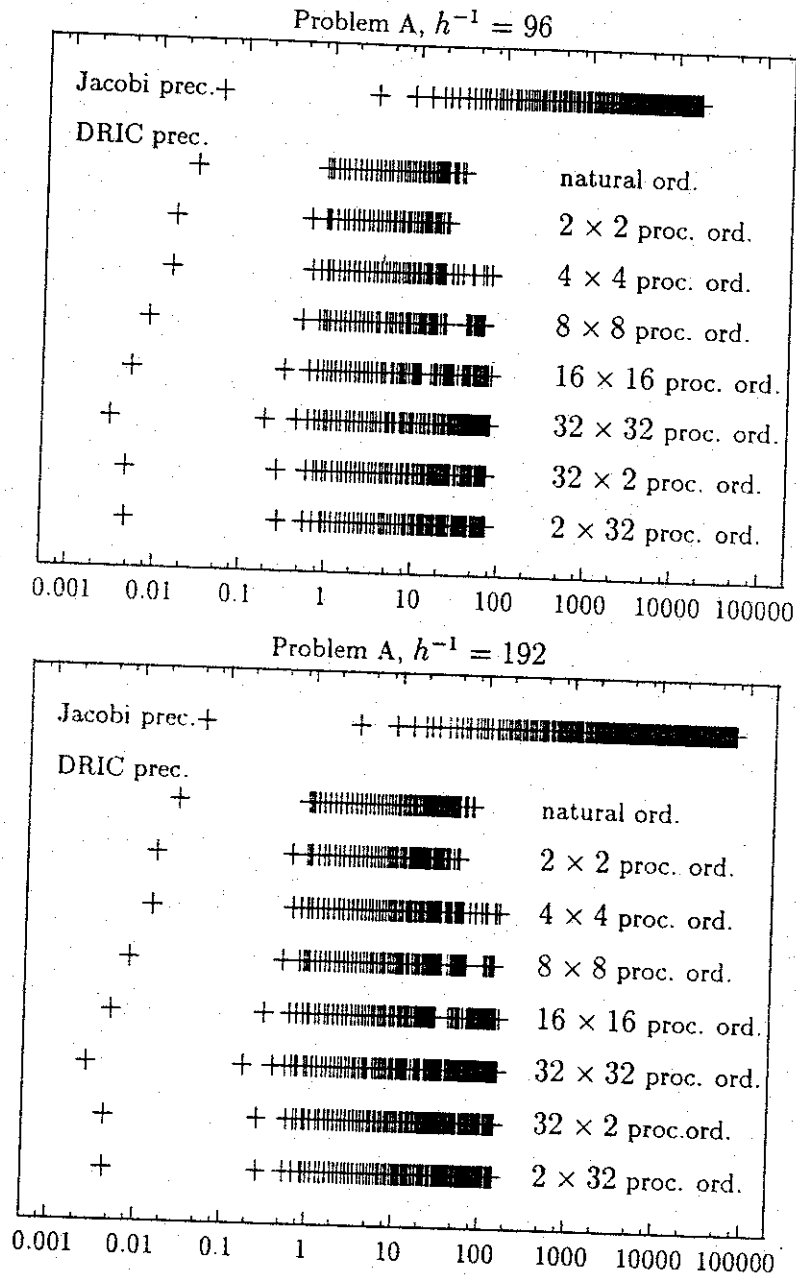


Figure 2: Ritz values for the DRIC preconditioning with $\alpha = h$, and Ritz values times n for the Jacobi preconditioning.



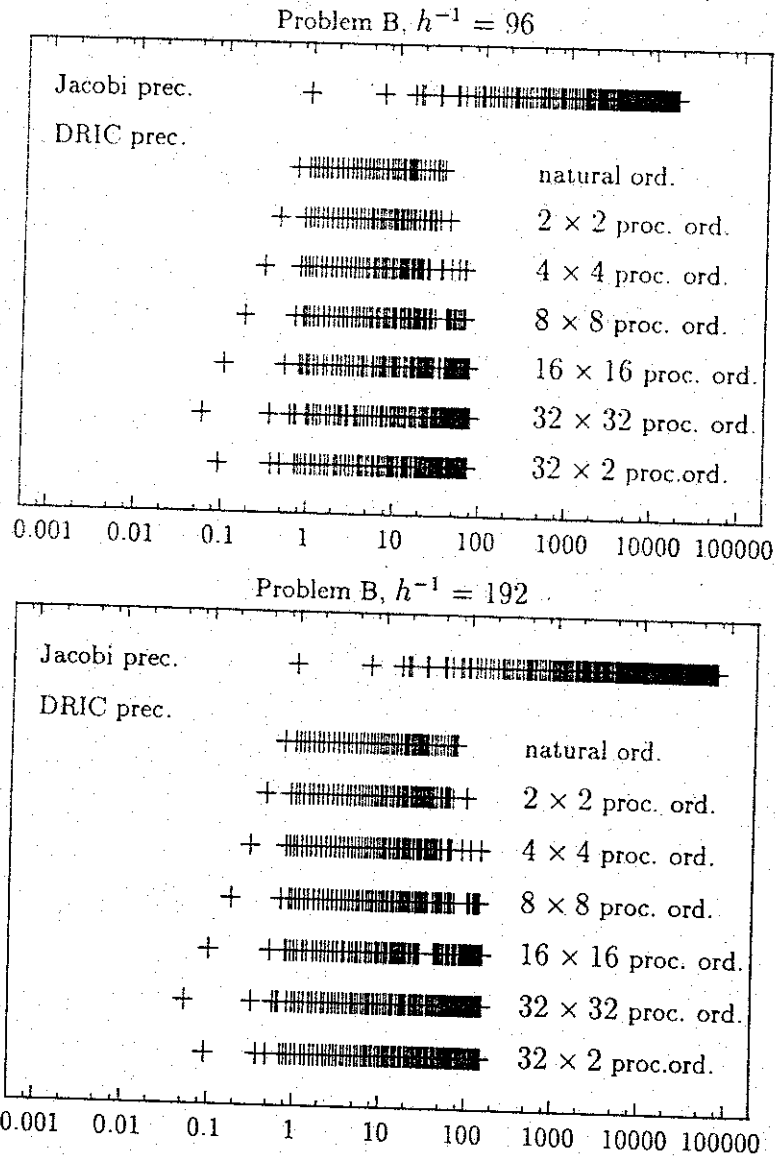


Figure 3: Ritz values for the DRIC preconditioning with $\alpha = h$, and Ritz values times n for the Jacobi preconditioning.



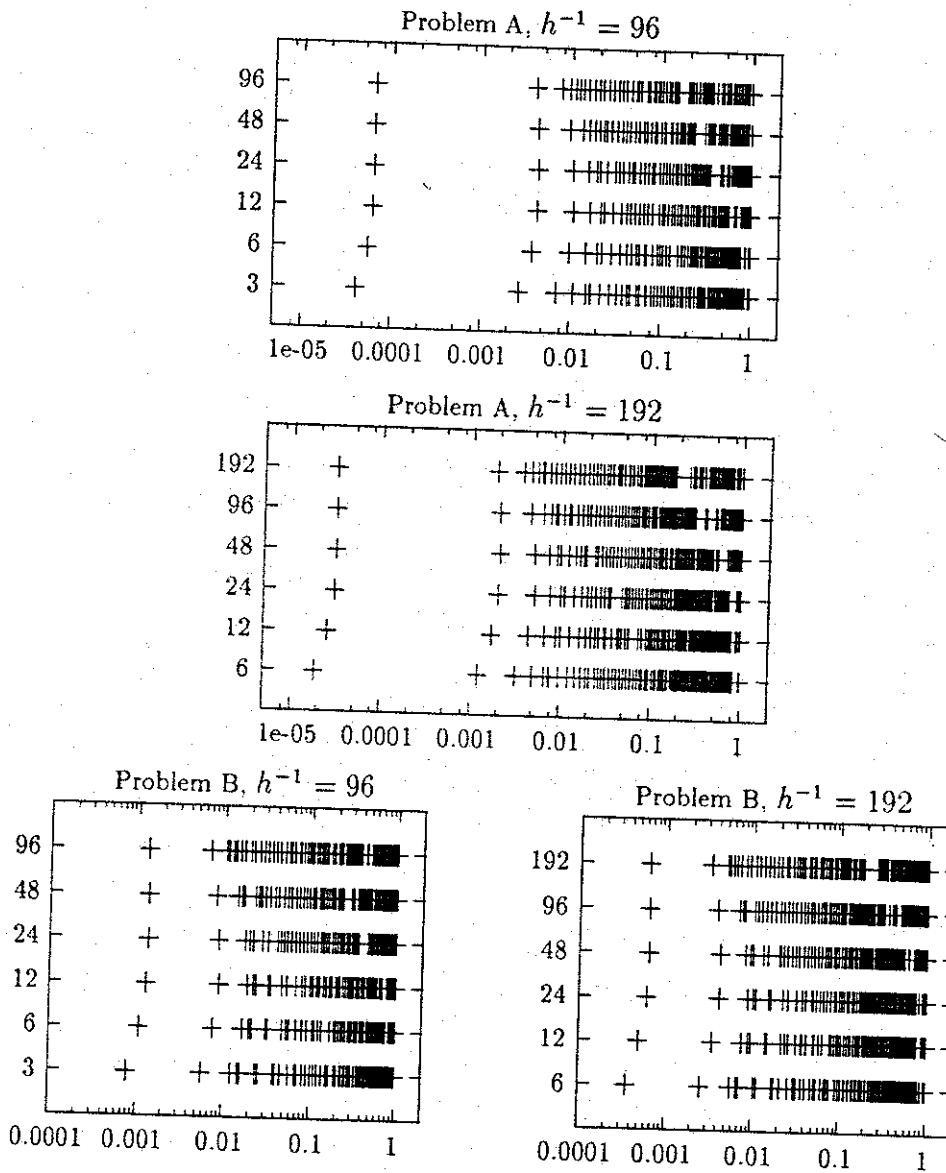


Figure 4: Ritz values for the DRIC preconditioning with the 16×16 processor ordering and various α ; the value of α^{-1} is indicated on the left axis; each distribution has been scaled so that the largest value is equal to 1.



3 Algebraic analysis of the coarse grid correction

We present in this section some algebraic results for preconditioners of the form (1.7), where B is any positive definite matrix.

We first recall Corollary 3.14 of [1] in the following lemma

LEMMA 1

Let C_1, C_2 be symmetric matrices.

$$\nu_i(C_1) + \nu_{\min}(C_2) \leq \nu_i(C_1 + C_2) \leq \nu_i(C_1) + \nu_{\max}(C_2) \quad (3.1)$$

THEOREM 1

Let A, B be $n \times n$ symmetric positive definite matrices, V an $n \times q$ rectangular matrix of rank q and $\hat{B}^{-1} = B^{-1} + V(V^t A V)^{-1} V^t$.

One has, for all $1 \leq i \leq n$,

$$\nu_i(B^{-1}A) \leq \nu_i(\hat{B}^{-1}A) \leq \nu_i(B^{-1}A) + 1. \quad (3.2)$$

Proof. Since $\nu_i(\hat{B}^{-1}A) = \nu_i(A^{\frac{1}{2}}\hat{B}^{-1}A^{\frac{1}{2}})$ and $\nu_i(B^{-1}A) = \nu_i(A^{\frac{1}{2}}B^{-1}A^{\frac{1}{2}})$, the results follows straightforwardly from Lemma 1 provided that $\nu_{\min}(A^{\frac{1}{2}}V(V^tAV)^{-1}V^tA^{\frac{1}{2}}) \geq 0$ and $\nu_{\max}(A^{\frac{1}{2}}V(V^tAV)^{-1}V^tA^{\frac{1}{2}}) \leq 1$. But

$$\left(A^{\frac{1}{2}}V(V^tAV)^{-1}V^tA^{\frac{1}{2}}\right)^2 = A^{\frac{1}{2}}V(V^tAV)^{-1}V^tA^{\frac{1}{2}}, \quad (3.3)$$

i.e. $A^{\frac{1}{2}}V(V^tAV)^{-1}V^tA^{\frac{1}{2}}$ is an orthogonal projector, whose only eigenvalues are therefore 0 and 1. ■

Let now $V_1 = [v_1^{(1)}, \dots, v_{q_1}^{(1)}]$ and $V_2 = [v_1^{(2)}, \dots, v_{q_2}^{(2)}]$. The next theorem shows that if $\text{span}\{v_1^{(1)}, \dots, v_{q_1}^{(1)}\} \subset \text{span}\{v_1^{(2)}, \dots, v_{q_2}^{(2)}\}$ (i.e. if $\mathcal{R}(V_1) \subset \mathcal{R}(V_2)$), then the eigenvalues of $\hat{B}_2^{-1}A$ are not smaller than the corresponding eigenvalues of $\hat{B}_1^{-1}A$, where $\hat{B}_i^{-1} = B^{-1} + V_i(V_i^tAV_i)^{-1}V_i^t$, $i = 1, 2$.

THEOREM 2

Let A, B be $n \times n$ symmetric positive definite matrices, V_1 a rectangular $n \times q_1$ matrix of rank q_1 and V_2 a rectangular $n \times q_2$ matrix of rank q_2 . Let

$$\hat{B}_i^{-1} = B^{-1} + V_i(V_i^tAV_i)^{-1}V_i^t, \quad i = 1, 2. \quad (3.4)$$

If $\mathcal{R}(V_1) \subset \mathcal{R}(V_2)$, then, for all $1 \leq i \leq n$,

$$\nu_i(\hat{B}_2^{-1}A) \geq \nu_i(\hat{B}_1^{-1}A). \quad (3.5)$$



Proof. The proposition holds if $V_2(V_2^t A V_2)^{-1} V_2^t - V_1(V_1^t A V_1)^{-1} V_1^t$ is nonnegative definite. But, since $\mathcal{R}(V_1) \subset \mathcal{R}(V_2)$, there exists some $q_2 \times q_1$ matrix Q of rank q_1 such that $V_1 = V_2 Q$.

Then, noting that $C = V_2^t A V_2$ is positive definite,

$$\begin{aligned} V_2(V_2^t A V_2)^{-1} V_2^t - V_1(V_1^t A V_1)^{-1} V_1^t &= V_2 \left(C^{-1} - Q(Q^t C Q)^{-1} Q^t \right) V_2^t \\ &= V_2 C^{-\frac{1}{2}} \left(I - C^{\frac{1}{2}} Q(Q^t C Q)^{-1} Q^t C^{\frac{1}{2}} \right) C^{-\frac{1}{2}} V_2^t. \end{aligned}$$

But $C^{\frac{1}{2}} Q(Q^t C Q)^{-1} Q^t C^{\frac{1}{2}}$ is an orthogonal projector whose only eigenvalues are 0 and 1, whence the required result. ■

Incidentally, the proof of Theorem 2 also shows that, if $\text{span}\{v_1^{(1)}, \dots, v_{q_1}^{(1)}\} = \text{span}\{v_1^{(2)}, \dots, v_{q_2}^{(2)}\}$, one has necessarily $\hat{B}_1 = \hat{B}_2$ (since $q_1 = q_2$, Q is invertible which implies $C^{\frac{1}{2}} Q(Q^t C Q)^{-1} Q^t C^{\frac{1}{2}} = I$).

The next theorem shows that, if $V = [v_1, \dots, v_q]$ is built with vectors v_i that are eigenvectors of $B^{-1}A$, these eigenvectors are still eigenvectors of $\hat{B}^{-1}A$ and the corresponding eigenvalues augmented of one unit, while the remaining eigenvectors & eigenvalues are not perturbed. A similar result is given in [5] for preconditioners of the form $(I + V V^t)$.

THEOREM 3

Let A, B be $n \times n$ symmetric nonnegative definite matrices, $V = [v_1, \dots, v_q]$ an $n \times q$ rectangular matrix of rank q , and $\hat{B}^{-1} = B^{-1} + V(V^t A V)^{-1} V^t$.

a) If, for some $1 \leq i \leq q$, v_i is an eigenvector of $B^{-1}A$ with eigenvalue λ_i :

$$B^{-1} A v_i = \lambda_i v_i, \quad (3.6)$$

then it is eigenvector of $\hat{B}^{-1}A$ with eigenvalue $\lambda_i + 1$:

$$\hat{B}^{-1} A v_i = (\lambda_i + 1) v_i. \quad (3.7)$$

b) If (3.6) holds for all $1 \leq i \leq q$, and if (v_i, λ_i) , $i = q+1, \dots, n$ denote the remaining eigenpairs of $B^{-1}A$, then the eigenpairs of $\hat{B}^{-1}A$ are $(v_i, \lambda_i + 1)$, $i = 1, \dots, q$ and (v_i, λ_i) , $i = q+1, \dots, n$.

Proof. For any v_i satisfying (3.6) and belonging to $\mathcal{R}(V)$, there exists a $q \times 1$ vector w_i such that $v_i = V w_i$. Then

$$\hat{B}^{-1} A v_i = \lambda_i v_i + V(V^t A V)^{-1} V^t A V w_i = (\lambda_i + 1) v_i, \quad (3.8)$$

whence a). To prove b), it is enough to note that, since $A v_i = \lambda B v_i$ for $i = 1, \dots, n$, the v_i are mutually A -orthogonal: $v_i^t A v_j = 0$ for $i \neq j$. Then, $V^t A v_i = 0$ for



$i = q + 1, \dots, n$, implying $\hat{B}^{-1} A v_i = B^{-1} A v_i = \lambda_i v_i$. ■

Some issues of the preceding result are gathered in the following corollary.

COROLLARY 1

Let A, B be $n \times n$ symmetric positive definite matrices, V an $n \times q$ rectangular matrix of rank q and $\hat{B}^{-1} = B^{-1} + V(V^t A V)^{-1} V^t$.

Let $z_i, i = 1, \dots, r$ be the eigenvectors associated to the r first eigenvalues of $B^{-1} A$ in increasing order, i.e. $B^{-1} A z_i = \nu_i(B^{-1} A) z_i, i = 1, \dots, r$.

If $\{z_1, \dots, z_r\} \subset \mathcal{R}(V)$, then, for any eigenvalue $\nu_k(\hat{B}^{-1} A)$ of $\hat{B}^{-1} A$,

$$\min(1 + \nu_1(B^{-1} A), \nu_{r+1}(B^{-1} A)) \leq \nu_k(\hat{B}^{-1} A) \leq \nu_{\max}(B^{-1} A) + 1. \quad (3.9)$$

Proof. The upper bound is that in Theorem 1, while the lower bound follows straightforwardly from Theorem 3 when $V = [z_1, \dots, z_r]$. The general case where $\mathcal{R}(V)$ is larger than $\text{span}\{r_1, \dots, r_z\}$ follows then from Theorem 2. ■

These results illustrate that one may shift to the right the smallest eigenvalues if one succeeds to include sufficiently good approximations of the associated eigenvectors in $\text{span}\{v_1, \dots, v_q\}$.

A somewhat similar result is obtained in [5] for preconditioners of the form $(I + V V^t)$ which are introduced through a bordering technique. The drawback in this approach is that the amplitude of the shifts depends on the scaling of the vectors in V , which has to be adjusted in function of the values of the eigenvalues to be shifted.

Our approach, besides including directly the case of already preconditioned matrices, also provides automatically an appropriate scaling through the term $(V^t A V)^{-1}$. Indeed, the largest eigenvalue cannot be significantly perturbed when it is much larger than 1, while the smallest eigenvalues will be efficiently shifted if there is a relatively dense distribution of eigenvalues between 1 and 2. Looking at the spectra depicted in Section 2, this matches perfectly our needs.

Note that this may be extended to other types of eigenvalue distribution by considering $\hat{B}^{-1} = B^{-1} + \lambda V(V^t A V)^{-1} V^t$; in any case, only a rough idea of the spectrum is needed to choose an appropriate λ .

4 Incomplete factorization with coarse grid correction

We now consider the coarse grid correction (1.7) applied to DRIC preconditioners.

As shown in Section 2, they do cluster the eigenvalues relatively well, apart from a few nodes, to which are associated small isolated eigenvalues. (This is so for the application to discrete second order elliptic PDEs.) The results of Section 3 show then



that the considered corrector term, with $V = [v_1, \dots, v_p]$ containing only a few vectors, is sufficient to shift to the right these eigenvalues, provided that $\text{span}\{v_1, \dots, v_q\}$ contains sufficiently good approximations of the associated eigenvectors. Since these eigenvectors are expected to be "smooth", it is natural to choose for v_i , $i = 1, \dots, q$, the vectors which interpolate the finite element basis functions associated to a coarse mesh.

Such a coarse grid correction is also used in additive Schwartz algorithms (see [27], for instance). In these methods however, the size of the basis is increased so that the coarse mesh size correspond to the subdomains diameter.

As observed in Section 2, the number of eigenvalues to be shifted to the right increases with the number of processors, but not in a dramatic way. Hence, we may hope obtain satisfying results while keeping the size of V reasonable. This is important in practice because the cost of computing $w_2 = (V^t A V)^{-1} w_1$ may become rapidly prohibitive in case of massive parallelism, if one is required to have q approximately equal to the number of processors.

Note that the computation of $w_1 = V^t x$ and $y = V w_2$ is not problematic because V become sparser as the coarse mesh is refined. More precisely, if r is the number of nodes per element in the considered finite element scheme, V will contain at most r nonzero per row, and each of these computation will require not more then rn multiply-add. These computations are also well suited for parallelization.

In the case of regular rectangular grids with a $p_x \times p_y$ processor ordering as described in Section 2, we propose more specifically to use a $q_x \times q_y$ coarse mesh of bilinear finite elements, locating preferably the element boundaries at (some of) the lines corresponding to gridpoints ordered first (depicted by squares on Fig. 1).

The lowest eigenvalues are indeed badly influenced by the presence of such lines, and we expect that irregularities in the associate eigenvectors, if any, will be located on them. This choice is also the most reasonable in view of a parallel implementation.

Some irregularities may also occur at lines corresponding to jumps in the PDE coefficients, but it would be unpractical to have to adapt the coarse mesh in function of these jumps. The small isolated eigenvalues already present with the natural or the 2×2 processor ordering may therefore not efficiently be shifted, the coarse mesh being more designed to compensate for the influence of the ordering. To preserve the significance of our results, we deliberately chose Problems A and B such that the jumps in the PDE coefficients never coincide with processor boundaries for the considered processor grids.

Letting $x = x_k$, $k = 0, \dots, q_x$ and $y = y_\ell$, $\ell = 0, \dots, q_y$ be the resulting element boundaries ($x = x_0$, $x = x_{q_x}$, $y = y_0$ and $y = y_{q_y}$ being the external boundaries of the domain), we have then $(q_x + 1)(q_y + 1)$ gridpoints in the coarse mesh, located at (x_k, y_ℓ) , $k = 0, \dots, q_x$, $\ell = 0, \dots, q_y$. The associate finite element basis functions are [2]:



$$\Phi_{k\ell}(x, y) = \begin{cases} \left(1 - \frac{x-x_k}{x_{k+1}-x_k}\right) \left(1 - \frac{y-y_\ell}{y_{\ell+1}-y_\ell}\right) & \text{if } x_k \leq x \leq x_{k+1}, y_\ell \leq y \leq y_{\ell+1} \\ & \text{and } k < q_x, \ell < q_y \\ \left(1 - \frac{x_k-x}{x_k-x_{k-1}}\right) \left(1 - \frac{y-y_\ell}{y_{\ell+1}-y_\ell}\right) & \text{if } x_{k-1} \leq x \leq x_k, y_\ell \leq y \leq y_{\ell+1} \\ & \text{and } k > 0, \ell < q_y \\ \left(1 - \frac{x-x_k}{x_{k+1}-x_k}\right) \left(1 - \frac{y_\ell-y}{y_\ell-y_{\ell-1}}\right) & \text{if } x_k \leq x \leq x_{k+1}, y_{\ell-1} \leq y \leq y_\ell \\ & \text{and } k < q_x, \ell > 0 \\ \left(1 - \frac{x_k-x}{x_k-x_{k-1}}\right) \left(1 - \frac{y_\ell-y}{y_\ell-y_{\ell-1}}\right) & \text{if } x_{k-1} \leq x \leq x_k, y_{\ell-1} \leq y \leq y_\ell \\ & \text{and } k > 0, \ell > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

To each $\Phi_{k\ell}$ corresponds a vector $v_{k\ell}$ in V , such that, for any node j in the fine grid, $(v_{k\ell})_j = \Phi_{k\ell}(\xi_j, \eta_j)$, where (ξ_j, η_j) are the coordinates of the gridpoint corresponding to j .

Note that the fine grid nodes located on external boundaries with Dirichlet boundary conditions are removed from the actual system to solve. In the numerical experiments below, we also removed the vectors in V associated to coarse gridpoints on such boundaries, motivated by the fact that the eigenvectors associated to small eigenvalues are generally near zero in the neighbouring regions. By Theorem 2, the eigenvalues are then a lower bound on the eigenvalues which one would obtain with the complete basis.

Numerical results

We applied the coarse grid correction (1.7) as described above to the DRIC preconditioner, and tested it on the Problems *A* and *B* introduced in Section 2. As there, we computed the Ritz values after convergence, and plotted them for several situations.

In a first set of experiments (see Figures 5 and 6), we considered the 16×16 processor ordering, and checked the influence of a 2×2 and a 4×4 coarse mesh correction on the eigenvalue distribution, in both cases of using the rule (2.4) and the rule (2.7) for the DRIC preconditioner.

One sees that a significant improvement is already obtained with the 2×2 coarse mesh, while all small isolated eigenvalues are efficiently shifted with 4×4 coarse mesh. The benefit seems superior when using the rule (2.7), probably because more eigenvalues are then smaller than 1 and well separated from the dense part of the spectrum.

In a second set of experiments, we fix the size of the coarse mesh to either 2×2 (Figure 7) or 4×4 (Figure 8), and check now the evolution of the eigenvalue distribution with the ordering while using the rule (2.7) for the DRIC parameter; the distribution associated to the natural and the 2×2 processor ordering without corrector term are also recalled as reference. One sees that the influence of the ordering on the eigenvalue distribution is now well mastered, even till massive parallelism.

This conclusion is corroborated by Table 1, where we report the number of conjugate gradient iterations necessary to reduce the relative residual error $\|r_k\| / \|r_0\|$



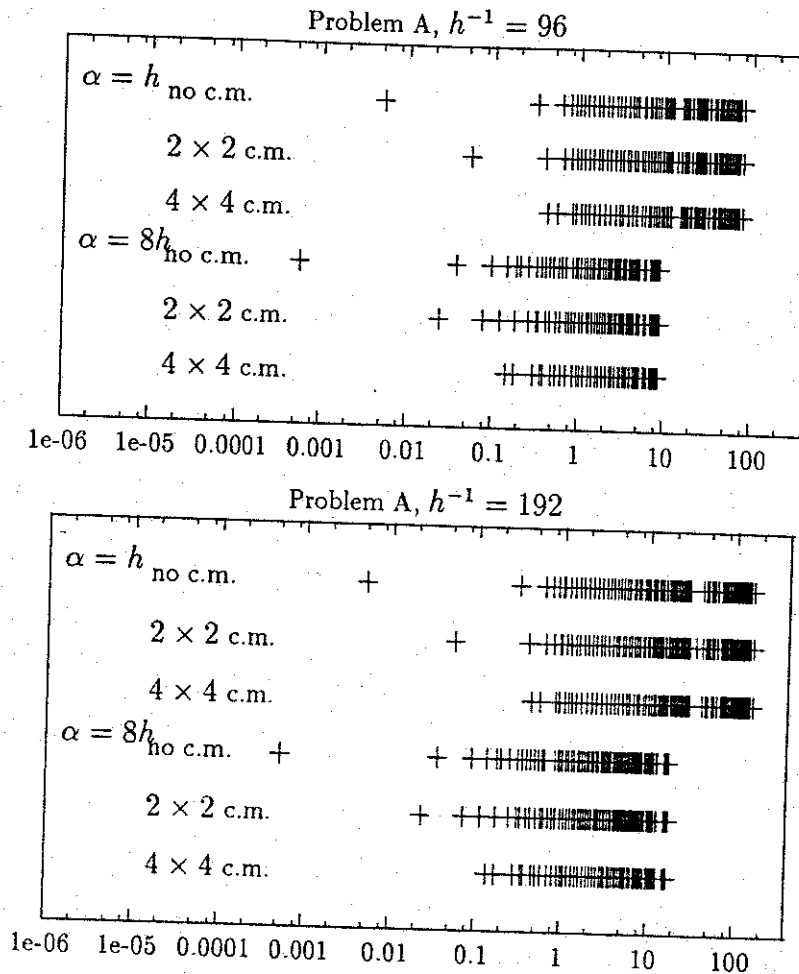


Figure 5: Ritz values for the preconditioning (1.7), where B is the DRIC preconditioner with the 16×16 processor ordering, and the corrector term either zero (no c.m.) or based on a $q_x \times q_y$ coarse finite element mesh ($q_x \times q_y$ c.m.).

below 10^{-7} while using zero initial approximation. One may check the very good scalability of the method when using the rule (2.7) together with the 4×4 coarse mesh acceleration.

The results are less favorable when using the rule (2.4), perhaps because part of the increase of the number of iterations with the number of processors has then to be explained by the increase of the largest eigenvalues, which our coarse grid correction cannot fight. Besides its intrinsic merits, the rule (2.7), by increasing α , decreases the upper bound (2.1), and thus ensures that the source of any deterioration in the conditioning properties is located at the lower end of the spectrum, and may therefore be compensated by a corrector term of the type investigated here.



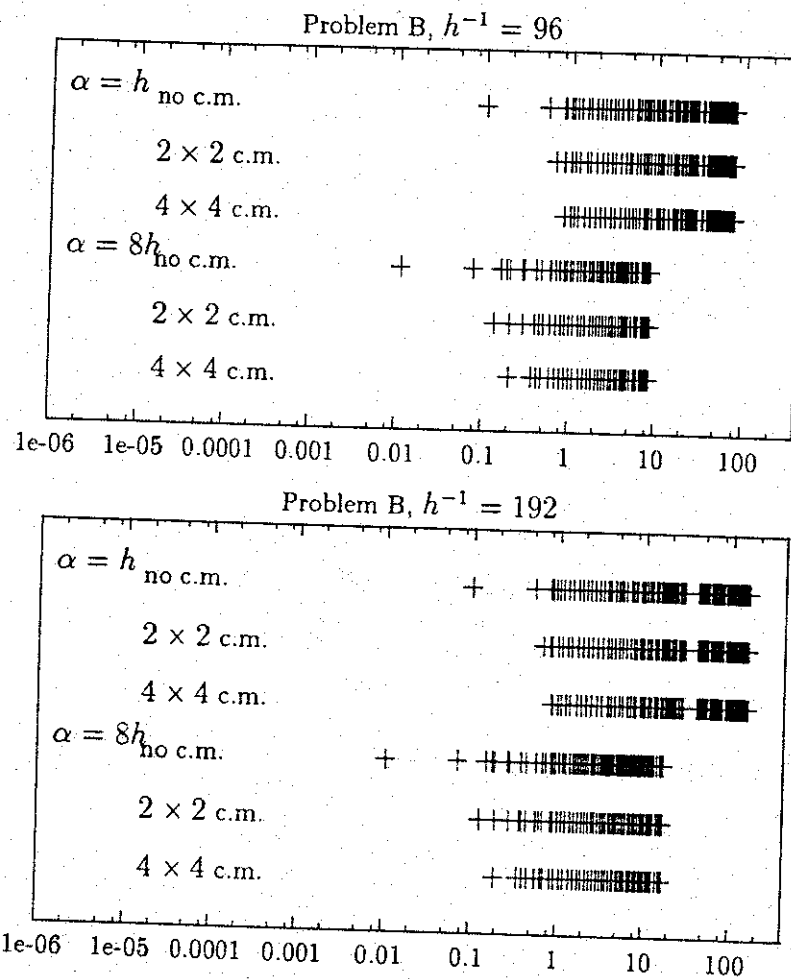


Figure 6: Ritz values for the preconditioning (1.7), where B is the DRIC preconditioner with the 16×16 processor ordering, and the corrector term either zero (no c.m.) or based on a $q_x \times q_y$ coarse finite element mesh ($q_x \times q_y$ c.m.).

Acknowledgments

This work presents research results of the Belgian Incentive Program "Information Technology" - Computer Science of the future, initiated by the Belgian State - Prime Minister's Service - Federal Office for Scientific, Technical and Cultural Affairs (Contract No. IT/IF/14). The Scientific responsibility is assumed by the authors.

This work was also supported by IBM through a research contract between ULB and IBM.



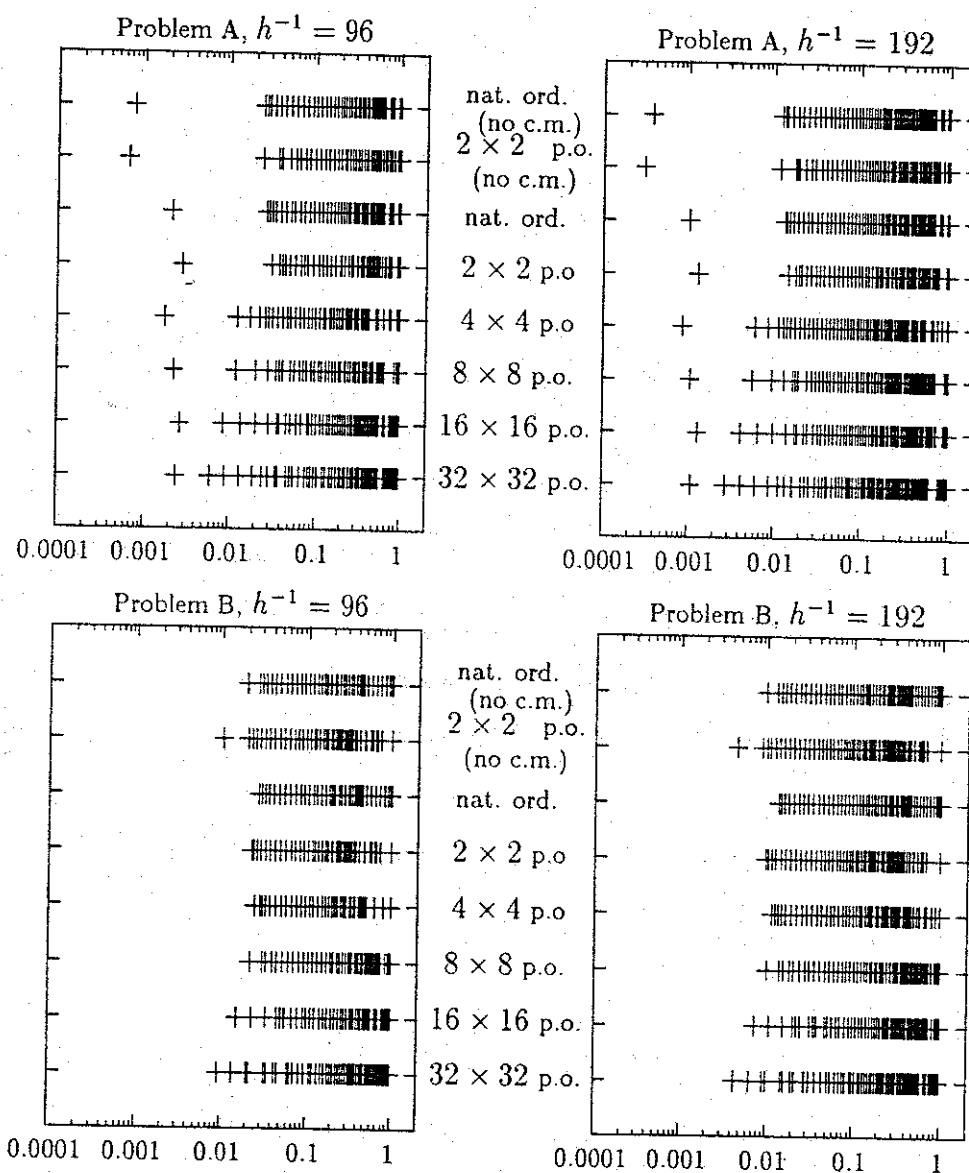
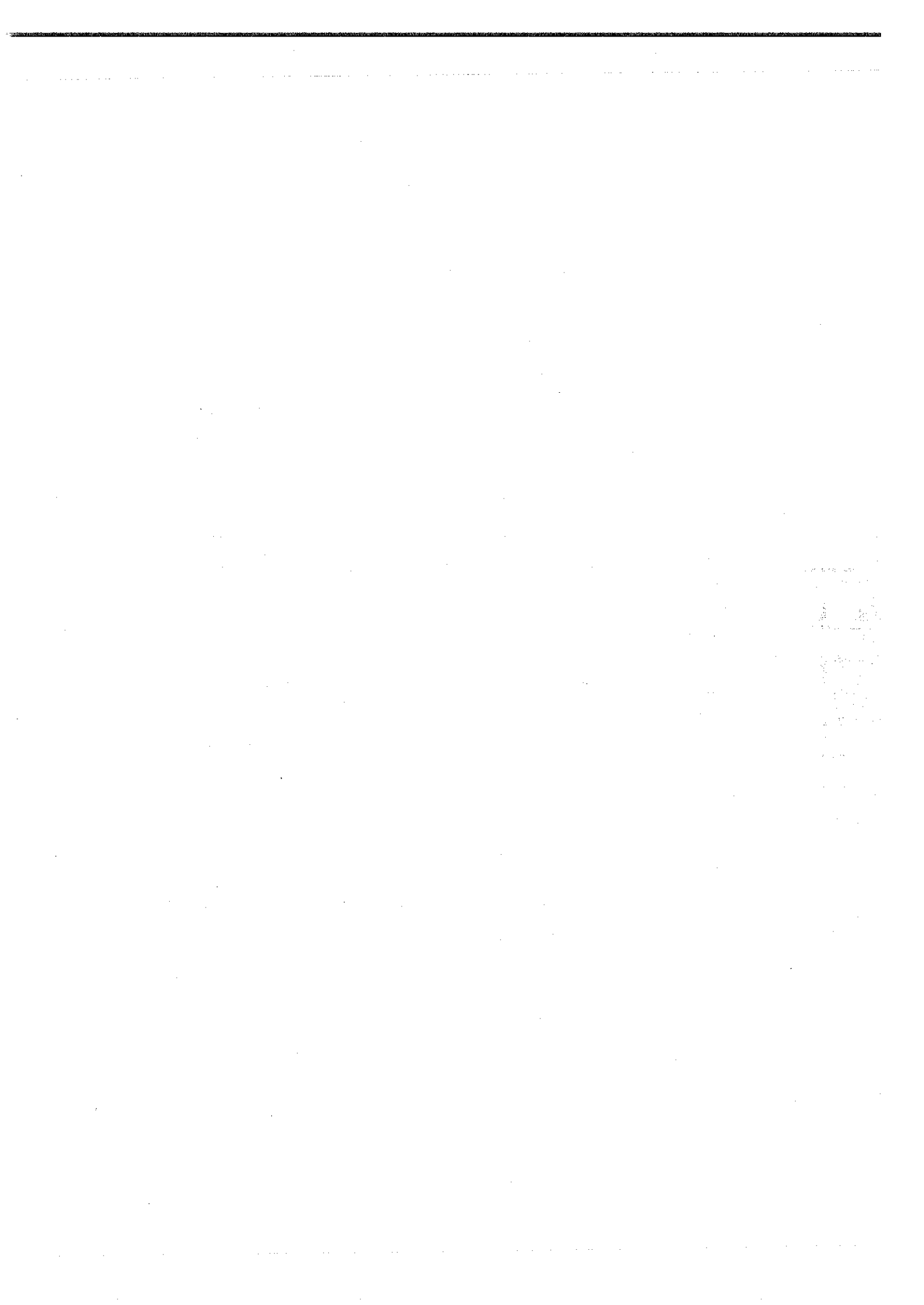


Figure 7: Ritz values for the preconditioning (1.7), where B is the DRIC preconditioner with $\alpha = h$ for the natural ordering and $\alpha = mh$ for the $2m \times 2m$ processor ordering, and the corrector based on a 2×2 coarse finite element mesh, except whenever specified "no c.m.", in which cases it is zero; each distribution has been scaled so that the largest value is equal to 1.



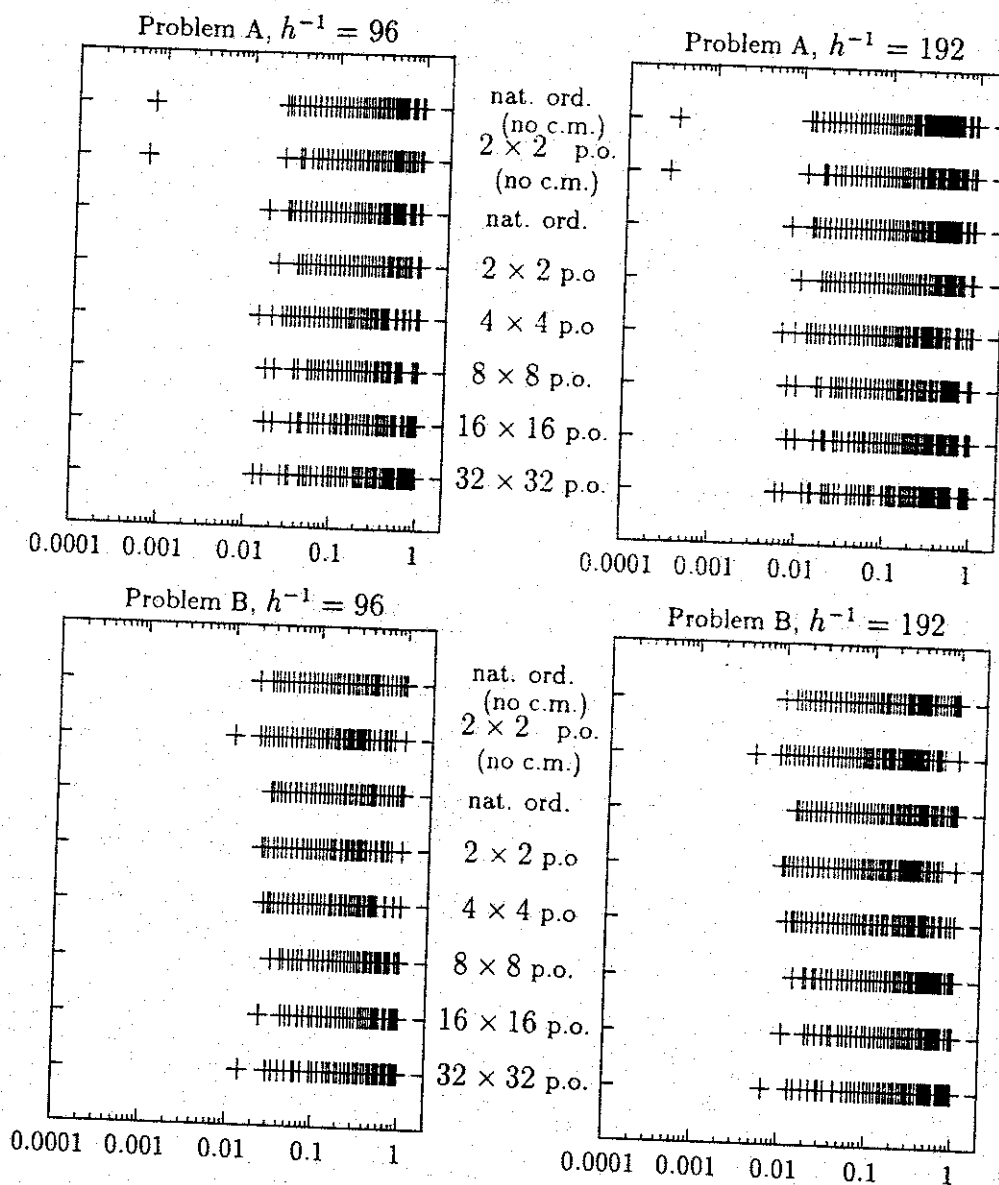


Figure 8: Ritz values for the preconditioning (1.7), where B is the DRIC preconditioner with $\alpha = h$ for the natural ordering and $\alpha = mh$ for the $2m \times 2m$ processor ordering, and the corrector based on a 4×4 coarse finite element mesh, except whenever specified "no c.m.", in which cases it is zero; each distribution has been scaled so that the largest value is equal to 1.



Problem A, $h^{-1} = 96$							
ordering (processor grid) :	nat. ord.	2×2	4×4	8×8	16×16	32×32	
$\alpha = h$	no c.m.	59	54	74	83	105	133
	2×2 c.m.	60	54	74	81	105	126
	4×4 c.m.	53	47	62	69	89	105
$\alpha = \frac{\sqrt{p}}{2}h$	no c.m.		54	67	70	83	103
	2×2 c.m.		54	64	64	74	87
	4×4 c.m.		47	53	50	55	65

Problem A, $h^{-1} = 192$							
ordering (processor grid) :	nat. ord.	2×2	4×4	8×8	16×16	32×32	
$\alpha = h$	no c.m.	87	80	117	130	158	203
	2×2 c.m.	90	81	115	125	157	189
	4×4 c.m.	78	69	98	109	131	155
$\alpha = \frac{\sqrt{p}}{2}h$	no c.m.		80	103	108	122	154
	2×2 c.m.		81	97	96	109	131
	4×4 c.m.		69	80	76	82	97

Problem B, $h^{-1} = 96$							
ordering (processor grid) :	nat. ord.	2×2	4×4	8×8	16×16	32×32	
$\alpha = h$	no c.m.	52	53	66	82	104	128
	2×2 c.m.	48	48	56	72	87	105
	4×4 c.m.	48	47	56	70	82	93
$\alpha = \frac{\sqrt{p}}{2}h$	no c.m.		53	60	65	75	92
	2×2 c.m.		48	47	53	55	70
	4×4 c.m.		47	48	48	47	55

Problem B, $h^{-1} = 192$							
ordering (processor grid) :	nat. ord.	2×2	4×4	8×8	16×16	32×32	
$\alpha = h$	no c.m.	77	80	103	126	155	193
	2×2 c.m.	72	72	87	111	130	155
	4×4 c.m.	73	71	87	106	123	139
$\alpha = \frac{\sqrt{p}}{2}h$	no c.m.		80	91	99	111	138
	2×2 c.m.		72	75	80	83	109
	4×4 c.m.		71	74	71	70	82

Table 1: Numbers of iterations for the preconditioning (1.7), where B is the DRIC preconditioner and the corrector term either zero (no c.m.) or based on a $q_x \times q_y$ coarse mesh ($q_x \times q_y$ c.m.); $\alpha = \frac{\sqrt{p}}{2}h$ means $\alpha = mh$ for the $2m \times 2m$ processor ordering.



References

- [1] O. AXELSSON, *Iterative Solution Methods*, University Press, Cambridge, 1994.
- [2] O. AXELSSON AND V. A. BARKER, *Finite Element Solution of Boundary Value Problems. Theory and Computation*, Academic Press, New York, 1984.
- [3] O. AXELSSON AND G. LINDSKOG, *On the eigenvalue distribution of a class of preconditioning methods*, Numer. Math., 48 (1986), pp. 479-498.
- [4] O. AXELSSON AND G. LINDSKOG, *On the rate of convergence of the preconditioned conjugate gradient method*, Numer. Math., 48 (1986), pp. 499-523.
- [5] O. AXELSSON, M. NEYTCHEVA, AND B. POLMAN, *The bordering method as a preconditioning method*, Technical Report Report 9348, Department of Mathematics, Catholic University, Nijmegen, The Netherlands, 1993.
- [6] R. BEAUWENS, *Lower eigenvalue bounds for pencils of matrices*, Lin. Alg. Appl., 85 (1987), pp. 101-119.
- [7] R. BEAUWENS, *Modified incomplete factorization strategies*, in Preconditioned Conjugate Gradient Methods, O. Axelsson and L. Kolotilina, eds., Lectures Notes in Mathematics No. 1457, Springer-Verlag, 1990, pp. 1-16.
- [8] R. BEAUWENS, *Approximate factorizations with modified S/P consistently ordered M-factors*, J. Num. Lin. Alg. with Appl., 1 (1994), pp. 3-17.
- [9] R. BEAUWENS AND R. WILMET, *Conditioning analysis of positive definite matrices by approximate factorizations*, J. Comput. Appl. Math., 26 (1989), pp. 257-269.
- [10] I. S. DUFF AND G. A. MEURANT, *The effect of ordering on preconditioned conjugate gradients*, BIT, 29 (1989), pp. 635-657.
- [11] A. GREENBAUM, *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences*, Lin. Alg. Appl., 113 (1989), pp. 7-63.
- [12] A. KUTCHEROV AND M. MAKAROV, *An approximate factorization method for solving discrete elliptic problems on stretched domains*, J. Numer. Lin. Alg. Appl., 1 (1992), pp. 1-26.
- [13] M. M. MAGOLU, *Lower eigenvalue bounds for singular pencils of matrices*, J. Comput. Appl. Math., 39 (1992), pp. 329-351.
- [14] P. MANNEBACK AND J. QIN, *Algorithmic on a distributed memory MIMD computer: a case study*, in Proceedings of Transputers'92, M. B. et al., ed., Amsterdam, 1992, IOS Press, pp. 172-178.



- [15] Y. NOTAY, *Solving positive (semi)definite linear systems by preconditioned iterative methods*, in Preconditioned Conjugate Gradient Methods, O. Axelsson and L. Kolotilina, eds., Lectures Notes in Mathematics No. 1457, Springer-Verlag, 1990, pp. 105-125.
- [16] Y. NOTAY, *On the robustness of modified incomplete factorization methods*, Inter. J. Comp. Math., 40 (1992), pp. 121-141.
- [17] Y. NOTAY, *On the convergence rate of the conjugate gradients in presence of rounding errors*, Numer. Math., 65 (1993), pp. 301-317.
- [18] Y. NOTAY, *DRIC : a dynamic version of the RIC method*, Journ. Num. Lin. Alg. with Appl., 1 (1994), pp. 511-532.
- [19] Y. NOTAY, *An efficient parallel discrete PDE solver*. Parallel Computing, to appear, 1995.
- [20] Y. NOTAY, *Parallel implementation of preconditioned iterative schemes by means of overlapping decomposition*. submitted for publication, 1995.
- [21] Y. NOTAY AND Z. OULD AMAR, *A nearly optimal preconditioning based on recursive red-black orderings*. J. Num. Lin. Alg. with Appl., submitted for publication, 1995.
- [22] J. ORTEGA, *Orderings for conjugate gradient preconditionings*, SIAM J. Optimization, 1 (1990), pp. 565-582.
- [23] Z. OULD AMAR. *An efficient preconditioning method for SIMD distributed memory-computers*, Tech. Rep. NM-IBM 95-02, Université Libre de Bruxelles, 1995.
- [24] P. SAINT-GEORGES, G. WARZEE, R. BEAUWENS, AND Y. NOTAY, *Development of an efficient iterative solver for linear systems in FE structural analysis*, in in Transaction of the SMiRT 12 Conference in Stuttgart, K. Kussmaul, ed., North Holland, 1993, pp. 201-206. Proceedings of the 12th International Conference on Structural Mechanics in Reactor Technology, Stuttgart, August 15-20, 1993.
- [25] A. VAN DER SLUIS AND H. A. VAN DER VORST, *The rate of convergence of conjugate gradients*, Numer. Math., 48 (1986), pp. 543-560.
- [26] T. WASHIO AND K. HAYAMI, *Parallel block preconditioning based on SSOR and MILU*, Journ. Num. Lin. Alg. with Appl., 1 (1994), pp. 533-553.
- [27] J. XU, *Iterative methods by space decomposition and subspace correction*, SIAM Review, 34 (1992), pp. 581-613.